
Generalized Top- k Mallows Model for Ranked Choices

Shahrzad Haddadan
Rutgers Business School
Piscataway, NJ
shaddadan@business.rutgers.edu

Sara Ahmadian
Google research
Seattle, WA
sahmadian@gmail.com

Abstract

The classic Mallows model is a foundational tool for modeling user preferences. However, it has limitations in capturing real-world scenarios, where users often focus only on a limited set of preferred items and are indifferent to the rest. To address this, extensions such as the top- k Mallows model have been proposed, aligning better with practical applications. In this paper, we address several challenges related to the generalized top- k Mallows model, with a focus on analyzing buyer choices. Our key contributions are: (1) a novel sampling scheme tailored to generalized top- k Mallows models, (2) an efficient algorithm for computing choice probabilities under this model, and (3) an active learning algorithm for estimating the model parameters from observed choice data. These contributions provide new tools for analysis and prediction in critical decision-making scenarios. We present a rigorous mathematical analysis for the performance of our algorithms. Furthermore, through extensive experiments on synthetic data and real-world data, we demonstrate the scalability and accuracy of our proposed methods, and we compare the predictive power of Mallows model for top- k lists compared to the simpler Multinomial Logit model.

1 Introduction

User preferences over a set of alternative items play a crucial role in various decision-making scenarios. A key concept in this context is the *choices* customers make when presented with a subset of alternatives, referred to as an *assortment*, drawn from a larger pool of items. The mathematical modeling of preferences and choices is both essential and challenging. It enables researchers and business leaders to analyze and predict customer behavior, thereby informing more effective decision-making. Probabilistic models over rankings, such as the Plackett-Luce (PL) and Mallows models, are widely used to represent user preferences. Based on the Plackett-Luce (PL) model, the Multinomial Logit (MNL) model has been suggested for modeling choice, and it has been extensively applied in choice modeling due to its simplicity and interpretability.

The Mallows model (Mallows, 1957) is a distance-based probability distribution defined over permutations and is reminiscent of the Gaussian distribution for scalar variables. It has been used successfully to model preferences, particularly where ranking data is available. Recent work has demonstrated the high predictive power of the Mallows model in modeling choices (Désir et al., 2023, 2021), sparking follow-up research into revenue management and assortment optimization in this framework Désir et al. (2021, 2023); Feng & Tang (2022); Rieger & Segev (2023). One key challenge in applying the Mallows model more broadly is that its classic definition is restricted to full permutations.

In many real-world scenarios, however, user preferences are often observed as *top- k lists* rather than full rankings. For example, many platforms display only a subset of items to customers and ask them to rank a fixed number of preferred items. In recommender systems, advertising platforms, search engines, news aggregators, and social media friend suggestions, users are typically shown only the

top- k most relevant items, rather than an exhaustive list. Likewise, users often express preferences for a limited number of favorite items and show indifference toward the remainder.

This recurring structure in many applications has motivated the extension of the Mallows model to handle top- k lists. The design of algorithm for this variant is significantly more complex than in the traditional permutation-based setting, leading to a growing body of research focused on learning, inference, and aggregation under the top- k Mallows model (Lu & Boutilier, 2011; Chierichetti et al., 2018a; Collas & Irurozki, 2021; Vitelli et al., 2018; Fotakis et al., 2021; Boehmer et al., 2023; Goibert et al., 2023; Awadelkarim & Ugander, 2024; Qian & Philip, 2019; Akbari & Escobedo, 2022).

In this paper, we propose employing a generalized Mallows model for top- k lists for choice modeling, and we develop efficient algorithms related to generating samples from it, learning its parameters and finding choice probabilities. Our approach addresses more realistic preference structures, where users are unlikely to hold complete rankings over a large set of items, an assumption that is often impractical in real-world applications.

1.1 Related Work

In this section we present relevant related work on choice modeling and Mallows model.

Choice modeling Various probabilistic models have been developed to capture choice behavior, with the Multinomial Logit (MNL) (Bradley & Terry, 1952) model being the most widely used due to its simplicity and interpretability. In the MNL model each product is assigned a positive score or weight, and the probability of selecting a product from an assortment is proportional to its score. Importantly, the MNL model satisfies Luce’s Choice Axiom, also known as the Independence of Irrelevant Alternatives (IIA). While this property makes the MNL model analytically convenient, it also limits its expressiveness in capturing more complex choice behaviors.

To overcome the limitations of MNL, the mixture MNL model (also known as mixed logit) was introduced and popularized by McFadden & Train (2000). Learning the parameters of a mixed MNL model from observed choices—where each observation is an assortment and a chosen item—is a key challenge. Early approaches used heuristics based on maximum likelihood estimation (Dempster et al., 1977), and more recent work has provided statistically rigorous methods with provable sample complexity guarantees (Chierichetti et al., 2018b; Oh & Shah, 2014; Ma et al., 2022).

Beyond MNL-based models, other frameworks such as the Mallows-based choice model (Désir et al., 2021) and Markov chain-based models (Blanchet et al., 2016) have been proposed. These models offer greater flexibility but come with added complexity: tasks that are straightforward in MNL, such as computing choice probabilities or generating samples become substantially more challenging.

Mallows model The Mallows model (MM), originally introduced by Mallows (1957), defines a distance-based distribution over full rankings, where the probability of a permutation decays exponentially with its distance from a central (ground-truth) ranking. This property has made MM a useful foundation for preference modeling in machine learning. To better accommodate real-world data—where users often provide partial rather than full rankings—several extensions have been proposed. Early work by Fligner & Verducci (1986) and Lebanon & Mao (2008) adapted MM to top- k rankings. Later Chierichetti et al. (2018a) proposed a new distance measure on top- k lists and defined a parametrized Mallows model for top- k lists based on it (TopKMM).

One of the strengths of the standard Mallows model on permutations is the closed-form expression of its normalization constant which enables the design of several algorithms. For instance, the Repeated Insertion Method (RIM) (Doignon et al., 2004) allows efficient sampling from the full-ranking Mallows model. However, this tractability does not extend to TopKMM: computing probabilities or generating samples becomes non-trivial due to the lack of a closed-form normalizing constant. While Chierichetti et al. (2018a) proposed a dynamic programming approach to generate samples, they highlighted the absence of an RIM-like sampling method for top- k lists as an open problem.

Désir et al. (2021) proposed leveraging the Mallows model to capture choice behavior, demonstrating improved predictive accuracy over traditional models like the Multinomial Logit (MNL). However, a key challenge in applying the Mallows model to choice modeling lies in computing choice probabilities—i.e., the likelihood that a given item is selected from an assortment. Unlike MNL, where such probabilities have closed-form solutions the Mallows model does not admit such tractable computation in general. To address this, some works have introduced Mallows-like models designed to simplify choice probabilities calculations (Feng & Tang, 2022). In contrast, Désir et al. (2023)

tackle the original model directly and develop a dynamic programming approach for computing choice probabilities by leveraging key ideas from the Repeated Insertion Method (RIM).

Learning the parameters of the Mallows model or its mixtures from full permutation, or top- k list¹, samples has been extensively studied in prior work (Liu & Moitra, 2018a; Braverman & Mossel, 2008; Awasthi et al., 2014; Seshadri et al., 2020; Tang, 2019; Collas & Irurozki, 2021; Akbari & Escobedo, 2022; Liu & Moitra, 2018b; Chierichetti et al., 2015). Some studies focus on learning these parameters from historical data, while others consider an *active learning* setting, where upon each consumer's arrival, the platform adaptively selects the assortment of items to offer based on past observations Susan et al. (2022). In contrast, learning model parameters from partial observations presents a greater challenge. Several works address this problem by studying learning from pairwise comparisons Lu & Boutilier (2014); Vitelli et al. (2018); Tang (2019). A less explored problem is estimating the model parameters when only the users' choices from offered assortments of arbitrary sizes are observed. Existing methods in this setting often lack finite-sample complexity guarantees, limiting their theoretical robustness.

1.2 Summary of Contributions

In this paper, we focus on Mallows model on top- k lists, and we make several algorithmic contributions for the usage of this model. In order to obtain our results in full generality, we extend Chierichetti et al. (2018a)'s model on Mallows model for top- k list to a generalized version by associating a weight to each product. Our algorithmic contribution are listed as follows:

1. *Sampling*: PROFILE BASED REPEATED INSERTION METHOD (PRIM): The *Repeated Insertion Method* (RIM) is a common method for sampling permutations in the classic Mallows model. However, extending this method to the top- k Mallows model remains an open problem. Our proposed algorithm, PRIM, offers similar functionality to RIM and improves upon the prior dynamic programming approach by Chierichetti et al. (2018a), reducing the runtime from $O(k^2 4^k + k^2 \log n)$ to $O(k 2^k + k^2 \log n)$.
2. *Choice Probabilities*: DYNAMIC PROGRAMMING FOR CHOICE PROBABILITIES (DYPCHIP): DYPCHIP is an algorithm to calculate the choice probabilities when they are inferred from a Mallows model on top- k lists. This result extends the work of Désir et al. (2021) who consider the classic Mallows model for permutations.
3. *Learning the Center*: BUILD CENTER FROM CHOICES (BUCCHOI): BUCCHOI is an *active learning algorithm* designed to learn the center of a top- k Mallows model distribution. It operates by presenting assortments of a specified size r to customers and, based on their observed choices, infers both the ranking of the center and the size of the center k .

The accuracy and complexity of these algorithms are demonstrated through rigorous mathematical analysis as well as experiments on real-world and synthetic data.

Furthermore, we apply these algorithms and fit a top- k Mallows model to a real-world publicly available dataset including users' preferences over 100 sushi types, represented as top-10 lists (Kamishima et al., 2005). This model helps us predict choice probabilities with high accuracy, and our results demonstrate that the top- k Mallows model achieves significantly higher predictive accuracy than the Multinomial Logit model on this dataset

2 Preliminaries and Definitions

Let $N = [n] := \{1, 2, \dots, n\}$ represent a universe of n elements. A top- k list is a partial order on N structured as $i_1 \succ i_2 \succ \dots \succ i_k \succ \{i_{k+1}, \dots, i_n\}$, where the top- k elements are strictly ordered, while the remaining $n - k$ elements are incomparable to each other. The collection of all top- k lists over N is represented by $T_{k,N}$, where $T_{n,N} = S_N$ corresponds to the symmetric group on N .

For a top- k list τ and a position $l \in [k]$, $\tau(l)$ refers to the element ranked at position l , while $\bar{\tau}$ denotes the set of elements ranked below the top- k . For simplicity, we sometimes use τ to represent the top- k elements $\{\tau(1), \tau(2), \dots, \tau(k)\}$. Thus, $i \in \tau$ indicates that i is ranked among the top- k elements of τ , and set operations like $\subseteq$ and $\cap$ are applied accordingly. For $i, j \in N$, $i \succ_{\tau} j$ means i is ranked

¹We remark that some prior work considers scenarios where users select multiple items or a list of k items—referred to as a top- k choice. In contrast, our focus is solely on the selection of a single item, and we use the term *top- k* only as a parameter of the Mallows model.

above j in τ , i.e., $i \in \tau$ and either $j \in \bar{\tau}$ or $j \in \tau$ but ranked below i . Additionally, $i \parallel_{\tau} j$ indicates i and j are incomparable (both are in $\bar{\tau}$), while $i \perp_{\tau} j$ means they are comparable ($i \succ_{\tau} j$ or $j \succ_{\tau} i$).

In this paper, we utilize the widely recognized Kendall's Tau distance, a commonly used metric that quantifies the number of pairwise disagreements between two permutations (Fagin et al., 2003; Critchlow, 2012). This concept has been extended to top- k lists, where it no longer forms a true metric but retains useful mathematical properties. Given a parameter $p \geq 0$, the p -parametrized distance between $\tau, \tau' \in T_{k,N}$ is defined as

$$\mathcal{K}^p(\tau, \tau') = \sum_{i,j \in \tau \cup \tau' : i < j} \mathcal{K}_{i,j}^p(\tau, \tau'), \text{ where } \mathcal{K}_{i,j}^p(\tau, \tau') = \begin{cases} 1 & \text{if } (i \succ_{\tau'} j \ \& \ j \succ_{\tau} i) \text{ or vice-versa} \\ p & \text{if } (i \perp_{\tau'} j \ \& \ i \parallel_{\tau} j) \text{ or vice-versa} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Using this distance measure, Chierichetti et al. (2018a) define the *Mallows model for the top- k lists*. Given a center τ^* and a decay parameter β , the probability distribution $\mathcal{D}$ over top- k lists $T_{k,N}$ is defined as:

$$\mathbb{P}_{\mathcal{D}}[\tau \in T_{k,N}] \propto \exp(-\beta \mathcal{K}^p(\tau, \tau^*)) . \quad (\text{TopKMM})$$

To simplify our notation, we assume, without loss of generality, that the center τ^* is always the identity list $1 \succ 2 \succ \dots \succ k \succ \{k+1, \dots, n\}$. Therefore, we denote $\mathcal{K}^p(\tau, \tau^*)$ as $\mathcal{K}^p(\tau)$. For full rankings, where all elements are comparable, we simply use $\mathcal{K}(\tau)$.

A natural extension of this model arises when the elements have associated weights. *Generalized Mallows Model* (GMM) Fligner & Verducci (1986) considers this case for full rankings: Given a decay parameter β and non-negative weights $w_i \in \mathbb{R}_{\geq 0}$ for each $i \in N$, the probability distribution $\mathcal{D}$ over rankings is defined as:

$$\mathbb{P}_{\mathcal{D}}[\tau \in T_{n,N}] \propto \exp\left(-\beta \sum_{i,j:i < j} w_i \mathcal{K}_{i,j}(\tau)\right) \quad (\text{GMM})$$

where $\mathcal{K}_{i,j}(\tau)$ is 1 iff τ disagrees with τ^* , i.e., τ ranks j before i for $j > i$ and 0 otherwise, as defined in (1). In this model, each disagreement contributes the weight of the item ranked higher in τ^* . This formulation can be simplified by the use of inversion vectors as follows:

$$\mathbb{P}_{\mathcal{D}}[\tau \in T_{n,N}] \propto \exp\left(-\beta \sum_{i \in [k]} w_i I_i(\tau)\right) \text{ where } I_i(\tau) = \sum_{j:j > i} \mathcal{K}_{i,j}(\tau) = \sum_{j:j > i} \mathbb{1}(j \succ_{\tau} i)$$

where $\mathbb{1}$ is an indicator function that takes the value 1 when true and 0 otherwise.

The generalized mallows model has traditionally been defined only for full rankings, and we extend it to *Generalized Mallows Model for Top- k lists*. In this setting, each element $j \in \tau^*$ is assigned a non-negative weight, i.e., $w_i \in \mathbb{R}_{\geq 0}$ for $i \in [k]$, along with an additional weight $w_0 \in \mathbb{R}_{\geq 0}$ for any element in $\bar{\tau}^*$. We use $\mathbf{w} \in \mathbb{R}_{\geq 0}^{k+1}$ to represent this collection of weights, which uses the following extension for inversion vectors:

Definition 2.1 (Inversion Vectors of a Top- k list). *Given a top- k list $\tau \in T_{k,N}$, there are three components for inversion vectors: vectors $\mathbf{I}(\tau), \mathbf{P}(\tau) \in \mathbb{R}_{\geq 0}^k$ where for $i \in [k]$:*

$$I_i(\tau) = \sum_{j:j > i} \mathbb{1}(j \succ_{\tau} i), P_i(\tau) = \sum_{j:j > i} \mathbb{1}(i, j \in \bar{\tau})$$

and $Q(\tau) = \binom{k-\ell}{2}$ where $\ell = |\tau \cap \tau^*|$.

Note that this definition is an alternative way to count the disagreement between τ and τ^* where the disagreement is always assigned to the higher-ranked element by τ^* . When neither element is ranked higher ($i, j \notin \tau^*$), disagreements are assigned to Q . For example, for $\tau = (2, 1, 6, 5) \in T_{4,[8]}$, we have $\mathbf{I}(\tau) = [1, 0, 2, 2]$, $\mathbf{P}(\tau) = [0, 0, 1, 0]$, $Q(\tau) = 1$. $I_1(\tau) = 1$ since element 2 is ranked higher than 1 by τ , $I_3(\tau) = 2$ since elements 5 and 6 are ranked higher than 3 and the same argument applies to $I_4(\tau)$. $P_3(\tau) = 1$ since elements 3 and 4 are not comparable by τ (but they are ranked by τ^*) and this disagreement is assigned to element 3 as it is ranked higher than element 4 by τ^* . Q counts disagreements for elements not ranked by τ^* , i.e., elements 5 and 6.

Generalized Mallows Model for Top- k lists Given the center $\tau^* \in T_{k,N}$ and parameters $\beta \geq 0$, $\mathbf{w} \in \mathbb{R}_{\geq 0}^{k+1}$ and $p > 0$, the probability distribution $\mathcal{D}$ over $T_{k,N}$ is defined as:

$$\mathbb{P}_{\mathcal{D}}[\tau \in T_{k,N}] \propto \exp(-\beta \mathcal{K}^{p,\mathbf{w}}(\tau, \tau^*)) \quad (\text{TopKGMM})$$

where

$$\mathcal{K}^{p,\mathbf{w}}(\tau, \tau^*) := w_0 p Q(\tau) + \sum_{i \in [k]} w_i \cdot (I_i(\tau) + p P_i(\tau)).$$

Note that setting $\mathbf{w} = \mathbf{1}$, we obtain Equation (TopKMM), and $k = n$, recovers Equation (GMM).

3 Sampling from TopKGMM

We begin by investigating the challenge of efficiently sampling from TopKGMM. Although Chierichetti et al. (2018a)'s sampling algorithms for TopKMM can easily be extended to incorporate weights as in TopKGMM, here our concentration is to develop a sampling algorithm with similar functionality to the *repeated insertion model*, which was left open (Chierichetti et al., 2018a). The theory we develop here not only helps in the design of the sampling algorithm PRIM but also plays a crucial role in the development of DYPCHIP, our proposed algorithm for choice probabilities.

Theorem 3.1 (Sampling from TopKGMM). *For a given instance $\mathcal{D}$ of TopKGMM there exists an algorithm that efficiently samples a top- k list according to $\mathcal{D}$ in time complexity $O(k \cdot 2^\gamma + k \log n)$,² and space complexity $O(k \cdot 2^\gamma)$; where $\gamma = \min\{k, n - k\}$.*

Similar to the RIM method for permutations (Doignon et al., 2004), the core idea of our approach involves iteratively adding elements to a partially ordered sequence until exactly k elements are sampled. While this strategy results in a correct sampling scheme in the context of permutations, for top- k lists it is essential to partition the sample space based on several features before iterative insertions begin. Our careful definition of inversion vectors in Equation (1) plays a crucial role here, as it allows us to focus on the behavior of elements in $[k]$ and specifically those sampled by τ . This insights leads us to introduce *profiles* which represent the shared top- k elements τ and τ^* .

3.1 Profile-based TopKGMM Distribution

Let profile $S \subseteq [k]$ represent the subset of sample top- k elements for a given top- k list τ . Formally,

Definition 3.2 (top- k Profile). *Given a center $\tau^* \in T_{k,N}$ with corresponding TopKGMM distribution $\mathcal{D}$, we call a set $S \subseteq \tau^*$ a profile and we define $T_{\tau^*}(S)$ and its probability with respect to $\mathcal{D}$ as:*

$$T_{\tau^*}(S) = \{\tau \mid \tau \cap \tau^* = S\}, \quad \mathbb{P}_{\mathcal{D}}[S] = \sum_{\tau \in T_{\tau^*}(S)} \mathbb{P}_{\mathcal{D}}[\tau]. \quad (2)$$

When the center is clear from the context, we may simply use $T(S)$.

For any $\tau \in T(S)$, since $Q(\tau) = \binom{k-|S|}{2}$ only depends on S , we simply write $Q(S)$. The inversion vector P counts the number of lower-priority elements that were previously ranked strictly lower but are now incomparable. An element j has a positive $P_j(\tau)$ only if it is not ranked by τ , in which case its value is given by $P_j(\tau) = n - k - (k - \ell)$ as exactly $k - \ell$ elements from $\{k + 1, \dots, n\}$ are now ranked by τ . Since P strictly depends on S , we can use $P(S)$ instead of $P(\tau)$.

The inversion vector $\mathbf{I}$ is defined for elements in $[k]$, and for each element, is the number of lower-priority elements (w.r.t. the center) that are now ranked higher in τ . For any element j not ranked by τ , i.e., $j \in [k] \setminus S$, $I_j(\tau) = |\{j + 1, \dots, k\} \cap S|$. Thus part of vector $\mathbf{I}$, is entirely determined by S and independent of τ . Since $Q(S)$, $\mathbf{P}(S)$, and $\mathbf{I}_{j \in [k] \setminus S(\tau)}$ are independent of rankings among τ 's elements and depend only on S , we compute the probability of S given this information.

Lemma 3.3. *Given a TopKGMM distribution $\mathcal{D}$ with parameters β , p and $\mathbf{w}$, any profile $S = \{s_1, s_2, \dots, s_\ell\} \subseteq [k]$ ($s_1 < s_2 < \dots < s_\ell$) has probability $\mathbb{P}_{\mathcal{D}}[S]$ proportional to $\exp(-\beta f(S)) Z(S)$, where:*

$$f(S) = w_0 p Q(S) + \sum_{j \in [k] \setminus S} w_j (I_j(S) + p P_j(S)), \text{ and } Z(S) = \binom{n-k}{k-\ell} (k-\ell)! \prod_{j=1}^{\ell} \sum_{r=0}^{k-j} e^{-\beta w_{s_j} r}.$$

²Here $O(k2^\gamma)$ is the complexity of a pre-processing step, after that each sample can be generate at cost $O(k \log n)$.

and it can be sampled in $O(2^\gamma k)$; where $\gamma = \min\{k, n - k\}$.

3.2 Sampling Algorithm

Building on the results from the previous section and leveraging the concept of a *profile*, we now propose a method for sampling from the TopKGMM distribution (Algorithm 3). We first sample a profile S with probability $\mathbb{P}_{\mathcal{D}}[S]$ (according to Lemma 3.3), and then we sample $\tau \in T(S)$ by inserting the elements $j \in S$ based on their contribution in inversion vectors Equation (TopKGMM)). The later step can be viewed as generalizing the Repeated Insertion Method (RIM). We refer to our method as PROFILE-BASED-REPEATED-INSERTION-METHOD (PRIM).

In PRIM, we generate a top- k list proportional to its sampling probability in $\mathcal{D}$ conditioned on the fact that the common elements with the center top- k ranking is S . Specifically, let $\ell = |S|$. We begin with an empty array A and then sample $k - \ell$ elements from $[n] \setminus [k]$ uniformly at random. Then we sequentially, insert the elements in S in increasing order of their priority. When processing element s , we insert s in the current array A at position $j \in \{0, 1, \dots, |A|\}$ with probability proportional to

$$Pr(\text{inserting } s \text{ at position } j) \propto \exp(-\beta w_s \cdot j) \quad (3)$$

as j is the number of inversion associated with element s when it is inserted at position j . Note that higher priority elements do not contribute to inversion of s so when their position when they are inserted later is not important for s . See Appendix A.1 for pseudocodes and proofs.

4 Choice modeling and probabilities

In this section, we focus on the problems related to *choice*. In particular we focus on the analysis of problems which help us predict the (*top*) *choice* of a customer from a set of alternatives, a.k.a an *assortment* using TopKGMM. After presenting definitions we design an algorithm which efficiently calculates choice probabilities having the distributions parameters. In Section 5 we study the opposite problem in this context, which is learning the center of a TopKGMM distribution by observing its choice data.

Given a set of products $[n]$, an *assortment* is any subset $\mathcal{A}$ of $[n]$. When $\mathcal{A}$ is offered to a customer, she may *choose* any of its elements or the “no purchase” option denoted by $\emptyset$. We use $N = [n] \cup \{\emptyset\}$ to denote all purchase options and correspondingly, we define $T_{k,N}$ to denote all top- k lists over N . We assume that the preferences of customers are derived from TopKGMM, and if we have multiple customer types, we use a mixture of several TopKGMM’s where each may have different parameters.

Given a top- k list $\tau \in T_{k,N}$, we define the *choice* function $\mathcal{C}_\tau : 2^{[n]} \rightarrow N$ as follows: for any assortment $\mathcal{A} \subset [n]$, $\mathcal{C}_\tau(\mathcal{A}) = i$, iff i is the highest ranked element in $\mathcal{A} \cup \{\emptyset\}$ with respect to τ . If all elements of $\mathcal{A} \cup \{\emptyset\}$ are incomparable w.r.t. τ , then i is taken uniformly from $\mathcal{A} \cup \{\emptyset\}$.

We focus on applying the TopKGMM model to represent customer preferences. Specifically, we assume that τ is sampled from a distribution $\mathcal{D}$, where $\mathcal{D}$ is a TopKGMM distribution characterizing a single customer type. More generally, $\mathcal{D}$ can represent several customer types and we may use a mixture distribution. Since choice probabilities of a mixture distribution can simply be obtained as a linear combination of its singleton components, here we focus on singleton distributions.

4.1 Calculation of Choice Probabilities: DYPCHIP

Let $\mathcal{D}$ be a TopKGMM distribution on $T_{k,N}$. We let $\mathcal{C}_{\mathcal{D}}$ be a function mapping any assortment $\mathcal{A} \subseteq [n]$ and an option $i \in \mathcal{A} \cup \{\emptyset\}$ to the probability that $\mathcal{C}_\tau(\mathcal{A}) = i$ where τ is sampled from $\mathcal{D}$. Formally, $\mathcal{C}_{\mathcal{D}} : N \times 2^{[n]} \rightarrow [0, 1]$ is defined as follows:

$$\mathcal{C}_{\mathcal{D}}(i, \mathcal{A}) = \sum_{\tau \in T_{k,N}} \mathbb{P}_{\mathcal{D}}[\tau] \cdot \mathbb{1}(\mathcal{C}_\tau(\mathcal{A}) = i) . \quad (4)$$

We now introduce DYPCHIP which calculates *choice probabilities* as defined in Equation (4), and its correctness and runtime complexity is stated in the following theorem:

Theorem 4.1 (Calculation of choice probabilities). *Given an assortment $\mathcal{A} \subseteq [n]$ and a TopKGMM instance $\mathcal{D}$, DYPCHIP calculates $\mathcal{C}_{\mathcal{D}}(j, \mathcal{A})$ for all $j \in \mathcal{A} \cup \{\emptyset\}$ in $O\left(2^{\min\{k, n-k\}} k^3 |\mathcal{A}|^2\right)$.*

The main idea of this algorithm is to find these probabilities by conditioning on a given profile. Our dynamic programming tables are defined based on the order in which items are considered in PRIM.

Overview of DYPCHIP Consider profile $S \subseteq [k]$ with $\ell = |S|$. Let $\mathcal{A}^\varnothing = \mathcal{A} \cup \{\varnothing\}$. For a top- k list $\tau \in T(S)$, item a from $\mathcal{A}$ is picked if (i) τ does not include any item from $\mathcal{A}^\varnothing$ and so a is picked randomly from $\mathcal{A}^\varnothing \subseteq \bar{\tau}$, or (ii) a is top-ranked in $\mathcal{A}^\varnothing$ w.r.t τ . We handle the two cases separately:

1. Let $\bar{\pi}_S(a)$ denote the probability associated to case (i). Note that $\tau \cap \mathcal{A}^\varnothing = \emptyset$ implies that τ is from a profile S with $S \cap \mathcal{A} = \emptyset$. Thus for any $a \in \mathcal{A}^\varnothing$:

$$\bar{\pi}_S(a) = \mathbf{P}(\tau \cap \mathcal{A}^\varnothing = \emptyset) \cdot \frac{1}{|\mathcal{A}| + 1} = \mathbb{1}(\mathcal{A}^\varnothing \cap S = \emptyset) \cdot \frac{\binom{n-k-(|\mathcal{A}|+1)}{k-\ell}}{\binom{n-k}{k-\ell}} \cdot \frac{1}{|\mathcal{A}| + 1}.$$

2. Let $\bar{\mathcal{A}} \doteq \mathcal{A}^\varnothing \cap \bar{\tau}$. The elements in $\mathcal{A}^\varnothing$ who have a non-zero probability of being top ranked at some iteration of DP table are only in $\bar{\mathcal{A}} \cup S$. We calculate these probabilities by conditioning on two other parameters: (1) where in PRIM algorithm they have a chance of being sampled, and (2) the position in which they are positioned in the top- k list when they are the winner, i.e, ranked highest among $\mathcal{A}^\varnothing$. We use a three dimensional dynamic programming table π_S as follows: Let $\{a_1, a_2, \dots, a_r, s_\ell, s_{\ell-1}, \dots, s_1\}$ be an ordering of the elements in $\bar{\mathcal{A}} \cup S$ where the first segment is an arbitrary ordering of $\bar{\mathcal{A}}$ and the second is the ordering of S used in PRIM.

For $q = 0$, we let $\pi_S(i, j, q)$ be the probability that any element $a_i \in \{a_1, a_2, \dots, a_r\}$ is ranked j th and higher than all other elements of $\bar{\mathcal{A}}$ (see Equation (6)). Then, by iterating over $q = 1, 2, \dots, \ell$, we consider the q th iteration of the for loop in PRIM, and for any position $1 \leq j \leq q$, we define $\pi_S(i, j, q)$ be the probability that after completion of the q th iteration of the for loop in PRIM, a_i is the highest element in $\mathcal{A}^\varnothing$ (the winner) which has so far been sampled, and a_i is so far ranked j -th.

We update these probabilities by conditioning whether the newly inserted element, i.e., $s_{\ell+1-q}$ is added before the prior winner, or after it. Note that, since we the profile S is fixed, the probability of inserting a new element in a particular location may be obtained from Equation (3). For details of the recursive definition of $\pi_S(i, j, q)$ please see Appendix A.2. After the dynamic programming table is filled, we may calculate the choice probability of each $a_i \in \mathcal{A}^\varnothing$ conditioned on profile S with $|S| = \ell$ by:

$$\pi_S(a_i) = \sum_{j=1}^{\ell} \pi_S(i, j, \ell).$$

In each cell of the table, we need to look at $O(k)$ other cells which involves at most $O(k|\mathcal{A}|)$ operations so each table entry can be calculated in $O(k|\mathcal{A}|)$ time. Finally, we may calculate the

$$\forall a \in \mathcal{A}, \mathcal{C}_D(a, \mathcal{A}) = \sum_{S \subseteq [k]} \mathbb{P}_D[S] (\pi_S(a) + \bar{\pi}_S(a)). \quad (5)$$

Runtime of DYPCHIP The size of the dynamic programming table is $|\mathcal{A}|k^2$, and calculation of each element needs $k|\mathcal{A}| + n$ operations. In Equation (5) we have a sum over all profiles; which is bounded by $O(2^{\min\{k, n-k\}})$. Thus, in total all choice probabilities may be calculated in time $\Theta(2^\gamma k^3 |\mathcal{A}|^2 + 2^\gamma k^2 |\mathcal{A}|n)$; $\gamma = \min\{k, n-k\}$.

5 Learning the center from choice data

In this section, we focus on the problem of learning the center of a TopKGMM distribution from *choice data*. This task is more challenging than learning from complete top- k samples, as each data point provides limited information and the choice data depends on the specific assortments presented.

Consider a TopKGMM distribution $\mathcal{D}$ on $T_{k,N}$, the goal is to construct the center of $\mathcal{D}$, namely τ^* by observing choice data. The choice data consists of pairs $(\mathcal{A}_t, c_t)$, where $\mathcal{A}_t$ represents the assortment offered at round t , and c_t is the item selected by the customer from $\mathcal{A}_t$. Formally, we let $\mathbf{D} = \langle (c_1, \mathcal{A}_1), (c_2, \mathcal{A}_2), \dots, (c_T, \mathcal{A}_T) \rangle$; where for $t = 1, 2, \dots, T$ we have $c_t = \mathcal{C}_\tau(\mathcal{A}_t)$, $\tau \sim \mathcal{D}$. While in traditional learning algorithms $\mathbf{D}$ is given as input. Here, we focus of *active* learning and collect choice data by presenting assortments to the customers and recording observed choices.

Our active learning algorithm for the estimation of the center is BUCCHOI. It takes as input the set of products N and assortment size ℓ and offers a sequence of assortments $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_T$ to the

Algorithm 1 FINDTOP

```

1: Input: Assortment  $\mathcal{A}$ , a sequence
   ( $c_1, c_2, \dots, c_m$ ) where  $c_i = \mathcal{C}_\tau(\mathcal{A})$ 
   for  $\tau$  sampled from  $\mathcal{D}$ .
2: Output:  $\mathcal{C}_{\tau^*}(\mathcal{A})$  if  $\tau^* \cap \mathcal{A}^\varnothing \neq \emptyset$ .
3:  $X \leftarrow \mathbf{0}_{\mathcal{A}^\varnothing \times \mathcal{A}^\varnothing}$ 
4: for  $i = 1 : m$  do
5:   for  $a \in \mathcal{A}^\varnothing \setminus \{c_i\}$  do
6:      $X_{c_i a} = X_{c_i a} + 1$ 
7:      $X_{a c_i} = X_{a c_i} - 1$ 
8:   end for
9: end for
10:  $Y \leftarrow X/m$ 
11: if  $\exists a : Y_{aa'} > \frac{1+|\mathcal{A}|}{2} \forall a' \in \mathcal{A}^\varnothing \setminus \{a\}$ 
   then
12:   Return  $a$ 
13: else
14:   Return None
15: end if

```

Algorithm 2 BUCCHOI

```

1: Input:  $N$ : set of products,  $r$ : assortment size,  $m$ : num-
   ber of samples, choice oracle  $\mathcal{C}_\tau$ ;  $\tau \sim \text{TopKGMM } \mathcal{D}$ .
2: Output:  $k$  size of center,  $\tau^*$  center of  $\mathcal{D}$ 
3:  $T = \emptyset, B = \emptyset, U = N$ 
4: repeat
5:    $\mathcal{A} =$  assortment of size  $r$  from  $U^a$ 
6:    $S = \emptyset$  # collect choice data  $S$  by showing  $\mathcal{A}$  repeatedly
7:   for  $j = 1 : m$  do
8:      $S = S \cup \mathcal{C}_\tau(\mathcal{A})$ 
9:   end for
10:   $a = \text{FINDTOP}(\mathcal{A}, S)$ 
11:  if  $a \neq \text{None}$  then
12:     $T = T \cup \{a\}, U = U \setminus \{a\}$ 
13:  else
14:     $B = B \cup \mathcal{A}, U = U \setminus \mathcal{A}$ 
15:  end if
16: until  $U = \emptyset$ 
17:  $k = |U|$ 
18:  $\tau^* = \text{SORTCNTR}(U, r, m)$  # (Algorithm 6)
19: Return  $k, \tau^*$ 

```

^aIf $|U| < r$ use some elements from B

customers, recoding corresponding choices c_t for $t = 1, 2, \dots, T$. A Pseudocode of BUCCHOI is presented in Algorithm 2. Assume TopKGMM distribution $\mathcal{D}$ with center τ^* , and parameters $\beta, p, \bar{w}$. Let $w_{\min} \doteq \min_{i \in k} w_i$, and $n = |N|$. The following theorem shows the sample complexity of BUCCHOI:

Theorem 5.1. Assume that $\beta \geq \log 3/w_{\min}$ and $\varnothing \notin \tau^*$. By only receiving N and r as input and being able to collect choice samples $\mathbf{D}$ by selecting assortments, with probability at least $1 - o(1)$, we are able to learn τ^* and k from $\mathbf{D}$ using only $\Theta(r^2 \log n)$ choice samples.

The main building block of BUCCHOI is a procedure FINDTOP which given an assortment $\mathcal{A}$ and a set of choice samples $\mathbf{D}$, outputs $i \in \mathcal{A}^\varnothing$ such that i has the highest rank w.r.t. τ^* among elements of $\mathcal{A}^\varnothing$. If all elements of $\mathcal{A}^\varnothing$ are incomparable in τ^* , FINDTOP returns None. We note that FINDTOP is *not* an active learning algorithm and receives $\mathbf{D}$ as input. We also assume that the choice data were collected by presenting a single arbitrary assortment $\mathcal{A}$.

FINDTOP A pseudocode for FINDTOP is presented in Algorithm 1. The main idea is to maintain for each $i, j \in \mathcal{A}^\varnothing$, a variable X_{ij} . For any choice sample c_t we increment X_{ij} if i is chosen over j , i.e., $i = c_t$ and $j \in \mathcal{A} \setminus \{c_t\}$, and we decrement it otherwise. Taking $Y_{ij} = X_{ij}/m$ for all $i, j \in \mathcal{A}^\varnothing$, it is not difficult to see that:

$$\mathbb{E}[Y_{ij}] = \mathcal{C}_\tau(i, \mathcal{A}) - \mathcal{C}_\tau(j, \mathcal{A}), \quad \tau \sim \mathcal{D}.$$

Based on this observation and by calculating a lower bound on $\mathcal{C}_\tau(i, \mathcal{A}) - \mathcal{C}_\tau(j, \mathcal{A})$ when i is $\mathcal{A}^\varnothing$'s top element w.r.t. τ^* and $j \in \mathcal{A}^\varnothing \setminus \{i\}$ we are able to show that the top element will be found by FINDTOP if the discrepancy parameter β is large enough and we have enough samples. Formally we show the following lemma whose proof is presented in full details in Appendix A.3:

Lemma 5.2. Assume that $\beta \geq \log(3)/w_{\min}$ and let $r = |\mathcal{A}|$ and $\zeta \geq 1$ arbitrary constant. If $\mathcal{A}$ appears at least $\Theta(\zeta(r+1)^2 \log n)$ times among the displayed assortments, with probability at least $1 - o(n^{-\zeta})$ we have: if $\mathcal{A}^\varnothing \cap \tau^* \neq \emptyset$, FINDTOP will return $i \in \mathcal{A}^\varnothing$ such that $i \succ_{\tau^*} j$ for any $j \in \mathcal{A}^\varnothing \setminus \{i\}$, otherwise, it returns None, and we can conclude that $\mathcal{A}^\varnothing \cap \tau^* = \emptyset$.

BUCCHOI BUCCHOI first identifies all elements in N which are ranked above $\varnothing$ and are in τ^* . To this end, we maintain three sets: $T \subseteq \tau^*$ and $B \subseteq \bar{\tau}^*$, U unknown whether they are in τ^* or $\bar{\tau}^*$. Assortments of size r are selected repeatedly and after calling FINDTOP we either find the top

element in the assortment – which has to be in τ^* or we find out that none of the elements in the assortment are in τ^* . Note that the number of times that Repeat loop iterates is bounded by $k + n/r$. Finally we call Algorithm 6 to find the rank of items in τ^* . Theorem 5.1 will be concluded from Lemma 5.2 and using a union bound on all FINDTOP calls. We remark that if $\emptyset \in \tau^*$, BUCCHOI will be able to return the top prefix of τ^* constituting of the elements ranked above $\emptyset$ (see Appendix A.3).

6 Experiments

In this section, we present our experimental analysis, designed to achieve two main objectives: (1) to compare the predictive power of the top- k Mallows model (TopKGMM) with that of the multinomial logit model (MNL), and (2) to evaluate the accuracy and computational complexity of our methods, namely PRIM sampling algorithm, the DYPCHIP choice probability computation, and the two learning algorithms, FINDTOP and BUCCHOI. The code and log files are available publicly³. Results are generated by running the code on a MacBook Pro M1 Max, 32GM RAM.

Predictive power of top- k MM compared to MNL: experiments on real-world data We used “Sushi Preference Data Set” (Kamishima et al., 2005) which contains preference of customers over a set of 100 different sushi types⁴. The data-set includes 5K preferences in the form of top-10 lists.

Set-Up. We begin by randomly splitting the 5K top-10 preference data into a training set (80%) and a test set (20%). We apply BUCCHOI using assortments of size one or two (Algorithm 7) to the training set using various values of p and β to learn the center of the distribution. With the learned parameters, we use DYPCHIP to compute the corresponding choice probabilities.

For evaluation, we use empirical choice probabilities on the test set by repeatedly sampling random assortments and recording corresponding choices. These empirical estimates are then compared to the predictions from DYPCHIP to assess out-of-sample accuracy; the errors are reported in Table 1. Based on these results, we identify the values of p and β that yield the lowest test error. Figure 1a shows the prediction error of the TopKGMM compared to MNL after tuning.

Considering multiple customer types, we cluster the training data (into 2–5 groups) using the Kendall’s Tau distance $\mathcal{K}^p$, varying p . Clusters with positive silhouette scores are retained, and choice probabilities are computed within each cluster using both TopKGMM and MNL. Final predictions are weighted averages based on cluster sizes. Figure 1b shows the results for the two-clusters.

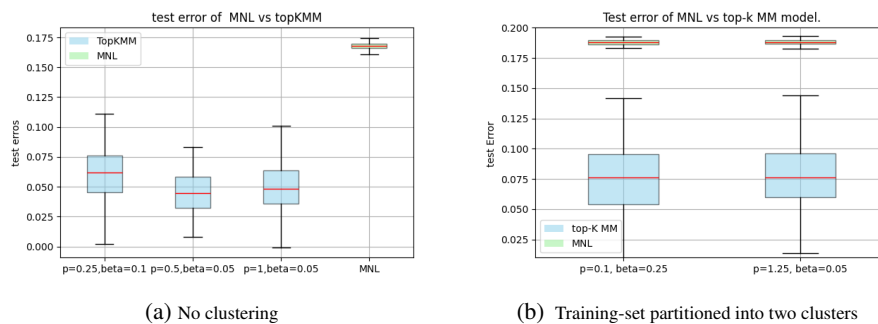


Figure 1: Test error of the top- k Mallows model compared to MNL. Parameter β and p have been selected to derive highest accuracy. Tables 1 and 2 show the test error for all choices of p and β .

Experimental Findings. Our findings show high accuracy of out-of-sample choice probability prediction of TopKGMM compared to the accuracy obtained from MNL. These results are consistent with findings of Désir et al. (2021) who observe the same for classic Mallows model on permutations.

Accuracy and complexity of algorithms: experiments on synthetic Data We use synthetic data to evaluate the accuracy and complexity of our algorithms, as it provides access to ground-truth choice probabilities and distribution centers—information unavailable in real-world datasets. This enables controlled analysis of sample complexity trade-offs with respect to key parameters: n (number of products), k (size of top- k), r (assortment size), β (decay parameter) and p (Kendall’s Tau parameter).

³Link to the code <https://github.com/ShahrzadGit/topkmallows-choices>

⁴Link to of Sushi Preference Data Set <https://www.kamishima.net/sushi/>.

Accuracy and time complexity of PRIM and DYPCHIP. We evaluate algorithm accuracy by generating samples from a TopKGMM using PRIM and comparing empirical choice frequencies over random assortments to the probabilities predicted by DYPCHIP. This is repeated across 20 assortments, with the mean and standard deviation of results shown in Figure 2a. The run-times of DYPCHIP and PRIM are reported in Appendix B.2.

Accuracy and sample complexity of learning algorithms of BUCCHOI and FINDTOP. We evaluate our two learning algorithms by generating m samples from TopKGMM distribution using PRIM and running FINDTOP and BUCCHOI to learn the top element or distribution center. We assess the convergence of these methods by comparing the learned and true values. When learning the center in BUCCHOI, we use the Kendall's Tau distance $\mathcal{K}^p$ of learned and true center. In FINDTOP we check whether the learned top element is the same as the true top element and directly calculate accuracy based on the frequency of matching values. Each experiments is repeated 10 times across a range of model parameters and average and standard deviation are obtained; see Figure 2, and Appendix B.3.

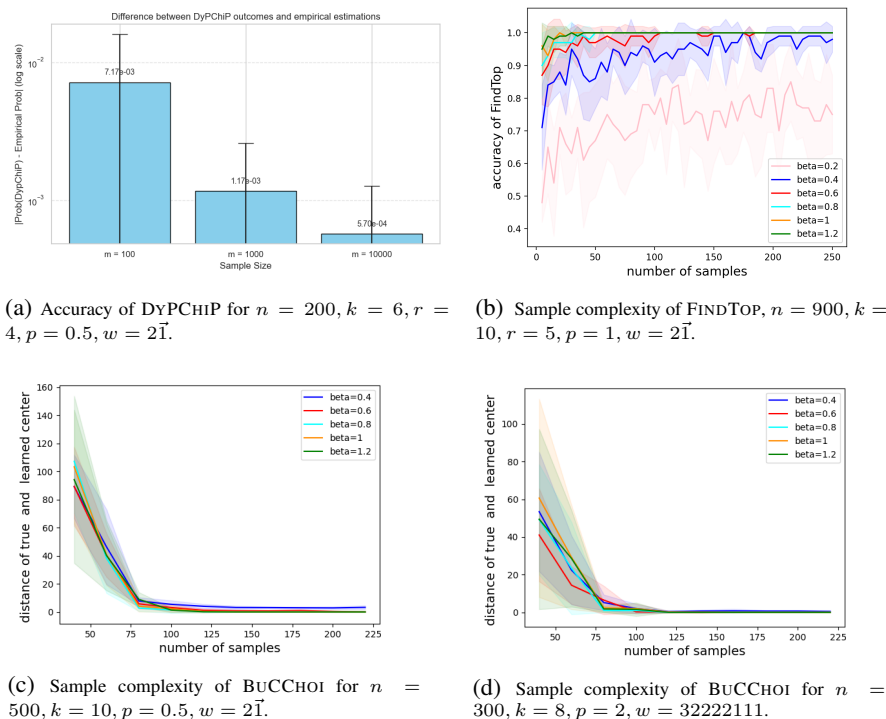


Figure 2: performance of algorithms DYPCHIP, FINDTOP and BUCCHOI on synthetic data.

Experimental Findings. Our experiments in this section support the theoretical results on the accuracy and complexity of our algorithms. Notably, we observe that our methods achieve high accuracy with a relatively small number of samples—often logarithmic in the number of items n .

For DYPCHIP, we find that time complexity increases rapidly with k , as expected due to its exponential dependence on this parameter (See Figure 3). In contrast, the runtime shows minimal sensitivity to the number of products n . For PRIM, the exponential dependence on k is less restrictive since it primarily affects the preprocessing step; once this step is completed, generating a large number of samples remains efficient with low amortized cost (See Table 3).

In the learning algorithms BUCCHOI and FINDTOP, we observe that the sample complexity increases as β decreases. This is expected, as smaller values of β cause the distribution to approach uniformity, reducing the concentration of samples around the center and making learning more challenging.

7 Conclusion

In conclusion, the generalized Mallows model for top- k lists provides a more realistic framework for understanding user preferences, particularly when users care only about a limited set of options. Our work centers on applying TopKGMM to choice modeling and developing several key algorithms. An important open direction is learning the model parameters for mixture versions of these models.

8 Acknowledgment

We are thankful to Anonymous NeurIPS reviewers for helping us strengthen the presentation of our paper. Shahrzad Haddadan is supported by Rutgers Business School's Dean's Research Seed Fund.

References

- Akbari, S. and Escobedo, A. R. Top-k list aggregation: Mathematical formulations and polyhedral comparisons. In *International Symposium on Combinatorial Optimization*, pp. 51–63. Springer, 2022.
- Awadelkarim, A. and Ugander, J. Statistical models of top-k partial orders. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 39–48, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3672014. URL <https://doi.org/10.1145/3637528.3672014>.
- Awasthi, P., Blum, A., Sheffet, O., and Vijayaraghavan, A. Learning mixtures of ranking models. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2609–2617, Cambridge, MA, USA, 2014. MIT Press.
- Blanchet, J., Gallego, G., and Goyal, V. A markov chain approximation to choice modeling. *Operations Research*, 64(4):886–905, 2016.
- Boehmer, N., Celis, L. E., Huang, L., Mehrotra, A., and Vishnoi, N. K. Subset selection based on multiple rankings in the presence of bias: Effectiveness of fairness constraints for multiwinner voting score functions. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2641–2688. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/boehmer23a.html>.
- Bradley, R. A. and Terry, M. E. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.
- Braverman, M. and Mossel, E. Noisy sorting without resampling. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pp. 268–276, USA, 2008. Society for Industrial and Applied Mathematics.
- Chierichetti, F., Dasgupta, A., Kumar, R., and Lattanzi, S. On learning mixture models for permutations. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pp. 85–92, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450333337. doi: 10.1145/2688073.2688111. URL <https://doi.org/10.1145/2688073.2688111>.
- Chierichetti, F., Dasgupta, A., Haddadan, S., Kumar, R., and Lattanzi, S. Mallows models for top-k lists. *Advances in Neural Information Processing Systems*, 31, 2018a.
- Chierichetti, F., Kumar, R., and Tomkins, A. Learning a mixture of two multinomial logits. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 961–969. PMLR, 10–15 Jul 2018b. URL <https://proceedings.mlr.press/v80/chierichetti18a.html>.
- Collas, F. and Irurozki, E. Concentric mixtures of mallows models for top- k rankings: sampling and identifiability. In *International Conference on Machine Learning*, pp. 2079–2088. PMLR, 2021.
- Critchlow, D. E. *Metric methods for analyzing partially ranked data*, volume 34. Springer Science & Business Media, 2012.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22, 1977.
- Désir, A., Goyal, V., Jagabathula, S., and Segev, D. Mallows-smoothed distribution over rankings approach for modeling choice. *Operations Research*, 69(4):1206–1227, 2021.
- Désir, A., Goyal, V., Jiang, B., Xie, T., and Zhang, J. Robust assortment optimization under the markov chain choice model. *Operations Research*, 2023.

- Doignon, J.-P., Pekeč, A., and Regenwetter, M. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika*, 69(1):33–54, 2004.
- Fagin, R., Kumar, R., and Sivakumar, D. Comparing top k lists. *SIAM Journal on discrete mathematics*, 17(1):134–160, 2003.
- Feng, Y. and Tang, Y. On a mallows-type model for (ranked) choices. *Advances in Neural Information Processing Systems*, 35:3052–3065, 2022.
- Fligner, M. A. and Verducci, J. S. Distance based ranking models. *Journal of the Royal Statistical Society*, 1986. URL <http://www.jstor.org/stable/2345433>.
- Fotakis, D., Kalavasis, A., and Stavropoulos, K. Aggregating incomplete and noisy rankings. In *International Conference on Artificial Intelligence and Statistics*, pp. 2278–2286. PMLR, 2021.
- Goibert, M., Calauzènes, C., Irurozki, E., and Cléménçon, S. Robust consensus in ranking data analysis: Definitions, properties and computational issues. In *International Conference on Machine Learning*, pp. 11584–11597. PMLR, 2023.
- Kamishima, T., Kazawa, H., and Akaho, S. Supervised ordering — an empirical survey. In *Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM '05*, pp. 673–676, USA, 2005. IEEE Computer Society. ISBN 0769522785. doi: 10.1109/ICDM.2005.138. URL <https://doi.org/10.1109/ICDM.2005.138>.
- Lebanon, G. and Mao, Y. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9(79):2401–2429, 2008. URL <http://jmlr.org/papers/v9/lebanon08a.html>.
- Liu, A. and Moitra, A. Efficiently learning mixtures of mallows models. *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 627–638, 2018a.
- Liu, A. and Moitra, A. Efficiently learning mixtures of mallows models. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 627–638. IEEE, 2018b.
- Lu, T. and Boutilier, C. Learning mallows models with pairwise preferences. In *Proceedings of the 28th international conference on machine learning (icml-11)*, pp. 145–152, 2011.
- Lu, T. and Boutilier, C. Effective sampling and learning for mallows models with pairwise-preference data. *J. Mach. Learn. Res.*, 15(1):3783–3829, 2014.
- Ma, J., Zhang, X., and Mei, Q. Fast learning of mnl model from general partial rankings with application to network formation modeling. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM '22*, pp. 715–725, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391320. doi: 10.1145/3488560.3498506. URL <https://doi.org/10.1145/3488560.3498506>.
- Mallows, C. L. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.
- McFadden, D. and Train, K. Mixed mnl models for discrete response. *Journal of applied Econometrics*, 15(5):447–470, 2000.
- Oh, S. and Shah, D. Learning mixed multinomial logit model from ordinal data. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/b33336b853dc43b9f4a96cedf6bd4b30-Paper.pdf.
- Qian, Z. and Philip, L. Weighted distance-based models for ranking data using the r package rankdist. *Journal of Statistical Software*, 90:1–31, 2019.
- Rieger, A. and Segev, D. Quasi-polynomial time approximation schemes for assortment optimization under mallows-based rankings. *Math. Program.*, 208(1–2):111–171, December 2023. ISSN 0025-5610. doi: 10.1007/s10107-023-02033-4. URL <https://doi.org/10.1007/s10107-023-02033-4>.

- Seshadri, A., Ragain, S., and Ugander, J. Learning rich rankings. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9435–9446. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6affee954d76859baa2800e1c49e2c5d-Paper.pdf.
- Susan, F., Golrezaei, N., Emamjomeh-Zadeh, E., and Kempe, D. Active learning for non-parametric choice models. *arXiv preprint arXiv:2208.03346*, 2022.
- Tang, W. Mallows ranking models: maximum likelihood estimate and regeneration. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6125–6134. PMLR, 09–15 Jun 2019. URL <https://proceedings.mlr.press/v97/tang19a.html>.
- Vitelli, V., Sørensen, Ø., Crispino, M., Frigessi Di Rattalma, A., and Arjas, E. Probabilistic preference learning with the mallows rank model. *Journal of Machine Learning Research*, 18(158):1–49, 2018.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: All claims in the introduction and abstract are later clearly stated in the main body of the paper as theorems with a clear statement of all theoretical assumptions.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: The runtime of some algorithms are large and this is explained in the experiments sections.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: The assumptions are clearly stated and all the proofs are presented in the appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: The details of set-up, evaluation metrics and the data-set are explained in detail in the appendix of the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: We have used publicly available datasets and cited them in our paper. The code is attached to this submission.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: All hyperparameter settings are explained in detail in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[No\]](#)

Justification: We have ran all experiments 5 times and average results are reported but we don't have statistical significance analysis. We don't think it is applicable to our problem.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: They are reported at the beginning of the experiments section.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: It does.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: We are not aware of any negative societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We believe that our work does not have such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: We have only used publicly available datasets which are properly cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: We don't release new datasets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: human subjects

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our paper does not involve human subjects

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Missing proofs and details

A.1 Missing details from Section 3.2

In this section, we prove Theorem 3.1. To this end, we first establish the correctness of Lemma 3.3 that proves profile S can be sampled according to $\mathbb{P}_{\mathcal{D}}[S]$.

Lemma (Restatement of Lemma 3.3). *Given a TopKGMM distribution $\mathcal{D}$ with parameters β , p and w , any profile $S = \{s_1, s_2, \dots, s_\ell\} \subseteq [k]$ ($s_1 < s_2 < \dots < s_\ell$) has probability $\mathbb{P}_{\mathcal{D}}[S]$ proportional to $\exp(-\beta f(S)) Z(S)$, where:*

$$f(S) := w_0 p Q(S) + \sum_{j \in [k] \setminus S} w_j (I_j(S) + p P_j(S))$$

$$Z(S) := \binom{n-k}{k-\ell} (k-\ell)! \prod_{j=1}^{\ell} \sum_{r=0}^{k-j} e^{-\beta w_{s_j} r}$$

and it can be sampled in $O(2^k k)$.

Proof. Let $\mathcal{M}$ be the normalizing factor in Equation (TopKGMM), so $\mathcal{M} = \sum_{\tau \in T_{k,N}} \mathbb{P}_{\mathcal{D}}[\tau]$. Then

$$\begin{aligned} \mathbb{P}_{\mathcal{D}}[S] &= \frac{1}{\mathcal{M}} \sum_{\tau \in T(S)} \mathbb{P}_{\mathcal{D}}[\tau] && \text{(Equation (2))} \\ &= \frac{1}{\mathcal{M}} \sum_{\tau \in T(S)} \exp \left(-\beta (w_0 p Q(\tau) + \sum_{j \in [k]} w_j \cdot (I_j(\tau) + p P_j(\tau))) \right) && \text{(Equation (TopKGMM))} \\ &= \frac{1}{\mathcal{M}} \sum_{\tau \in T(S)} \exp \left(-\beta (w_0 p Q(S) + \sum_{i \in [k] \setminus S} w_i \cdot (I_i(S) + p P_i(S)) + \sum_{j \in S} w_j \cdot I_j(\tau)) \right) \\ &= \frac{1}{\mathcal{M}} \sum_{\tau \in T(S)} \exp \left(-\beta (w_0 p Q(S) + \sum_{i \in [k] \setminus S} w_i \cdot (I_i(S) + p P_i(S))) \right) \cdot \exp \left(-\beta \sum_{j \in S} w_j \cdot I_j(\tau) \right) \\ &= \frac{\exp(-\beta f(S))}{\mathcal{M}} \sum_{\tau \in T(S)} \exp \left(-\beta \sum_{j \in S} w_j \cdot I_j(\tau) \right) \end{aligned}$$

The remaining sum relates to the inversions of elements in S . Based on Definition 2.1, an element $s_j \in S \subseteq [k]$ makes an inversion with another element s iff $s_j \succ_{\tau^*} s$ and $s \succ_{\tau} s_j$. Note that since $\tau \in T_{\tau^*}(S)$, the possibilities for such s are $\{s_{j+1}, \dots, s_\ell\}$ and the elements of $[n] \setminus [k]$ which are now in τ , i.e., $\tau \cap \{k+1, \dots, n\}$. Note that $\tau \cap \{k+1, \dots, n\} = k-\ell$. Thus, the inversion of j can be any number between 0 to $(\ell-j) + (k-\ell)$ which is any number from 0 to $k-j$. Furthermore, for any valid selection of these values for elements $j \in S$ w.r.t. defined ranges, then the position of these elements in τ are uniquely determined. This can be achieved by starting with a sequence of $k-\ell$ stars which would correspond to any selection of elements from $\{k+1, \dots, n\}$, and then inserting element $s \in (s_\ell, s_{\ell-1}, \dots, s_1)$ iteratively based on their value of I_s . This shows the 1:1 correspondence between values of I_{j_s} and how elements of S are positioned in τ . Since there are $\binom{n-k}{k-\ell} (k-\ell)!$ possible cases for the arrangement of the remaining elements, we get that

$$\sum_{\tau \in T(S)} \exp \left(-\beta \sum_{j \in S} w_j \cdot I_j(\tau) \right) = \binom{n-k}{k-\ell} (k-\ell)! \sum_{\substack{\text{valid choice of } I_s \\ \text{for all } s \in S}} \exp \left(-\beta \sum_{j \in S} w_j \cdot I_j \right) = Z(S).$$

where the last equality follows from the fact that depending on the value of I_{s_j} , any of the terms in the summation $1 + e^{-w_{s_j} \beta} + \dots + e^{-(k-j)w_{s_j} \beta}$ appears once. Taking product over the terms selected from these sums would exactly correspond to a valid selection of I_s for all $s \in S$. Since $\mathcal{M} =$

$\sum_{\tau \in T_{k,N}} \mathbb{P}_{\mathcal{D}}[\tau]$ and each $\mathbb{P}_{\mathcal{D}}[\tau]$ contributes to the $\mathbb{P}_{\mathcal{D}}[S]$, the defined values form a probability distribution over different subsets of $[k]$ where $\mathbb{P}_{\mathcal{D}}[S]$ is proportional to $\exp(-\beta f(S)) Z(S)$.

It remains to show that this value can be computed in $O(2^\gamma k)$; $\gamma = \min\{k, n - k\}$. We argue that for each S , $Z(S)$ and $f(S)$ can be calculated in $O(k)$, the total number of profiles is bounded by 2^γ , hence all the computation can be done in $O(2^\gamma k)$. For $Z(S)$, the coefficient before sum has at most $2k$ terms and there are at most k terms in the sum where each term corresponds to a geometric series, so $Z(S)$ can easily be calculated in $O(k)$. $f(S)$ can be calculated in $O(k)$ using the following algorithm which clearly has $O(k)$ runtime.

Algorithm 3 TOPKGMMSAMPLING (TOPKGMM)

Input: TopKGMM $\mathcal{D}(\beta, w)$.
Output: Sampled Top- k ordering τ proportional to $\mathcal{D}$
 $f(S) = \text{PROFILE PROBABILITY}$
 Sample S proportional to $\mathbb{P}_{\mathcal{D}}[S] = f(S) \cdot Z(S)$.
Return PRIM(β, w, S).

Algorithm 4 PROFILE-BASED RIM(PRIM)

Input: TopKGMM(β, w), $S = \{s_1, s_2, \dots, s_\ell\} \subseteq [k]$ where $s_1 < s_2 < \dots < s_\ell$.
Output: Top- k list $\tau \in T_\sigma(S)$.
 $A \leftarrow$ ordered random $k - \ell$ elements from $[n] \setminus [k]$
for $s \leftarrow s_\ell, s_{\ell-1}, \dots, s_1$ **do**
 Insert s at position j in A w.p. $\frac{\exp(-\beta w_s j)}{\sum_{x=0}^{|A|} \exp(-\beta w_s x)}$.
end for
Return $\tau = A$

Algorithm 5 PROFILE PROBABILITY

Input: Profile $S = \{s_1, s_2, \dots, s_\ell\} \subseteq [k]$.
Output: $f(S)$
 $Q(S) \leftarrow \binom{k-\ell}{2}$.
 $x, y \leftarrow 0, 0$
for $j \in \{k, k-1, \dots, 1\}$ **do**
 $I_j \leftarrow k - \ell + x$
 $P_j \leftarrow n - 2k + \ell + y$
 if $j \in S$ **then**
 $x \leftarrow x + 1$
 else
 $y \leftarrow y + 1$
 end if
end for
Return $f(S) := w_0 p Q(S) + \sum_{j \in [k] \setminus S} w_j (I_j(S) + p P_j(S))$.

□

Using Lemma 3.3 and analyzing Algorithm 3, we can prove Theorem 3.1.

Proof of Theorem 3.1. Any $\tau \in T_{k,N}$ with $S = \tau \cap [k]$, according to Algorithm 4, is sampled with probability

$$\frac{1}{\binom{n-k}{k-l}(k-l)!} \cdot \frac{\sum_{j \in S} \exp(-\beta w_j I_j)}{\sum_{j=1}^l \prod_{r=0}^{k-j} \exp(-\beta w_{s_j} r)} = \frac{\sum_{j \in S} \exp(-\beta w_j I_j)}{Z(S)}.$$

Since each S , is sampled by $\frac{\exp(-\beta f(S)) Z(S)}{\sum_{S \subseteq [k]} \exp(-\beta f(S)) Z(S)}$, we get that τ is sampled with

$$\frac{\exp(-\beta f(S)) Z(S)}{\sum_{S \subseteq [k]} \exp(-\beta f(S)) Z(S)} \cdot \frac{\exp(\sum_{j \in S} -\beta w_j I_j)}{Z(S)} = \frac{\exp(-\beta(f(S) + \sum_{j \in S} -w_j I_j))}{\sum_{S \subseteq [k]} \exp(-\beta f(S)) Z(S)}$$

where the denominator as previously discussed in proof of Lemma 3.3, is equal to normalizing factor $\mathcal{M} = \sum_{\tau \in T_{k,N}} \mathbb{P}_{\mathcal{D}}[\tau]$. Since numerator is just restating Equation (TopKGMM), each τ is sampled w.r.t Equation (TopKGMM).

Time complexity An upperbound on the number of profiles is $2^\gamma; \gamma = \min\{k, n - k\}$. Therefore $Z(S)$ and $f(S)$ can be computed in $O(2^\gamma k)$ by Lemma 3.3, and we can sample S in $O(2^\gamma k)$. Next, fixing S , we need to execute Algorithm 4, where we first need to sample $k - \ell$ elements which can be done in $O(k \log n)$. And then inserting elements of S with respect to probabilities that can be calculated in $O(k)$ for each element of S , so requiring $O(k^2)$ time for the recursive insertion loop. Hence overall the algorithm runs in $O(2^\gamma k + k \log n + k^2)$; $\gamma = \min\{k, n - k\}$. $\square$

A.2 Missing details from Section 4.1

In this section, we provide the details in design of DYPCHIP. The main remaining part is the calculation of $\pi_S(i, j, q)$ for given $\mathcal{A}, i = 1, \dots, r, r + 1, \dots, r + \ell, q = 0, 1, \dots, \ell$, and $j = r + 1, r + 2, \dots, r + q, k$; where $r = |\bar{\mathcal{A}}|$ and $\ell = |S|$. We remind the reader that in $\pi(i, j, q)$:

- i indicated the index of an item a whose choice probability is being calculated. We use L to refer to the ordered set $\{a_1, a_2, \dots, a_r, s_\ell, s_{\ell-1}, \dots, s_1\}$ a ranking of the elements in $\bar{\mathcal{A}} \cup S$ where $a_1, a_2, \dots, a_r$ is an arbitrary ranking of $\bar{\mathcal{A}}$ and $s_\ell, s_{\ell-1}, \dots, s_1$ is the ranking of S used in PRIM. Thus, when we say a_i we mean the i th element in L .
- j indicates the position of a_i if it is the winner until the q th iteration of the for loop in PRIM. When $j = k$ we mean that the winner is in $\bar{\tau}$.
- q denote that so far we are considering only elements that have been sampled until the q th iteration of the for loop in PRIM.

Since $\pi_S(i, j, q)$ can only take non-zero value when a_i is sampled in the top- k list, we either have $a_i \in \bar{\tau}$ or $a_i \in S$. We analyze each of these two possible cases separately. We first initialize the DP table for $q = 0$ by considering $i = 1, 2, \dots, r$. Then we consider items that are in S in the reverse order of their priority (consistent with the Algorithm 4).

1. Let $\bar{\mathcal{A}} = \mathcal{A}^\emptyset \cap \bar{\tau}$. Before start of the loop in PRIM, only elements of $\bar{\mathcal{A}}$ have been sampled and have nonzero probability of being selected. These elements are the first r elements in our list. Thus, for $i = 1, 2, \dots, r$, we store these probabilities in the DP table as:

$$\left\{ \begin{array}{l} \pi(i, j, 0) = \underbrace{\frac{1}{n - k - j + 1}}_{\text{the probability that } i \text{ is sampled at position } j} \underbrace{\prod_{j'=1}^{j-1} \left(1 - \frac{r}{n - k - j'}\right)}_{\text{the probability that no element of } \bar{\mathcal{A}} \text{ is sampled in positions } j - 1} \quad \text{for } j \leq r \\ \pi(i, k, 0) = \frac{\binom{n - k - (r+1)}{k - \ell}}{\binom{n - k}{k - \ell}} \cdot \frac{1}{r+1} \quad \text{the case that } \bar{\mathcal{A}} \subseteq \bar{\tau} \end{array} \right. \quad (6)$$

2. For $q = 1, 2, \dots, \ell$, consider the for loop in PRIM and let a_{cur} be the element that is inserted in the top- k list at iteration q of this loop. Therefore, $a_j = s_{\ell+1-q}$, and in our list it is ranked $(r + q)$ th, thus, $cur = r + q$. We distinguish between the two cases where $a_{cur} \in \mathcal{A}^\emptyset$ or not.

Case 1. $a_{cur} \notin \mathcal{A}^\emptyset$: since a_{cur} is not a choice option, we have that $\pi_S(cur, j, q) = 0$ for all $j \leq q$. For any $i < cur$, if a_{cur} is inserted higher than the previous winner, the winner position will increment. If a_{cur} is inserted lower than the previous winner, the winner position will not change. Therefore $\forall i < cur$,

$$\pi_S(i, j, q) = \pi_S(i, j, q-1) \cdot \text{PRIMPOS}(S, > j, q) + \pi_S(i, j-1, q-1) \cdot \text{PRIMPOS}(S, \leq j-1, q),$$

where $\text{PRIMPOS}(S, > j, q)$ is the probability that at iteration q of PRIM we insert an element in a position after (lower than) j , and $\text{PRIMPOS}(S, \leq j, q)$ is it is inserted before (higher than) j . Note that $\text{PRIMPOS}(S, \leq j, q) = 1 - \text{PRIMPOS}(S, > j, q)$, and the probability $\text{PRIMPOS}(S, \leq j, q)$ can be directly calculated using Equation (3) by summing over all insertion probabilities for $j' = 1, 2, \dots, j$.

Case 2. $a_{cur} \in \mathcal{A}^\emptyset$: In this case, if a_{cur} is positioned higher than the previous winner, it will become the new winner. Otherwise, the position of the previous winner will increment. Thus,

$$\forall i < cur, \quad \pi_S(i, j, q) = \pi_S(i, j, q-1) \cdot \text{PRIMPOS}(S, > j, q) .$$

Furthermore,

$$\pi_S(cur, j, q) = \text{PRIMPOS}(S, j, q) \cdot \sum_{i < cur} \sum_{j'=j}^{\ell} \pi_S(i, j', q-1) ,$$

where $\text{PRIMPOS}(S, j, q)$ is the probability that at the q th iteration of PRIM we insert an element in location j which again can be directly calculated from Equation (3).

Proof of Theorem 4.1. The above analysis forms a proof for the correctness of DYPCHIP. □

A.3 Missing proofs and details from Section 5

In this section we provide the missing proof for correctness and sample complexity of BUCCHOI and FINDER and we present the missing details.

Let us first present the pseudocode of SORTCNTR which is called in BUCCHOI.

Algorithm 6 SORTCNTR

Input:

choice oracle $\mathcal{C}_\tau$; $\tau \sim \mathcal{D}$ where $\mathcal{D}$ is a topKGMM with center τ^*

U containing elements of τ^* as an (unsorted) set,

r assortment size

m sample size

B set of elements out of center

Output: τ^* sorted

$k = |U|$

for $i = 1 : k$ **do**

find the top element in U and delete it

$T = U$

repeat

T maintains potential top elements of U

Divide T to non-intersecting assortments of size r : $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_\gamma$; $\gamma = \lceil i/r \rceil$ # use elements of B for $\mathcal{A}_\gamma$ if needed

for $\kappa = 1 : \gamma$ **do**

$T = \emptyset$

collect choice samples by showing $\mathcal{A}_\kappa$ to customers and find the top element of $\mathcal{A}_\kappa$

$S = \emptyset$

for $j = 1 : m$ **do**

$S = S \cup \mathcal{C}_\tau(\mathcal{A}_\kappa)$

end for

$T = T \cup \{\text{FINDER}(\mathcal{A}_\kappa, S)\}$

end for

until $|T| = 1$

$U = U \setminus T$

end for

Return τ^*

SORTCNTR When BUCCHOI finds all the elements that are in the center, we call SORTCNTR to find the ranking of these items as a list. In the for loop the item which is positioned the i th will be found and deleted from U . In other words, we first find the top element of U this is the element ranked first, after deleting it, we find the top element in the remainder which is ranked second and so on.

In order to find the top element we use a Repeat-Until loop. In this loop U is divided to assortments of size r and using FINDTOP we find the winner in each assortment and continue by having a tournament between the winners until the winner of all (which is the top element) is found.

Note that the repeat-until loop iterates at most $\log_r(k)$ times. We call FINDTOP at each iteration $\lceil k/r \rceil$ times, and since we have for loop the total number of times that FINDTOP is called is bounded by $\Theta(k^2 \log_r(k))$.

We now present the proof lemmas and theorems. As before, without loss of generality we assume that $\tau^* = 1 \succ 2 \succ \dots k \succ \{k+1, \dots, n\}$.

Lemma A.1. *Let $i < j \in \mathcal{A}_t$, and assume $\mathcal{I} = \{i_1, i_2, \dots, i_\ell\} \subseteq \mathcal{A}_t$ is such that $i < i_\kappa < j$ for all $i_\kappa \in \mathcal{I}$. We have:*

$$\mathcal{C}_{\mathcal{D}}(i, \mathcal{A}_t) \geq \mathcal{C}_{\mathcal{D}}(j, \mathcal{A}_t) \cdot \exp\left(\beta(w_i + \sum_{\kappa=1}^{\ell} w_{i_\kappa})\right).$$

Consider now a fixed $\mathcal{A}$ and let $a_r, a_{r-1}, \dots, a_1, \mathcal{A}_0$ be such that, for each a_i , there are i elements in $\mathcal{A}^\varnothing$ which are ranked under it by τ^* , i.e., $|\{a \in \mathcal{A}^\varnothing \mid a \succ a_i\}| = i$. As a consequence of this definition we have: $|\mathcal{A}| = r$ and $\mathcal{A}^\varnothing \cap \tau^* = \{a_r, a_{r-1}, \dots, a_1\}$, and $\mathcal{A}_0 \subseteq \bar{\tau}$. We denote any arbitrary element of $\mathcal{A}_0$ by a_0 .

Lemma A.2. *For any $\kappa = r-1, r-2, \dots, 1, 0$, we have:*

$$\mathcal{C}(a_r, \mathcal{A}) - \mathcal{C}(a_\kappa, \mathcal{A}) \geq \frac{1 - \exp(-\beta w_{a_r})}{1 + r \exp(-\beta w_{a_r})}.$$

Proof. From Lemma A.1 we can immediately conclude that: for any $\kappa = r-1, r-2, \dots, 1, 0$, $\mathcal{C}(a_r, \mathcal{A}) \cdot \exp(-\beta w_{a_r}) \geq \mathcal{C}(a_\kappa, \mathcal{A})$.

Let us now rearrange this inequality:

$$\begin{aligned} \mathcal{C}(a_r, \mathcal{A}) \cdot \exp(-\beta w_{a_r}) &\geq \mathcal{C}(a_\kappa, \mathcal{A}) \\ \mathcal{C}(a_r, \mathcal{A}) \cdot \exp(-\beta w_{a_r}) - \mathcal{C}(a_r, \mathcal{A}) &\geq \mathcal{C}(a_\kappa, \mathcal{A}) - \mathcal{C}(a_r, \mathcal{A}) \\ \mathcal{C}(a_r, \mathcal{A})[1 - \exp(-\beta w_{a_r})] &\leq \mathcal{C}(a_r, \mathcal{A}) - \mathcal{C}(a_\kappa, \mathcal{A}) \end{aligned} \quad (7)$$

On the other hand we have that: $\sum_{\kappa=1}^r \mathcal{C}(a_\kappa, \mathcal{A}) + \sum_{a_0 \in \mathcal{A}_0} \mathcal{C}(a_0, \mathcal{A}) = 1$. Thus,

$$\begin{aligned} 1 &\leq \mathcal{C}(a_r, \mathcal{A}) + r\mathcal{C}(a_r, \mathcal{A}) \cdot \exp(-\beta w_{a_r}) \\ \mathcal{C}(a_r, \mathcal{A}) &\geq 1/(1 + r \exp(-\beta w_{a_r})). \end{aligned}$$

We now substitute this lower bound in Equation (7) and obtain the premise. $\square$

Consider now for any $i, j \in \mathcal{A}^\varnothing$, for $t = 1, 2, \dots, T$, assume we keep displaying assortment $\mathcal{A}$ and consider the following random variable:

$$X_{ij}^t = \begin{cases} +1 & \text{if } i \text{ is chosen} \\ -1 & \text{if } j \text{ is chosen} \\ 0 & \text{otherwise} \end{cases}$$

We denote $Y_{ij} = \sum_{t=1}^T X_{ij}^t / T$, we have that:

$$\mathbb{E}[Y_{ij}] = \mathcal{C}(i, \mathcal{A}) - \mathcal{C}(j, \mathcal{A}).$$

In the following lemma we let $w_{\min}$ be the minium weight assigned to the elements in τ^* , i.e., $w_{\min} \doteq \min_{i \in k} w_i$.

Lemma A.3 (Restatement of Lemma 5.2). Assume that $w_{\min} \geq \log(3)/\beta$ and let $r = |\mathcal{A}|$ and $\zeta \geq 1$ arbitrary constant. If $\mathcal{A}$ appears at least $\Theta(\zeta(r+1)^2 \log n)$ times among the displayed assortments, with probability at least $1 - o(n^{-\zeta})$ we have: if $\mathcal{A}^\varnothing \cap \tau^* \neq \emptyset$ we are able to identify $i \in \mathcal{A}^\varnothing$ such that $i < j$ for any $j \in \mathcal{A}^\varnothing \setminus \{i\}$, otherwise, we can conclude that $\mathcal{A}^\varnothing \cap \tau^* = \emptyset$.

Proof. Note that if $\mathcal{A}^\varnothing \cap \tau^* \neq \emptyset$ there is at least one element with rank r in $\mathcal{A}^\varnothing$. For this element we may apply Lemma A.2. Note that this element is unique. If $\mathcal{A}^\varnothing \cap \tau^* = \emptyset$, then the choice probability of all the elements in $\mathcal{A}^\varnothing$ are equal to $1/(r+1)$.

Assuming $w_{\min} \geq \log 3/\beta$, we may bound the right-hand side of Lemma A.2 as follows:

Since $a_r \in \tau^*$

$$1 - \exp(-\beta w_{a_r}) \geq 1 - \exp(-\beta w_{\min}) \geq 1 - \exp(\log 3) \geq 2/3.$$

Therefore,

$$\frac{1 - \exp(\beta w_{a_r})}{1 + r \exp(\beta w_{a_r})} = \frac{1 - \exp(-\beta w_{a_r})}{1 - r(1 - \exp(-\beta w_{a_r})) + r}$$

We substitute $x = 1 - \exp(-\beta w_{a_r})$, note $1 \geq x \geq 2/3$.

$$\frac{1 - \exp(-\beta w_{a_r})}{1 + r \exp(-\beta w_{a_r})} = \frac{x}{1 + r - rx}$$

Note that we have: $\frac{1+r}{1+2r} \leq 2/3$ for $r \geq 1$.

Thus,

$$\begin{aligned} \frac{x}{1 + r - rx} \geq \frac{1}{1 + r} &\iff x(1 + r) \geq 1 + r - rx \\ &\iff x + rx \geq 1 + r - rx \\ &\iff x(1 + 2r) \geq 1 + r \\ &\iff x \geq (1 + r)/(1 + 2r) \\ &\iff x \geq 2/3. \end{aligned}$$

We now consider random variables Y_{ij} for any $i, j = 1, 2, \dots, n$ as defined before.

We have:

$$\mathbb{E}[Y_{ij}] \begin{cases} \geq 1/(r+1) & \text{if } i = a_r \\ = 0 & \text{if } i, j \in \tau^* \end{cases} \quad (8)$$

Using Hoeffding bound, we have:

$$\mathbb{P}(|Y_{ij} - \mathbb{E}[Y_{ij}]| \geq 1/2(1+r)) \leq 2 \exp\left(-\frac{T}{32(1+r)^2}\right)$$

Which for $T \geq 64\zeta(r+1)^2 \log(n+r)$ is bounded by:

$$\begin{aligned} 2 \exp\left(-\frac{64\zeta(r+1)^2 \log(n+r)}{32(1+r)^2}\right) &= 2 \exp(-2\zeta \log(n+r)) \\ &= 2n^{-2\zeta} \cdot r^{-2\zeta}. \end{aligned}$$

In order to find a_r , we estimate Y_{ij} for all $\Theta(r^2)$ pairs of ij in $\mathcal{A}^\emptyset$. Using the above bound, and applying a union bound over all pairs we have:

With probability $1 - 2n^{-2\zeta}$:

$$\forall i, j \in \mathcal{A}^\emptyset, |Y_{ij} - \mathbb{E}[Y_{ij}]| \leq 1/2(1 + r)$$

Using Equation (8), this means that:

With probability $1 - 2n^{-2\zeta}$:

$$\begin{aligned} \forall j \in \mathcal{A}^\emptyset, Y_{a_r j} &> 1/2(1 + r), \text{ and} \\ \forall i, j \in \bar{\tau}^*, Y_{ij} &< 1/2(1 + r) \end{aligned}$$

We use the above rule to find a_r : first we find all Y_{ij} s, and output the i for which we have:

$$\forall j \in \mathcal{A}^\emptyset \setminus \{i\}, Y_{ij} > 1/2(1 + r). \quad (9)$$

Note that if a_r exists, there will be one unique $i \in \mathcal{A}^\emptyset$ Equation (9) holds. Therefore, we take $a_r = i$. Alternatively, if there is no element with rank r we have $\forall i, j \in \mathcal{A}^\emptyset, i, j \in \bar{\tau}^*$ thus, $Y_{ij} < 1/2(1 + r)$. $\square$

Theorem A.4 (Restatement of Theorem 5.1). *Assume that $\beta \geq \log 3/w_{\min}$ and $\emptyset \notin \tau^*$. By only receiving N and r as input and being able to collect choice samples $\mathbf{D}$ adaptively, with probability at least $1 - o(1)$, BUCCHOI is able to learn τ^* and k from $\mathbf{D}$ using only $\Theta(r^2 \log n)$ choice samples.*

Proof. Note that BUCCHOI calls FINDTOP at most $\Theta(k + n/r + k^2 \log_r k) = \Theta(n/r + k^2 \log_r k)$ times. We take $\zeta = 3$, and using Lemma 5.2 we have that by $\Theta(3(r+1)^2 \log n)$ choice samples which are obtained from presentation of $\mathcal{A}$ with probability $1 - o(n^{-3})$ we will be able to identity the top element of $\mathcal{A}$. Since the total number of pairs i, j is $\Theta(n/r + k^2 \log k) = (n^2 \log n)$, we need to take a union bound over $n^2 \log n$ variables thus, in total the probability of failure is bounded by $o(n^2 n^{-3}) = o(1)$, from which we conclude the premise. $\square$

B Experimental setup and extra tables and figures

B.1 Details of learning choice probabilities on the Sushi data-set.

As explained in the main body, we have divided the data set into two non-intersecting parts for training (80%) and test sets (20%). We use the training set to learn choice probabilities. First we employ BUCCHOI-II (Algorithm 7). BUCCHOI-II uses the top-10 sushi types as samples generated from an unknown distribution and collects choice data from it by actively selecting assortments. Since the result of BUCCHOI-II may produce partial orders, we break the ties, by using a score function based on the random variables X_{ij} that are used in FINDTOP. In particular, since X_{ij} shows the number of times item i beats j we use $\sum_{j \in N} X_{ij}$ as a tie breaking score for item i .

This is needed because we need to feed the learned center from BUCCHOI-II to DYPCHIP to learn choice probabilities and DYPCHIP receives as input a top- k list. Note that k is unknown to the algorithm and we let BUCCHOI learn it. If k is too large (>15) we truncate the learned center to reduce time complexity.

We tune parameters β and p by performing a grid search over a range of values as in Table 1. From this table we conclude that $p = 0.5$ and $\beta = 0.05$ have the best test error. Thus, we select them as the model parameters. The mean and std of test error of the best parameters are reported in Figure 1a.

We handle learning mixture models by clustering the data into several clusters. First we use $\mathcal{K}^p$ as a distance metric taking $p \in [0.01, 0.25, 0.05, 0.75, 0.1, 0.25, 0.5, 1, 1.5, 2.2.5, 5]$ and then we divide data set to 2, 3 or 5 clusters. Most of the clusters we obtain have a negative silhouette coefficient, and the only positive ones are:

Algorithm 7 BUCCHOI-II

Input, N : set of products, m : number of samples, choice oracle $\mathcal{C}_\tau$; $\tau \sim \mathcal{D}$ where $\mathcal{D}$ is a topKGMM.
Output center of $\mathcal{D}$
 $T = \emptyset$
for $i \in N$ **do**
 $S = \emptyset$
 $\mathcal{A} = \{i\}$
 for $j = 1 : m$ **do**
 $c_j = \mathcal{C}_\tau(\mathcal{A})$
 $\tilde{S} = S \cup \{c_j\}$
 end for
 $T = T \cup \text{FindTop}(\mathcal{A}, S)$
end for
for $i, j \in T$ **do**
 $\mathcal{A} = \{i, j\}$
 for $j = 1 : m$ **do**
 $c_j = \mathcal{C}_\tau(\mathcal{A})$
 $\tilde{S} = S \cup \{c_j\}$
 end for
 if $i == \text{FindTop}(\mathcal{A}, S)$ **then**
 $\sigma[i] > \sigma[j]$
 else
 $\sigma[i] < \sigma[j]$
 end if
end for
Return σ

p= 0.1 num clusters = 2: silhouette score = 0.006287393358176328
p= 1.25 num clusters = 2: silhouette score = 0.007796245381136862
p= 2.5 num clusters = 2: silhouette score = 0.004498089502999035
p= 5 num clusters= 2: silhouette score = 0.011002010781645042
p= 5 num clusters = 3: silhouette score = 0.0023222431733978116

As we can see, even when the silhouette scores are positive they are pretty low, which suggests that one cluster is the best choice. This is consistent with the test error we obtained, as the model trained without clustering has a lower test error compared to the model trained on two clusters or more; see Figure 1b.

$\beta \backslash p$	0.05	0.1	0.25	0.5	0.75	1	1.25	1.5	1.75	2
0.01	0.1162	0.1047	0.0888	0.0879	0.0925	0.0979	0.1030	0.1087	0.1145	0.1202
0.025	0.1133	0.1007	0.0858	0.0872	0.0922	0.09768	0.1028	0.1085	0.1144	0.1201
0.05	0.1083	0.0938	0.0817	0.0852	0.0907	0.0964	0.1017	0.1075	0.1135	0.1193
0.075	0.1032	0.0868	0.0797	0.0841	0.0898	0.0956	0.1011	0.1070	0.1130	0.1188
0.1	0.0980	0.0798	0.0768	0.0819	0.0880	0.0941	0.0997	0.1058	0.1119	0.1178
0.25	0.0709	0.0598	0.0658	0.0732	0.0808	0.0879	0.09433	0.1009	0.1075	0.1139
0.5	0.0445	0.0555	0.0629	0.07098	0.0789	0.0863	0.0929	0.0997	0.1064	0.1128
1	0.0504	0.0619	0.0677	0.0748	0.0821	0.0890	0.0953	0.1018	0.1083	0.1146
1.5	0.0696	0.07479	0.0773	0.0825	0.0885	0.09454	0.1001	0.1061	0.1122	0.1181
2	0.0791	0.08115	0.0821	0.0863	0.0917	0.0972	0.1025	0.1082	0.1141	0.1198
2.5	0.0825	0.0834	0.0838	0.08769	0.0928	0.09824	0.1033	0.1090	0.1148	0.1204
5	0.0851	0.0851	0.0851	0.0887	0.0937	0.0989	0.1039	0.1095	0.1153	0.1209

Table 1: Average tests errors for several choices of p and β used for parameter tuning. No clustering has been performed. The average test error we obtain for MNL is 0.168.

	$\beta = 0.05$	0.1	0.25	0.5	0.75	1	1.25	MNL
$p = 0.1$	0.0997	0.0771	0.0732	0.0748	0.0785	0.083	0.0876	0.1877
$p = 1.25$	0.0788	0.09273	0.1022	0.1116	0.1224	0.1324	0.1409	0.1883

Table 2: average test error of learning choice probabilities, comparison of Mallows model with different β s and MNL; 2 clusters are generated based on distance function $\mathcal{K}^p$ for given p . The best values are highlighted.

B.2 Runtime of PRIM and DYPCHIP

	pre-processing time					amortized time per sample				
	$k = 8$	$k = 10$	$k = 12$	$k = 14$	$k = 16$	$k = 8$	$k = 10$	$k = 12$	$k = 14$	$k = 16$
$n = 200$	0.007	0.035	0.19	1.06	8.64	5.67e-05	9.59e-05	0.00022	0.00074	0.0032
$n = 500$	0.008	0.044	0.23	1.24	7.83	7.89e-05	0.00012	0.00025	0.00076	0.0028
$n = 1000$	0.012	0.079	0.29	1.46	8.39	0.00011	0.00022	0.00031	0.00082	0.0030

Table 3: average (among 10 runs) runtime of sampling algorithm PRIM in seconds, $\beta = 0.2$

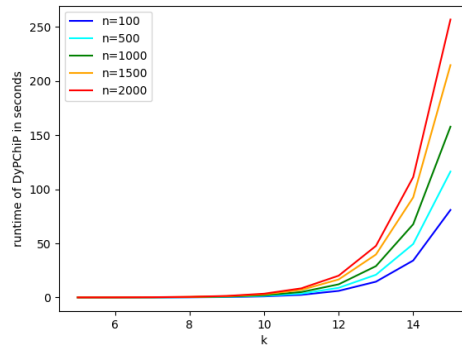


Figure 3: Runtime of DYPCHIP (in seconds) for various choices of n, k . Here $\beta = 0.6$, $p = 0.5$ and $w = 2\vec{1}$, size of assortment $r = k - 2$.

B.3 Additional tables and figures for the analysis of sample complexity of BUCCHOI and FINDTOP on synthetic data

	$k = 6$	$k = 10$		$k = 6$	$k = 10$
$n = 1000$	6.95 ± 5.06	0.4 ± 0.48	$n = 1000$	0.1 ± 0.3	0.0 ± 0.0
$n = 5000$	228.5 ± 53.3	228.6 ± 64.56	$n = 5000$	0.35 ± 0.50	1.92 ± 0.92
$n = 10000$	934.05 ± 209.35	870.1 ± 259.64	$n = 10000$	2.95 ± 2.28	3.0 ± 3.16
$n = 20000$	3505.05 ± 409.87	3282.55 ± 708.9	$n = 20000$	5.1 ± 3.5	9.0 ± 13.2

Table 4: Kendall's Tau distance of true and learned center by BUCCHOI with 100 (left table) or 200 (right table) samples for $n = 1000, 5000, 10000, 20000$ and $k = 6, 10$. We let $\beta = 0.6$. distance reported as *mean* $\pm$ *std* these numbers are taken from 10 runs of the algorithm.

	$\beta = 0.4$	$\beta = 0.6$	$\beta = 0.8$	$\beta = 1$	$\beta = 1.2$
50 samples	12.75 ± 4.4	8.25 ± 7.7	7.05 ± 5.64	4.35 ± 7.5	0.7 ± 1.65
100 samples	3.5 ± 0.67	0.35 ± 0.549	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0

Table 5: Kendall's Tau distance of true and learned center by BUCCHOI with 50 and 100 samples for various β , $n = 1000, k = 12, p = 0.5$ reported as *mean* $\pm$ *std*. mean and standard deviation is taken for 10 runs of the algorithm for each set of parameters.

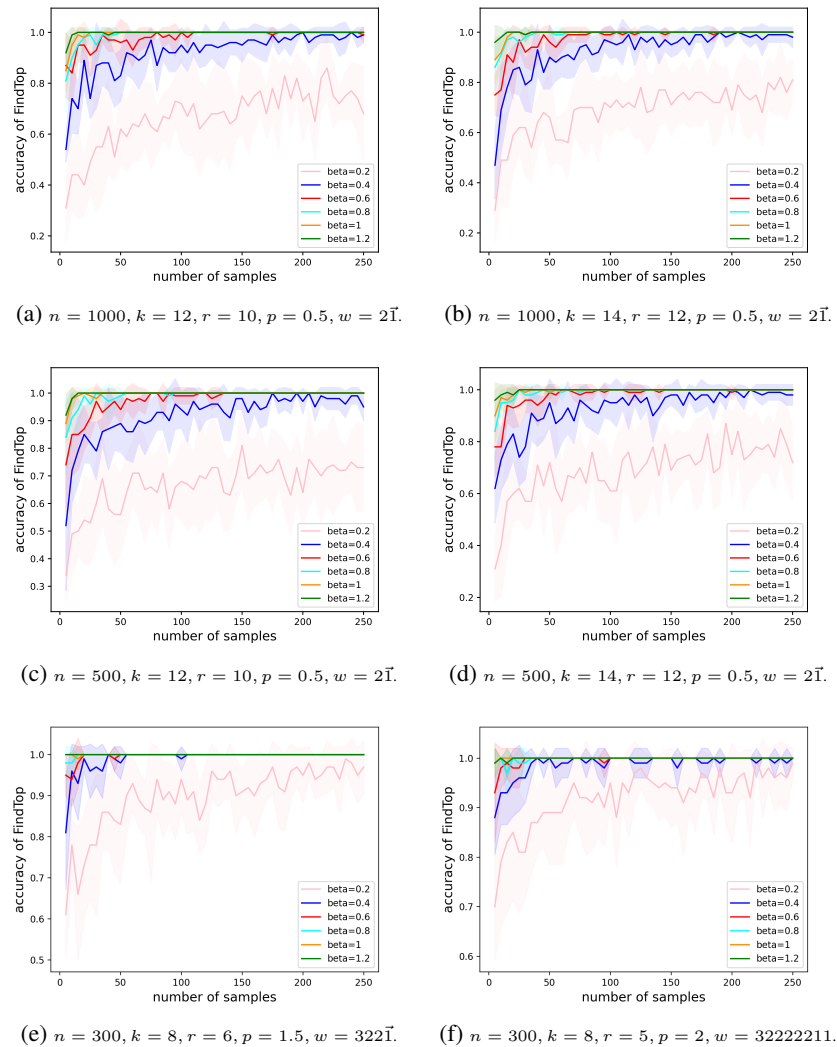


Figure 4: Sample complexity of FINDTOP for a wide range of parameters.

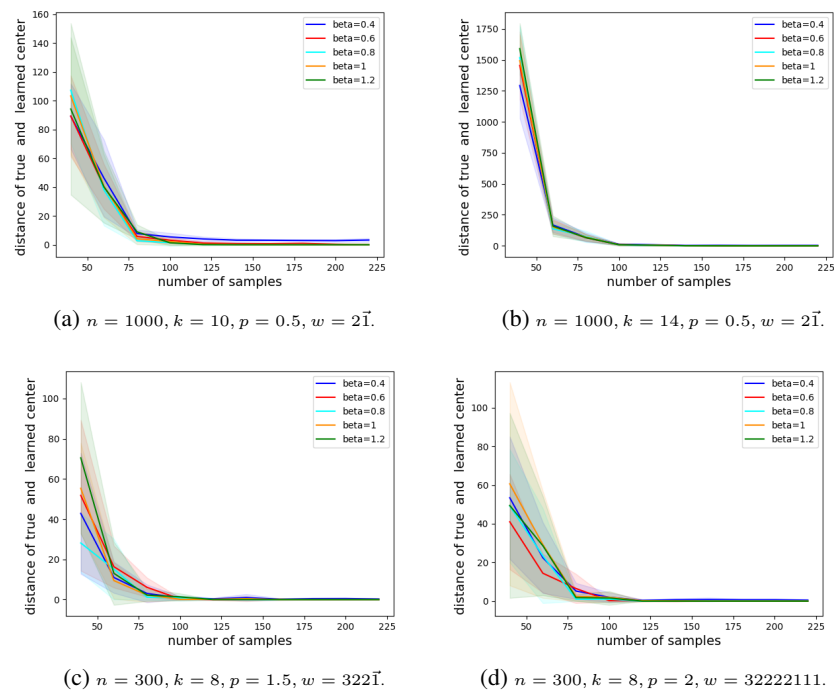


Figure 5: Sample complexity of BUCCHOI for a wide range of parameters. y axis shows the Kendall's Tau distance between true and learned center while x axis shows sample complexity