# Instance-Sensitive Algorithms for Pure Exploration in Multinomial Logit Bandit

**Nikolai Karpov, Qin Zhang**
Indiana University Bloomington
Luddy Hall, RM 3044
700 North Woodlawn Avenue
Bloomington, IN 47408-3901, USA
nkarpov@iu.edu, qzhangcs@indiana.edu

## Abstract

Motivated by real-world applications such as fast fashion retailing and online advertising, the Multinomial Logit Bandit (MNL-bandit) is a popular model in online learning and operations research, and has attracted much attention in the past decade. In this paper, we give efficient algorithms for *pure exploration* in MNL-bandit. Our algorithms achieve *instance-sensitive* pull complexities. We also complement the upper bounds by an almost matching lower bound.

## Introduction

We study a model in online learning called *multinomial logit bandit* (*MNL-bandit* for short), where we have $N$ substitutable items $\{1, 2, \ldots, N\}$, each of which is associated with a known reward $r_i \in (0, 1]$ and an *unknown* preference parameter $v_i \in (0, 1]$. We further introduce a null item 0 with reward $r_0 = 0$, which stands for the case of "no-purchase". We set $v_0 = 1$, that is, we assume that the no-purchase decision is the most frequent case, which is a convention in the MNL-bandit literature and can be justified by many real-world applications to be mentioned shortly.

Denote $[n] \triangleq \{1, 2, \ldots, n\}$. Given a subset (called *an assortment*) $S \subseteq [N]$, the probability that one chooses $i \in S \cup \{0\}$ is given by

$$p_i(S) = \frac{v_i}{v_0 + \sum_{j \in S} v_j} = \frac{v_i}{1 + \sum_{j \in S} v_j}.$$

Intuitively, the probability of choosing the item $i$ in $S$ is proportional to its preference $v_i$. This choice model is called the *MNL choice model*, introduced independently by Luce (1959) and Plackett (1975). We are interested in finding an assortment $S \subseteq [N]$ such that the following expected reward is maximized.

**Definition 1** (expected reward)**.** *Given an assortment $S \subseteq [N]$ and a vector of item preferences $\mathbf{v} = (v_1, \ldots, v_N)$, the expected reward of $S$ with respect to $\mathbf{v}$ is defined to be*

$$R(S, \mathbf{v}) = \sum_{i \in S} r_i p_i(S) = \sum_{i \in S} \frac{r_i v_i}{1 + \sum_{j \in S} v_j}.$$

The MNL-bandit problem was initially motivated by fast fashion retailing and online advertising, and finds many applications in online learning, recommendation systems, and operations research (see Avadhanula (2019) for an overview). For instance, in fast fashion retailing, each item corresponds to a product and its reward is simply the revenue generated by selling the product. The assumption that $v_0 \geq \max\{v_1, \ldots, v_N\}$ can be justified by the fact that most customers do not buy anything in a shop visit. A similar phenomenon is also observed in online advertising where it is most likely that a user does not click any of the ads on a webpage when browsing. We naturally want to select a set of products/ads $S \subseteq [N]$ to display in the shop/webpage so that $R(S, \mathbf{v})$, which corresponds to revenue generated by customer/user per visit, is maximized.

We further pose a capacity constraint $K$ on the cardinality of $S$, since in most applications the size of the assortment cannot exceed a certain size. For example, the number of products presented at a retail shop is capped due to shelf space constraints, and the number of ads placed on a webpage cannot exceed a certain threshold.

In the MNL-bandit model, we need to *simultaneously* learn the item preference vector $\mathbf{v}$ and find the assortment with the maximum expected reward under $\mathbf{v}$. We approach this by repeatedly selecting an assortment to present to the user, observing the user's choice, and then trying to update the assortment selection policy. We call each observation of the user choice given an assortment a *pull*. We are interested in minimizing the number of pulls, which is the most expensive part of the learning process.

In bandit theory we are interested in two objectives. The first is called *regret minimization*: given a pull budget $T$, try to minimize the accumulated difference (called *regret*) between the sum of expected rewards of the optimal strategy in the $T$ pulls and that of the proposed learning algorithm; in the optimal strategy we always present the best assortment (i.e., the assortment with the maximum expected reward) to the user at each pull. The second is called *pure exploration*, where the goal is simply to identify the best assortment.

Regret minimization in MNL-bandit has been studied extensively in the literature (Rusmevichientong, Shen, and Shmoys 2010; Sauré and Zeevi 2013; Davis, Gallego, and Topaloglu 2013; Agrawal et al. 2016, 2017; Chen and Wang 2018). The algorithms proposed by Rusmevichientong, Shen, and Shmoys (2010); Sauré and Zeevi (2013) for the regret minimization problem make use of an "exploration then exploitation" strategy, that is, they first try to find the best assortment and then stick to it. However, they need the prior knowledge of the *gap* between the expected reward of the optimal assortment and that of the second-best assortment, which, in our opinion, is *unrealistic* in practice since the preference vector $\mathbf{v}$ is unknown at the beginning.

In this paper we focus on pure exploration. Pure exploration is useful in many applications. For example, the retailer may want to perform a set of customer preference tests (e.g., crowdsourcing) to select a good assortment before the actual store deployment. We propose algorithms for pure exploration in MNL-bandit *without* any prior knowledge of preference vector. Our algorithms achieve *instance-sensitive* pull complexities which we elaborate next.

**Instance Complexity.** Before presenting our results, we give a few definitions and introduce *instance complexities* for pure exploration in MNL-bandit.

**Definition 2** (best assortment $S_{\mathbf{v}}$ and optimal expected reward $\theta_{\mathbf{v}}$)**.** *Given a capacity parameter $K$ and a vector of item preferences $\mathbf{v}$, let*

$$S_{\mathbf{v}} \triangleq \arg \max_{S \subseteq [N]:|S| \leq K} R(S, \mathbf{v})$$

*denote the best assortment with respect to $\mathbf{v}$. If the solution is not unique then we choose the one with the smallest cardinality which is unique (see the discussion after Lemma 2). Let $\theta_{\mathbf{v}} \triangleq R(S_{\mathbf{v}}, \mathbf{v})$ be the optimal expected reward.*

Denote $\eta_i \triangleq (r_i - \theta_{\mathbf{v}})v_i$; we call $\eta_i$ the *advantage* of item $i$. Suppose we have sorted the $N$ items according to $\eta_i$, let $\eta^{(j)}$ be the $j$-th largest value in the sorted list.

**Definition 3** (reward gap $\Delta_i$)**.** *For any item $i \in [N] \backslash S_{\mathbf{v}}$, we define its reward gap to be*

$$\Delta_i \triangleq \begin{cases} \eta^{(K)} - \eta_i, & \text{if } |S_{\mathbf{v}}| = K, \\ -\eta_i, & \text{if } |S_{\mathbf{v}}| < K. \end{cases}$$

*and for any item $i \in S_{\mathbf{v}}$, we define*

$$\Delta_i \triangleq \bar{\Delta} = \min \left\{ \left( \eta^{(K)} - \eta^{(K+1)} \right), \min_{j \in S_{\mathbf{v}}} \{ r_j - \theta_{\mathbf{v}} \} \right\}. \tag{1}$$

Definition 3 may look a bit cumbersome. The extra term $\min_{j \in S_{\mathbf{v}}} \{r_j - \theta_{\mathbf{v}}\}$ in (1) is added for a technical reason when handling the case that $|S_{\mathbf{v}}| < K$; we will discuss this in more detail in Remark 10. If $|S_{\mathbf{v}}| = K$, then the task of finding the best assortment is equivalent to the task of identifying the $K$ items with the largest advantage values $\eta_i$, and the reward gap in Definition 3 can be simplified as

$$\Delta_i = \begin{cases} \eta^{(K)} - \eta_i, & \forall i \in [N] \backslash S_{\mathbf{v}}, \\ \eta^{(K)} - \eta^{(K+1)}, & \forall i \in S_{\mathbf{v}}. \end{cases}$$

We now give two instance complexities for pure exploration in MNL-bandit. The second can be viewed as a refinement of the first.

**Definition 4** (instance complexity $H_1$)**.** *We define the first instance complexity for pure exploration in MNL-bandit to be*

$$H_1 \triangleq \sum\nolimits_{i \in [N]} \frac{1}{\Delta_i^2}.$$

In this paper we assume that $\forall i \in [N], \Delta_i \neq 0$, since otherwise the complexity $H_1$ will be infinity. This assumption implies that the best assortment is unique, which is also an essential assumption for works of literature whose pull complexities are based on "assortment-level" gaps.

Definition 4 bears some similarity to the instance complexity defined for pure exploration in the *multi-armed bandits* (MAB) model, where we have $N$ items each of which is associated with an unknown distribution, and the goal is to identify the item whose distribution has the largest mean. In MAB the instance complexity is defined to be $H_{\mathtt{MAB}} = \sum_{i=2}^{N} 1/\Delta_i^2$, where $\Delta_i = \mu^{(1)} - \mu^{(i)}$ where $\mu^{(1)}$ is the largest mean of the $N$ items and $\mu^{(i)}$ is the $i$-th largest mean of the $N$ items from work of Audibert, Bubeck, and Munos (2010). Our definition of $\Delta_i$ is more involved due to the more complicated combinatorial structure of the MNL-bandit model.

**Definition 5** (instance complexity $H_2$)**.**

$$H_2 \triangleq \sum\nolimits_{i \in [N]} \frac{v_i + 1/K}{\Delta_i^2} + \max_{i \in [N]} \frac{1}{\Delta_i^2}.$$

It is easy to see that $H_2 = O(H_1)$ (more precisely, $\frac{H_1}{K} \leq H_2 \leq 3H_1$). We comment that the $\max_{i \in [N]} \frac{1}{\Delta_i^2}$ term is needed only when $|S_{\mathbf{v}}| < K$.

**Our Results.** We propose two fixed-confidence algorithms for pure exploration in MNL-bandit. Algorithm 3 gives a pull complexity of $O\left(K^2 H_1 \ln\left(\frac{N}{\delta} \ln(K H_1)\right)\right)$ where $\delta$ is the confidence parameter. We then modify the algorithm using a more efficient preference exploration procedure at each pull, and improve the asymptotic pull complexity to $O\left(K^2 H_2 \ln\left(\frac{N}{\delta} \ln(K H_2)\right)\right)$. The second algorithm is presented in Algorithm 5.

Both algorithms can be implemented efficiently: the time complexity of Algorithm 3 is bounded by $\tilde{O}(T + N^2)$ where $T$ is the pull complexity and '$\tilde{O}()$' hides some logarithmic factors. That of Algorithm 5 is bounded by $\tilde{O}(TN + N^2)$.[1]

As we shall discuss in Remark 12, though having a larger pull complexity, Algorithm 3 still has the advantage that it better fits the *batched* model where we try to minimize the number of changes of the learning policy.

To complement our upper bounds, we prove that $\Omega(H_2/K^2)$ pulls is needed in order to identify the best assortment with probability at least 0.6. Note that when $K$ is a constant, our upper and lower bounds match up to a logarithmic factor.

## Related Work

Regret minimization in MNL-bandit was first studied by Rusmevichientong, Shen, and Shmoys (2010) in the setting of

[1]When we talk about *time complexity*, we only count the running time of the algorithm itself, and do not include the time for obtaining the pull results which depends on users' response time.

*dynamic assortment selection* under the MNL choice model. Since then there have been a number of follow-ups that further improve the regret bound and/or remove some artificial assumptions (Sauré and Zeevi 2013; Davis, Gallego, and Topaloglu 2013; Agrawal et al. 2016, 2017; Chen and Wang 2018). Agarwal, Johnson, and Agarwal (2020) studied *choice bandits* which can be seen as a generalization of MNL-bandit.

As mentioned previously, the algorithms by Rusmevichientong, Shen, and Shmoys (2010); Sauré and Zeevi (2013) also have a component of identifying the best assortment. Rusmevichientong, Shen, and Shmoys (2010); Sauré and Zeevi (2013) introduced the following "assortment-level" gap:

$$\Delta_{\mathtt{asso}} = \theta_{\mathbf{v}} - \max_{S\subseteq[N],|S|\le K,S\neq S_{\mathbf{v}}} R(S,\mathbf{v}),$$

that is, the difference between the reward of the best assortment and that of the second-best assortment. The pull complexity of the component in Sauré and Zeevi (2013) for finding the best assortment can be written as $\tilde{O}(KN/\Delta^2_{\mathtt{asso}})$, where'$\tilde{O}()$' hides some logarithmic factors. This result is better than that in Rusmevichientong, Shen, and Shmoys (2010). There are two critical differences between these results and our results: (1) More critically, in Rusmevichientong, Shen, and Shmoys (2010); Sauré and Zeevi (2013) it is assumed that the "assortment-level" gap $\Delta_{\mathtt{asso}}$ is known at the beginning, which is *not* practical since the fact that the preference vector is unknown at the beginning is a key feature of the MNL-bandit problem. (2) Our reward gaps $\Delta_i$ are defined at the "item-level"; the instance complexity $H_1$ (or $H_2$) is defined as the sum of the inverse square of these item-level gaps and the total pull complexity is $\tilde{O}(K^2H_1)$ (or $\tilde{O}(K^2H_2)$). Though the two complexities are not directly comparable, the following example shows that for certain input instances, our pull complexity is significantly better.

**Example 1.** *$K = 1, r_1 = \ldots = r_N = 1, v_1 = 1, v_2 = 1 - 1/\sqrt{N}, v_3 = \ldots = v_N = 1/\sqrt{N}$. We have $KN/\Delta_{asso} = \Omega(N^2)$, while $K^2H_1 = O(N)$. Thus, the pull complexity of the algorithm in Rusmevichientong, Shen, and Shmoys (2010) is* quadratic *of ours (up to logarithmic factors).*

The best assortment identification problem has also been studied in the static setting (Talluri and van Ryzin 2004; Désir, Goyal, and Zhang 2014), where the user preference vector $\mathbf{v}$ is known as a priori and our task is to conduct an offline computation to find the assortment that maximizes the expected reward. We refer readers to Kök and Fisher (2007) for an overview of this setting.

Chen, Li, and Mao (2018) studied the problem of top-$k$ ranking under the MNL choice model (but without the "no purchase" option). Their problem is different from ours: They aimed to find the $k$ items in $[N]$ with the largest preference $v_i$ (instead of the advantage $\eta_i = (r_i - \theta_{\mathbf{v}})v_i$). In some sense, their problem can be thought of as a special case of ours, where $r_1 = r_2 = \ldots = r_N$ (that is, the rewards of all items are the same); but in their model, there is *no* null item. It seems difficult to extend their approach to our setting. We would also like to mention the work on battling-bandits by Saha and Gopalan (2018), who considered the problem of using the MNL choice model as one of the natural models to draw a winner from a set of items. But their problem settings and the notion of the optimal solution are again different from the problem we consider here.

Pure exploration has been studied extensively in the model of MAB (Even-Dar, Mannor, and Mansour 2002; Mannor and Tsitsiklis 2004; Audibert, Bubeck, and Munos 2010; Gabillon et al. 2011; Gabillon, Ghavamzadeh, and Lazaric 2012; Karnin, Koren, and Somekh 2013; Jamieson et al. 2014; Kaufmann, Cappé, and Garivier 2016; Garivier and Kaufmann 2016; Russo 2016; Chen, Li, and Qiao 2017). MNL-bandit can be viewed as an MAB-type model with $\sum_{j\in[K]}\binom{N}{j}$ items, each corresponding to an assortment $S \subseteq [N]$ with $|S| \le K$. However, these items may "intersect" with each other since assortments may contain the same items. Due to such dependencies, the algorithms designed for pure exploration in the MAB model cannot be adopted to the MNL-bandit model. Audibert, Bubeck, and Munos (2010) designed an instance-sensitive algorithm for the pure exploration problem in the MAB model. The result of Audibert, Bubeck, and Munos (2010) was later improved by Karnin, Koren, and Somekh (2013) and Chen, Li, and Qiao (2017), and extended into the problem of identifying multiple items (Bubeck, Wang, and Viswanathan 2013; Zhou, Chen, and Li 2014; Chen et al. 2017).

Finally, we note that recently, concurrent and independent of our work, Yang (2021) has also studied pure exploration in MNL-bandit. But the definition of instance complexity in Yang (2021) is again at the "assortment-level" (and thus the results are not directly comparable), and the algorithmic approaches in Yang (2021) are also different from ours. The pull complexity of Yang (2021) can be written as $\tilde{O}(H_{yang})$ where $H_{yang} = \sum_{i\in[N]}\frac{1}{(\Delta'_i)^2}$, where $\Delta'_i$ is defined to be the difference between the best reward among assortments that include item $i$ and that among assortments that exclude item $i$. The following example shows that for certain input instances, our pull complexity based on item-level gaps is better.

**Example 2.** *$r_1 = \ldots = r_N = 1, v_1 = \ldots = v_K = 1, v_{K+1} = \ldots = v_N = \epsilon$. For $\epsilon \in (0, 1/K)$ and $\omega(1) \le K \le o(N)$, we have $H_{yang} = \Theta(NK^4)$, while our $K^2H_2 = \Theta(K^5 + NK^3) = o(H_{yang})$.*

## Preliminaries

Before presenting our algorithms, we would like to introduce some tools in probability theory and give some basic properties of the MNL-bandit model. Due to space constraints, we leave the tools in probability theory (including Hoeffding's inequality, concentration results for the sum of geometric random variables, etc.) to the full version of this paper.

The following (folklore) observation gives an effective way to check whether the expected reward of $S$ with respect to $\mathbf{v}$ is at least $\theta$ for a given value $\theta$. The proof can be found in the full version of this paper.

**Observation 1.** *For any $\theta \in [0, 1]$, $R(S, \mathbf{v}) \ge \theta$ if and only if $\sum_{i\in S}(r_i - \theta)v_i \ge \theta$.*

With Observation 1, to check whether the maximum expected reward is at least $\theta$ for a given value $\theta$, we only need

**Algorithm 1:** EXPLORE($i$)

```
Input: Item $i$.
Output: $0/1$ (choose or not choose $i$).
1 Offer a singleton set $S_i \leftarrow \{i\}$ and observe a feedback $a$;
2 if $a = 0$ then return 1;
3 return 0
```

to check whether the expected reward of the particular set $S \subseteq [N]$ containing the up to $K$ items with the largest *positive* values $(r_i - \theta)v_i$ is at least $\theta$.

To facilitate the future discussion we introduce the following definition.

**Definition 6** (Top$(I, \mathbf{v}, \theta)$)**.** *Given a set of items I where the i-th item has reward $r_i$ and preference $v_i$, and a value $\theta$, let $T$ be the set of* $\min\{K, |I|\}$ *items with the largest values $(r_i - \theta)v_i$. Define* $\mathtt{Top}(I, \mathbf{v}, \theta) \triangleq T \setminus \{i \in I \mid (r_i - \theta) \le 0\}$, *where* $\mathbf{v}$ *stands for* $(v_1, \dots, v_{|I|})$.

The following lemma shows that $\mathtt{Top}(I, \mathbf{v}, \theta_{\mathbf{v}})$ is exactly the best assortment. Its proof can be found in the full version of this paper.

**Lemma 2.** $\mathtt{Top}(I, \mathbf{v}, \theta_{\mathbf{v}}) = S_{\mathbf{v}}$.

Note that the set $\mathtt{Top}(I, \mathbf{v}, \theta_{\mathbf{v}})$ is unique by its definition. Therefore by Lemma 2 the set $S_{\mathbf{v}}$ is also uniquely defined.

We next show a monotonicity property of the expected reward function $R(\cdot, \cdot)$. Given two vectors $\mathbf{v}, \mathbf{w}$ of the same dimension, we write $\mathbf{v} \preceq \mathbf{w}$ if $\forall i, v_i \le w_i$. We comment that similar properties appeared in Agrawal et al. (2016, 2017), but were formulated a bit differently from ours. The proof of Lemma 3 can be found in the full version of this paper.

**Lemma 3.** *If* $\mathbf{v} \preceq \mathbf{w}$, *then* $(\theta_{\mathbf{v}} =)R(S_{\mathbf{v}}, \mathbf{v}) \le (\theta_{\mathbf{w}} =)R(S_{\mathbf{w}}, \mathbf{w})$, *and for any $S \subseteq I$ it holds that*

$$R(S, \mathbf{w}) - R(S, \mathbf{v}) \le \sum_{i \in S} (w_i - v_i).$$

The following is an immediate corollary of Lemma 3.

**Corollary 4.** *If* $\forall i : v_i \le w_i \le v_i + \frac{\epsilon}{K}$, *then* $\theta_{\mathbf{v}} \le \theta_{\mathbf{w}} \le \theta_{\mathbf{v}} + \epsilon$.

## The Basic Algorithm

In this section, we present our first algorithm for pure exploration in MNL-bandit. The main algorithm is described in Algorithm 3, which calls PRUNE (Algorithm 2) and EXPLORE (Algorithm 1) as subroutines. EXPLORE describes a pull of the assortment consisting of a single item.

Let us describe the Algorithm 2 and 3 in more detail. Algorithm 3 proceeds in rounds. In round $\tau$, each "surviving" item in the set $I_\tau$ has been pulled by $T_\tau$ times in total. We try to construct two vectors $\mathbf{a}$ and $\mathbf{b}$ based on the empirical means of the items in $I_\tau$ such that the (unknown) true preference vector $\mathbf{v}$ of $I_\tau$ is tightly sandwiched by $\mathbf{a}$ and $\mathbf{b}$ (Line 7-8). We then feed $I_\tau$, $\mathbf{a}$, and $\mathbf{b}$ to the PRUNE subroutine which reduces the size of $I_\tau$ by removing items that have no chance to be included in the best assortment (Line 9). Finally, we test whether the output of PRUNE is indeed the best assortment (Line 10). If not we proceed to the next round, otherwise we return the solution.

**Algorithm 2:** PRUNE($I, K, \mathbf{a}, \mathbf{b}$)

```
Input: a set of items $I = \{1, \dots, N\}$, capacity
       parameter $K$, two vectors
       $\mathbf{a} = (a_1, \dots, a_N), \mathbf{b} = (b_1, \dots, b_N)$ such that
       for any $i \in [N]$ it holds that $a_i \le v_i \le b_i$,
       where $\mathbf{v} = (v_1, \dots, v_N)$ is the (unknown)
       preference vector of the $N$ items.
Output: a set of candidate items for constructing the
        best assortment.
1 $\theta_{\mathbf{a}} \leftarrow \max_{S \subseteq I : |S| \le K} R(S, \mathbf{a}), \theta_{\mathbf{b}} \leftarrow \max_{S \subseteq I : |S| \le K} R(S, \mathbf{b})$;
2 $C \leftarrow \emptyset$;
3 foreach $i \in I$ do
4     form a vector $\mathbf{g} = (g_1, \dots, g_N)$ s.t. $g_j = a_j$ for
      $j \ne i$, and $g_i = b_i$;
5     if $\exists \theta \in [\theta_{\mathbf{a}}, \theta_{\mathbf{b}}]$ s.t. $i \in \mathtt{Top}(I, \mathbf{g}, \theta)$ then add $i$ to
      $C$;
6 return $C$
```

Now we turn to the PRUNE subroutine (Algorithm 2), which is the most interesting part of the algorithm. Recall that the two vectors $\mathbf{a}$ and $\mathbf{b}$ are constructed such that $\mathbf{a} \preceq \mathbf{v} \preceq \mathbf{b}$. We try to prune items in $I$ by the following test: For each $i \in I$, we form another vector $\mathbf{g}$ such that $\mathbf{g} = \mathbf{a}$ in all coordinates except the $i$-th coordinate where $g_i = b_i$ (Line 4). We then check whether there exists a value $\theta \in [\theta_{\mathbf{a}}, \theta_{\mathbf{b}}]$ such that $i \in \mathtt{Top}(I, \mathbf{g}, \theta)$, where $\theta_{\mathbf{a}}, \theta_{\mathbf{b}}$ are the maximum expected rewards with $\mathbf{a}$ and $\mathbf{b}$ as the item preference vectors respectively; if the answer is Yes then item $i$ survives, otherwise it is pruned (Line 5). Note that our test is fairly conservative: we try to put item $i$ in a more favorable position by using the upper bound $b_i$ as its preference, while for other items we use the lower bounds $a_j$ as their preferences. Such a conservative pruning step makes sure that the output $C$ of the PRUNE subroutine is always a superset of the best assortment $S_{\mathbf{v}}$.

**Theorem 5.** *For any confidence parameter $\delta > 0$, Algorithm 3 returns the best assortment with probability $(1 - \delta)$ using at most* $\Gamma = O\left(K^2 H_1 \ln\left(\frac{N}{\delta} \ln(K H_1)\right)\right)$ *pulls. The running time of Algorithm 3 is bounded by* $O\left(N\Gamma + N^2 \ln N \ln\left(\frac{K}{\min_{i \in I} \Delta_i}\right)\right)$.

In the rest of this section we prove Theorem 5.

**Correctness.** We start by introducing the following event which we will condition on in the rest of the proof. The event states that in any round $\tau$, the estimated preference $v_i^{(\tau)}$ for each item $i$ (computed at Line 7 of Algorithm 3) is at most $\epsilon_\tau = 2^{-\tau-3}$ away from the true preference $v_i$.

$$\mathcal{E}_1 \triangleq \{\forall \tau \ge 0, \forall i \in I_\tau : \left|v_i^{(\tau)} - v_i\right| < \epsilon_\tau\}.$$

The proof of the following lemma can be found in full version of this paper. This lemma states that event $\mathcal{E}_1$ holds with high probability.

**Algorithm 3:** The Fixed Confidence Algorithm for MNL-Bandit

```
Input: a set of items $I = \{1, \ldots, N\}$, a capacity
       parameter $K$, a confidence parameter $\delta$.
Output: the best assortment.
1  $I_0 \leftarrow I$, set $\epsilon_\tau = 2^{-\tau-3}$ for $\tau \geq 0$;
2  set $T_{-1} \leftarrow 0$ and $T_\tau \leftarrow \left\lceil \frac{32}{\epsilon_\tau^2} \ln \frac{16N(\tau+1)^2}{\delta} \right\rceil$ for $\tau \geq 0$;
3  for $\tau = 0, 1, \ldots$ do
4      foreach $i \in I_\tau$ do call EXPLORE$(i)$ for
         $(T_\tau - T_{\tau-1})$ times;
5      let $x_i^{(\tau)}$ be the mean of the outputs of the $T_\tau$ calls
         of EXPLORE$(i)$;
6      foreach $i \in I_\tau$ do
7          set $v_i^{(\tau)} \leftarrow \min\{\frac{1}{x_i^{(\tau)}} - 1, 1\}$,
           $a_i^{(\tau)} \leftarrow \max\{v_i^{(\tau)} - \epsilon_\tau, 0\}$, and
           $b_i^{(\tau)} \leftarrow \min\{v_i^{(\tau)} + \epsilon_\tau, 1\}$;
8      let $\mathbf{a}^{(\tau)}$ be the vector containing the $|I_\tau|$ estimated
         preferences $a_i^{(\tau)}$, and $\mathbf{b}^{(\tau)}$ be the vector
         containing the $|I_\tau|$ estimated preferences $b_i^{(\tau)}$;
9      $C \leftarrow$ PRUNE$(I_\tau, \mathbf{a}^{(\tau)}, \mathbf{b}^{(\tau)})$;
10     if $(|C| \leq K) \wedge \left( \bigwedge_{i \in C} \left(r_i > R\left(C, \mathbf{b}^{(\tau)}\right)\right) \right)$ then
11         return $C$ ;
12     $I_{\tau+1} \leftarrow C$;
```

**Lemma 6.** $\Pr[\mathcal{E}_1] \geq 1 - \delta$.

It is easy to see from Line 7 of Algorithm 3 that conditioned on $\mathcal{E}_1$, we have

$$\forall \tau \geq 0 : \mathbf{a}^{(\tau)} \preceq \mathbf{v}^{(\tau)} \preceq \mathbf{b}^{(\tau)}, \tag{2}$$

where $\mathbf{v}^{(\tau)}$ is the preference vector of items in $I_\tau$.

The following lemma shows that if (2) holds, then the PRUNE subroutine (Algorithm 2) always produces a set of candidate items $C$ which is a superset of the best assortment.

**Lemma 7.** *If the preference vector* $\mathbf{v}$ *of* $I$ *satisfies* $\mathbf{a} \preceq \mathbf{v} \preceq \mathbf{b}$*, then* PRUNE$(I, K, \mathbf{a}, \mathbf{b})$ *(Algorithm 2) returns a set* $C$ *such that* $S_\mathbf{v} \subseteq C$.

*Proof.* First, if $\mathbf{a} \preceq \mathbf{v} \preceq \mathbf{b}$, then by Lemma 3 we have $\theta_\mathbf{v} \in [\theta_\mathbf{a}, \theta_\mathbf{b}]$.

Consider any item $i \in S_\mathbf{v}$, by the construction of $\mathbf{g}$ (Line 4 of Algorithm 2) we have for every $j \in I$:

- if $j \neq i$, then $(r_j - \theta_\mathbf{v})g_j \leq \max\{(r_j - \theta_\mathbf{v})v_j, 0\}$;
- if $j = i$, then $(r_j - \theta_\mathbf{v})g_j \geq (r_j - \theta_\mathbf{v})v_j$.

By these two facts and the definition of $\texttt{Top}(I, \mathbf{v}, \theta_\mathbf{v})$, we know that if $i \in \texttt{Top}(I, \mathbf{v}, \theta_\mathbf{v})$, then $i \in \texttt{Top}(I, \mathbf{g}, \theta_\mathbf{v})$. Therefore for the particular value $\theta = \theta_\mathbf{v} \in [\theta_\mathbf{a}, \theta_\mathbf{b}]$ we have $i \in \texttt{Top}(I, \mathbf{g}, \theta)$, and consequently $i$ will be added to the candidate set $C$ at Line 5, implying that $S_\mathbf{v} \subseteq C$. □

Now suppose Algorithm 3 stops after round $\tau$ and outputs a set $C \supseteq S_\mathbf{v}$ of size at most $K$ (Line 10-11), then for any $i \in C$, we have $r_i > \theta_\mathbf{b}$. By Lemma 3 we also have $\theta_\mathbf{b} \geq \theta_\mathbf{v}$ (since $\mathbf{v} \preceq \mathbf{b}$). We thus have $r_i > \theta_\mathbf{v}$. Consequently, it holds that for every $i \in C$, $(r_i - \theta_\mathbf{v}) > 0$. We thus have $C = S_\mathbf{v}$.

Up to this point we have shown that conditioned on $\mathcal{E}_1$, if Algorithm 3 stops, then it outputs the best assortment $S_\mathbf{v}$. We next bound the number of pulls the algorithm uses.

**Pull Complexity.** We again conditioned on event $\mathcal{E}_1$. The next lemma essentially states that an item $i \in I \backslash S_\mathbf{v}$ will be pruned if its reward gap $\Delta_i$ is much larger than $K$ times its preference estimation error $\max\{b_i - v_i, v_i - a_i\}$.

**Lemma 8.** *In* PRUNE$(I, K, \mathbf{a}, \mathbf{b})$ *(Algorithm 2), if* $\mathbf{a} \preceq \mathbf{v} \preceq \mathbf{b}$*, and* $\forall i \in I : \max\{b_i - v_i, v_i - a_i\} \leq \epsilon/K$ *for any* $\epsilon \in (0, 1)$*, then any item* $i \in I \backslash S_\mathbf{v}$ *satisfying* $\Delta_i > 8\epsilon$ *will not be added to set* $C$.

*Proof.* By Corollary 4, if $\mathbf{a} \preceq \mathbf{v} \preceq \mathbf{b}$, and $\forall i \in I : \max\{b_i - v_i, v_i - a_i\} \leq \epsilon/K$, then we have

$$\theta_\mathbf{v} - \epsilon \leq \theta_\mathbf{a} \leq \theta_\mathbf{v} \leq \theta_\mathbf{b} \leq \theta_\mathbf{v} + \epsilon. \tag{3}$$

Consider any item $i \in I \backslash S_\mathbf{v}$ with $\Delta_i > 8\epsilon$. We analyze in two cases.

**Case 1:** $\theta_\mathbf{v} - r_i > 8\epsilon$. By (3) we have $\theta_\mathbf{a} - r_i > 7\epsilon$. Therefore, for any $\theta \in [\theta_\mathbf{a}, \theta_\mathbf{b}]$ we have $r_i < \theta_\mathbf{a} \leq \theta$, and consequently $i \notin \texttt{Top}(I, \mathbf{g}, \theta)$ for any $\theta \in [\theta_\mathbf{a}, \theta_\mathbf{b}]$ by the definition of $\texttt{Top}()$.

**Case 2:** $\theta_\mathbf{v} - r_i \leq 8\epsilon$. First, note that if $|S_\mathbf{v}| < K$, then we have

$$\Delta_i = -(r_i - \theta_\mathbf{v})v_i = (\theta_\mathbf{v} - r_i)v_i \leq \theta_\mathbf{v} - r_i \leq 8\epsilon,$$

contradicting our assumption that $\Delta_i > 8\epsilon$. We thus focus on the case that $|S_\mathbf{v}| = K$. We analyze two subcases.

1. $\theta \in (r_i, 1]$. In this case, by the definition of $\texttt{Top}()$ and the fact that $r_i - \theta < 0$, we have $i \notin \texttt{Top}(I, \mathbf{g}, \theta)$.
2. $\theta \in [\theta_\mathbf{a}, \theta_\mathbf{b}] \cap [0, r_i]$. For any $j \in S_\mathbf{v}$, we have

$$\begin{aligned}
& (r_i - \theta)g_i - (r_j - \theta)g_j \\
= \; & (r_i - \theta)b_i - (r_j - \theta)a_j \\
\leq \; & (r_i - \theta)(v_i + \epsilon) - (r_j - \theta)a_j \\
\leq \; & (r_i - \theta)v_i - (r_j - \theta)a_j + \epsilon \\
\leq \; & (r_i - \theta_\mathbf{v})v_i - (r_j - \theta_\mathbf{v})a_j + (1 + a_j + v_i)\epsilon \\
\leq \; & (r_i - \theta_\mathbf{v})v_i - (r_j - \theta_\mathbf{v})(v_j - \epsilon) + 3\epsilon \\
\leq \; & (r_i - \theta_\mathbf{v})v_i - (r_j - \theta_\mathbf{v})v_j + 4\epsilon \\
\leq \; & -\Delta_i + 4\epsilon \\
< \; & -4\epsilon. \quad \text{(by the assumption } \Delta_i > 8\epsilon)
\end{aligned}$$

We thus have that for any $\theta \in [\theta_\mathbf{a}, \theta_\mathbf{b}] \cap [0, r_i]$, $(r_i - \theta)g_i < (r_j - \theta)g_j$ for any $j \in S_\mathbf{v}$, therefore $i \notin \texttt{Top}(I, \mathbf{g}, \theta)$ for any $\theta \in [\theta_\mathbf{a}, \theta_\mathbf{b}]$, and consequently $i \notin C$.

□

For any $i \in I$, we define

$$\tau(i) \triangleq \min\left\{\tau \geq 0 : \epsilon_\tau \leq \frac{\Delta_i}{32K}\right\}. \tag{4}$$

The next lemma shows that item $i$ will *not* appear in any set $I_\tau$ with $\tau > \tau(i)$, and thus will not be pulled further after round $\tau(i)$.

**Lemma 9.** *In Algorithm 3, for any item $i \in I$, we have $i \notin I_\tau$ for any $\tau > \tau(i)$.*

*Proof.* For any $i \in I \backslash S_{\mathbf{v}}$, setting $\epsilon = \Delta_i/16$. By (4) we have that for any $j \in I_{\tau(i)}$ it holds that

$$\max\left\{v_j - a_j^{(\tau(i))}, b_j^{(\tau(i))} - v_j\right\} \le \frac{\Delta_i}{16K} = \frac{\epsilon}{K}. \qquad (5)$$

Moreover, we have,

$$\Delta_i = 16\epsilon > 8\epsilon. \qquad (6)$$

By (5), (6) and Lemma 8, we have $i \notin I_{\tau(i)+1}$.

We next consider items in $S_{\mathbf{v}}$. Note that by Definition 3, all $i \in S_{\mathbf{v}}$ have the same reward gap:

$$\Delta_i = \min\{\min_{j \in I\backslash S_{\mathbf{v}}}\{\Delta_j\}, \min_{j\in S_{\mathbf{v}}}\{r_j - \theta_{\mathbf{v}}\}\} \le \min_{j \in I\backslash S_{\mathbf{v}}}\{\Delta_j\}.$$

Let

$$\bar{\tau} \triangleq \min\left\{\tau \ge 0 : \epsilon_\tau \le \frac{\bar{\Delta}}{32K}\right\}. \qquad (7)$$

We thus have $\bar{\tau} = \tau(i)$ for all $i \in S_{\mathbf{v}}$, and $\bar{\tau} \ge \tau(j)$ for any $j \in I\backslash S_{\mathbf{v}}$. Therefore, at the end of round $\bar{\tau}$, all items in $I\backslash S_{\mathbf{v}}$ have already been pruned, and consequently,

$$|C| \le K. \qquad (8)$$

By (4) and Corollary 4 we have $\theta_{\mathbf{b}^{(\bar{\tau})}} \le \theta_{\mathbf{v}} + \bar{\Delta}/16$. Consequently we have

$$\begin{aligned} r_i - R(C, \mathbf{b}^{(\bar{\tau})}) &= (r_i - \theta_{\mathbf{v}}) - (\theta_{\mathbf{b}^{(\bar{\tau})}} - \theta_{\mathbf{v}}) \\ &\ge \bar{\Delta} - \frac{\bar{\Delta}}{16} > 0\,. \end{aligned} \qquad (9)$$

By (8) and (9), we know that Algorithm 3 will stop after round $\bar{\tau}$ and return $C = S_{\mathbf{v}}$. □

With Lemma 9 we can easily bound the total number of pulls made by Algorithm 3. By (4) we have $\tau(i) = O\left(\ln\left(\frac{K}{\Delta_i}\right)\right)$. By the definition of $T_\tau$ (Line 2 of Algorithm 3), the total number of pulls is at most

$$\begin{aligned} \sum_{i\in I} T_{\tau(i)} &\le O\left(\sum_{i \in I}\frac{K^2}{\Delta_i^2}\ln\frac{N\tau^2(i)}{\delta}\right) \\ &= O\left(K^2 H_1 \ln\left(\frac{N}{\delta}\ln(KH_1)\right)\right). \end{aligned}$$

**Remark 10.** *The reason that we introduce an extra term $\min_{j\in S_{\mathbf{v}}}\{r_j - \theta_{\mathbf{v}}\}$ in the definition of reward gap $\Delta_i$ for all $i \in S_{\mathbf{v}}$ (Definition 3) is for handling the case when $|S_{\mathbf{v}}| < K$. More precisely, in the case $|S_{\mathbf{v}}| < K$ we have to make sure that for all items $i \in I$ that we are going to add into the best assortment $S_{\mathbf{v}}$, it holds that $r_i > \theta_{\mathbf{v}}$. In our proof this is guaranteed by (9). On the other hand, if we are given the promise that $|S_{\mathbf{v}}| = K$ (or $|S_{\mathbf{v}}| = K'$ for a fixed value $K' \le K$), then we do not need this extra term: we know when to stop simply by monitoring the size of $I_\tau$, since at the end all items $i \in I/S_{\mathbf{v}}$ will be pruned.*

**Running Time.** Finally, we analyze the time complexity of Algorithm 3. Although the time complexity of the algorithm is not the first consideration in the MNL-bandit model, we believe it is important for the algorithm to finish in a reasonable amount of time for real-time decision making. Observe that the running time of Algorithm 3 is dominated by the sum of the total number of pulls and the running time of the PRUNE subroutine, which is the main object that we shall bound next.

Let us analyze the running time of PRUNE. Let $n \triangleq |I|$. First, $\theta_{\mathbf{a}}$ and $\theta_{\mathbf{b}}$ can be computed in $O(n^2)$ time by an algorithm proposed by Rusmevichientong, Shen, and Shmoys (2010). We next show that Line 5 of Algorithm 2 can be implemented in $O(n \ln n)$ time, with which the total running time of PRUNE is bounded by $O(n^2 \ln n)$.

Consider any item $i \in I$. We can restrict our search of possible $\theta$ in the range of $\Theta_i = [\theta_{\mathbf{a}}, \theta_{\mathbf{b}}] \cap [0, r_i)$, since if $i \in \mathtt{Top}(I, \mathbf{g}, \theta)$, then by the definition of $\mathtt{Top}()$ we have $\theta < r_i$. For each $j \ne i, j \in I$, define

$$\Theta_j = \{\theta \in \Theta_i \mid (r_j - \theta)g_j > (r_i - \theta)g_i\}.$$

Intuitively speaking, $\Theta_j$ contains all $\theta$ values for which item $j$ is "preferred to" item $i$ for $\mathtt{Top}(I, \mathbf{g}, \theta)$. Consequently, for any $\theta \in \Theta_i$, if the number of $\Theta_j$ that contain $\theta$ is at least $K$, then we have $i \notin \mathtt{Top}(I, \mathbf{g}, \theta)$; otherwise if the number of such $\Theta_j$ is less than $K$, then we have $i \in \mathtt{Top}(I, \mathbf{g}, \theta)$. Note that each set $\Theta_j$ can be computed in $O(1)$ time.

Now think each set $\Theta_j$ as an interval. The problem of testing whether there exists a $\theta \in [\theta_{\mathbf{a}}, \theta_{\mathbf{b}}] \cap [0, r_i)$ such that $i \in \mathtt{Top}(I, \mathbf{g}, \theta)$ can be reduced to the problem of checking whether there is a $\theta \in [\theta_{\mathbf{a}}, \theta_{\mathbf{b}}] \cap [0, r_i)$ such that $\theta$ is contained in fewer than $K$ intervals $\Theta_j$ $(j \ne i)$. The later problem can be solved by the standard sweep line algorithm in $O(n \ln n)$ time.

Recall that the total number of rounds can be bounded by $\tau_{\max} = \max_{i \in I}\tau(i) = O\left(\ln\left(\frac{K}{\min_{i\in I}\Delta_i}\right)\right)$. Therefore the total running time of Algorithm 3 can be bounded by

$$O\left(\Gamma + N^2 \ln N \ln\left(\frac{K}{\min_{i \in I}\Delta_i}\right)\right),$$

where $\Gamma = O\left(K^2 H_1 \ln\left(\frac{N}{\delta}\ln(KH_1)\right)\right)$ is the total number of pulls made by the algorithm.

## The Improved Algorithm

In this section we try to improve our basic algorithm presented in Section 3. We design an algorithm whose pull complexity depends on $H_2$ which is asymptotically at most $H_1$. The improved algorithm is described in Algorithm 5.

The structure of Algorithm 5 is very similar to that of Algorithm 3. The main difference is that instead of using EXPLORE to pull a singleton assortment at each time, we use a new procedure EXPLORESET (Algorithm 4) which pulls an assortment of size up to $K$ (Line 6 of Algorithm 5). We construct the assortments by partitioning the whole set of items $I_\tau$ into subsets of size up to $K$ (Line 4-5). In the EXPLORESET procedure, we keep pulling the assortment $S$ until the output is 0 (i.e., a no-purchase decision is made). We then estimate the preference of item $i$ using the average

**Algorithm 4:** ExploreSet($S$)

```
    Input: a set of items $S$ of size at most $K$.
    Output: a set of empirical preferences $\{f_i\}_{i \in S}$.
1   Initialize $f_i \leftarrow 0$ for $i \in S$;
2   repeat
3       offer assortment $S$ and observe a feedback $a$;
4       if $a \in S$ then $f_a \leftarrow f_a + 1$ ;
5   until $a = 0$;
6   return $\{f_i\}_{i \in S}$
```

number of times that item $i$ is chosen in those ExploreSet calls that involve item $i$ (Line 8).

Intuitively, ExploreSet has the advantage over Explore in that at each pull, the probability for ExploreSet to return an item instead of a no-purchase decision is higher, and consequently ExploreSet extracts more information about the item preferences. We note that the ExploreSet procedure was first introduced in Agrawal et al. (2019) in the setting of regret minimization.

**Theorem 11.** *For any confidence parameter $\delta > 0$, Algorithm 5 returns the best assortment with probability $(1 - \delta)$ using at most $\Gamma = O\left(K^2 H_2 \ln\left(\frac{N}{\delta}\ln(KH_2)\right)\right)$ pulls. The running time of Algorithm 5 is bounded by $O\left(N\Gamma + N^2 \ln N \ln\left(\frac{K}{\min_{i \in I}\Delta_i}\right)\right)$.*

Compared with Theorem 5, the only difference in the pull complexity of Theorem 11 is that we have used $H_2$ instead of $H_1$. Since $H_2 = O(H_1)$, the asymptotic pull complexity of Algorithm 5 is at least as good as that of Algorithm 3.

**Remark 12.** *Though having a higher pull complexity, Algorithm 3 still has an advantage against Algorithm 5 in that Algorithm 3 can be implemented in the batched setting with $\max_{i \in I} \tau(i) = O\left(\ln \frac{K}{\min_{i \in I}\Delta_i}\right)$ policy changes, which cannot be achieved by Algorithm 5 since the subroutine* ExploreSet *is inherently sequential.*

Compared with the proof for Theorem 5, the challenge for proving Theorem 11 is that the number of pulls in each ExploreSet is a random variable. We thus need slightly more sophisticated mathematical tools to bound the sum of these random variables. Due to the space constraints, we leave the technical proof of Theorem 11 to the full version of this paper.

## Lower Bound

We manage to show the following lower bound to complement our algorithmic results.

**Theorem 13.** *For any algorithm $\mathcal{A}$ for pure exploration in multinomial logit bandit, there exists an input instance such that $\mathcal{A}$ needs $\Omega(H_2/K^2)$ pulls to identify the best assortment with probability at least $0.6$.*

Note that Algorithm 5 identifies the best assortment with probability 0.99 using at most $\tilde{O}(K^2 H_2)$ pulls (setting $\delta = 0.01$). Therefore our upper and lower bounds match up to a logarithmic factor if $K = O(1)$.

**Algorithm 5:** Improved Fixed Confidence Algorithm for MNL-bandit

```
    Input: a set of items $I = \{1, \ldots, N\}$, a capacity
           parameter $K$, and a confidence parameter $\delta$.
    Output: the best assortment.
1   set $I_0 \leftarrow I$, and $\epsilon_\tau = 2^{-\tau-3}$ for $\tau \geq 0$;
2   set $T_{-1} \leftarrow 0$, and $T_\tau \leftarrow \left\lceil \frac{8}{\epsilon_\tau^2} \ln \frac{16N(\tau+1)^2}{\delta} \right\rceil$ for
    $\tau \geq 0$;
3   for $\tau = 0, 1, \ldots$ do
4       $m_\tau \leftarrow \lceil |I_\tau| / K \rceil$;
5       let $S_1^\tau \uplus \ldots \uplus S_{m_\tau}^\tau$ be an arbitrary partition of $I_\tau$
          into subsets of size at most $K$;
6       foreach $j \in [m_\tau]$ do call ExploreSet($S_j^\tau$) for
          $(T_\tau - T_{\tau-1})$ times ;
7       foreach $i \in I_\tau$ do
8           let $v_i^{(\tau)}$ be the average of $f_i$'s returned by the
              multiset of calls $\{$ExploreSet$(S_j^\rho) \mid \rho \leq$
              $\tau,\ j \in [m_\rho], i \in S_j^\rho\}$;
9       foreach $i \in I_\tau$ do set $a_i^{(\tau)} \leftarrow \max\{0, v_i^{(\tau)} - \epsilon_\tau\}$
          and $b_i^{(\tau)} \leftarrow \min\{v_i^{(\tau)} + \epsilon_\tau, 1\}$ ;
10      let $\mathbf{a}^{(\tau)}$ be the vector containing the $|I_\tau|$ estimated
          preferences $a_i^{(\tau)}$, and $\mathbf{b}^{(\tau)}$ be the vector
          containing the $|I_\tau|$ estimated preferences $b_i^{(\tau)}$;
11      $C \leftarrow$ Prune$(I_\tau, \mathbf{a}^{(\tau)}, \mathbf{b}^{(\tau)})$;
12      if $(|C| \leq K) \wedge \left(\bigwedge_{i \in C} \left(r_i > R\left(C, \mathbf{b}^{(\tau)}\right)\right)\right)$ then
13          return $C$ ;
14      $I_{\tau+1} \leftarrow C$ ;
```

The proof of Theorem 13 bears some similarity with the lower bound proof of Chen, Li, and Mao (2018), but there are some notable differences. As mentioned in the introduction, Chen, Li, and Mao (2018) considered the problem of top-$k$ ranking under the MNL choice model, which differs from the best assortment searching problem in the following aspects:

1. The top-$k$ ranking problem can be thought as a special case of the best assortment searching problem where the rewards of all items are equal to 1. While to prove Theorem 13 we need to choose hard instances in which items have *different* rewards.

2. There is *no* null item (i.e., the option of "no purchase") in the top-$k$ ranking problem. Note that we cannot treat the null item as the $(N + 1)$-th item with reward 0 since the null item will appear implicitly in every selected assortment.

These two aspects prevent us to use the lower bound result in Chen, Li, and Mao (2018) as a blackbox. Due to the space constraints, we leave the technical proof to the full version of this paper.

## Acknowledgments

Nikolai Karpov and Qin Zhang are supported in part by NSF CCF-1844234 and CCF-2006591.

## References

Agarwal, A.; Johnson, N.; and Agarwal, S. 2020. Choice Bandits. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *NeurIPS*.

Agrawal, S.; Avadhanula, V.; Goyal, V.; and Zeevi, A. 2016. A Near-Optimal Exploration-Exploitation Approach for Assortment Selection. In *EC*, 599–600.

Agrawal, S.; Avadhanula, V.; Goyal, V.; and Zeevi, A. 2017. Thompson Sampling for the MNL-Bandit. In *COLT*, 76–78.

Agrawal, S.; Avadhanula, V.; Goyal, V.; and Zeevi, A. 2019. MNL-bandit: A dynamic learning approach to assortment selection. *Operations Research*, 67(5): 1453–1485.

Audibert, J.; Bubeck, S.; and Munos, R. 2010. Best Arm Identification in Multi-Armed Bandits. In *COLT*, 41–53.

Avadhanula, V. 2019. *The MNL-Bandit Problem: Theory and Applications*. Ph.D. thesis, Columbia University.

Bubeck, S.; Wang, T.; and Viswanathan, N. 2013. Multiple Identifications in Multi-Armed Bandits. In *ICML*, 258–265.

Chen, J.; Chen, X.; Zhang, Q.; and Zhou, Y. 2017. Adaptive multiple-arm identification. In *ICML*, 722–730.

Chen, L.; Li, J.; and Qiao, M. 2017. Towards Instance Optimal Bounds for Best Arm Identification. In *COLT*, volume 65, 535–592.

Chen, X.; Li, Y.; and Mao, J. 2018. A Nearly Instance Optimal Algorithm for Top-$k$ Ranking under the Multinomial Logit Model. In Czumaj, A., ed., *SODA*, 2504–2522.

Chen, X.; and Wang, Y. 2018. A note on a tight lower bound for capacitated MNL-bandit assortment selection models. *Oper. Res. Lett.*, 46(5): 534–537.

Davis, J.; Gallego, G.; and Topaloglu, H. 2013. Assortment planning under the multinomial logit model with totally unimodular constraint structures. *Technical Report*.

Désir, A.; Goyal, V.; and Zhang, J. 2014. Near-optimal algorithms for capacity constrained assortment optimization. *Available at SSRN 2543309*.

Even-Dar, E.; Mannor, S.; and Mansour, Y. 2002. PAC Bounds for Multi-armed Bandit and Markov Decision Processes. In *COLT*, 255–270.

Gabillon, V.; Ghavamzadeh, M.; and Lazaric, A. 2012. Best Arm Identification: A Unified Approach to Fixed Budget and Fixed Confidence. In *NIPS*, 3221–3229.

Gabillon, V.; Ghavamzadeh, M.; Lazaric, A.; and Bubeck, S. 2011. Multi-Bandit Best Arm Identification. In *NIPS*, 2222–2230.

Garivier, A.; and Kaufmann, E. 2016. Optimal Best Arm Identification with Fixed Confidence. In *COLT*, 998–1027.

Jamieson, K.; Malloy, M.; Nowak, R.; and Bubeck, S. 2014. lil'ucb: An optimal exploration algorithm for multi-armed bandits. In *COLT*, 423–439.

Janson, S. 2018. Tail bounds for sums of geometric and exponential variables. *Statistics & Probability Letters*, 135: 1–6.

Jin, Y.; Li, Y.; Wang, Y.; and Zhou, Y. 2019. On Asymptotically Tight Tail Bounds for Sums of Geometric and Exponential Random Variables. *CoRR*, abs/1902.02852.

Karnin, Z.; Koren, T.; and Somekh, O. 2013. Almost optimal exploration in multi-armed bandits. In *ICML*, 1238–1246.

Kaufmann, E.; Cappé, O.; and Garivier, A. 2016. On the Complexity of Best-Arm Identification in Multi-Armed Bandit Models. *J. Mach. Learn. Res.*, 17: 1:1–1:42.

Kök, A. G.; and Fisher, M. L. 2007. Demand Estimation and Assortment Optimization Under Substitution: Methodology and Application. *Operations Research*, 55(6): 1001–1021.

Luce, R. D. 1959. *Individual choice behavior: a theoretical analysis*. Wiley.

Mannor, S.; and Tsitsiklis, J. N. 2004. The Sample Complexity of Exploration in the Multi-Armed Bandit Problem. *J. Mach. Learn. Res.*, 5: 623–648.

Plackett, R. 1975. The analysis of permutations. *Applied Statistics*, 24: 193–302.

Rusmevichientong, P.; Shen, Z. M.; and Shmoys, D. B. 2010. Dynamic Assortment Optimization with a Multinomial Logit Choice Model and Capacity Constraint. *Operations Research*, 58(6): 1666–1680.

Russo, D. 2016. Simple Bayesian Algorithms for Best Arm Identification. In *COLT*, volume 49, 1417–1418. JMLR.org.

Saha, A.; and Gopalan, A. 2018. Battle of Bandits. In Globerson, A.; and Silva, R., eds., *UAI*, 805–814.

Sauré, D.; and Zeevi, A. 2013. Optimal Dynamic Assortment Planning with Demand Learning. *Manufacturing & Service Operations Management*, 15(3): 387–404.

Talluri, K. T.; and van Ryzin, G. J. 2004. Revenue Management Under a General Discrete Choice Model of Consumer Behavior. *Management Science*, 50(1): 15–33.

Yang, J. 2021. Fully Gap-Dependent Bounds for Multinomial Logit Bandit. In *AISTATS*, volume 130 of *Proceedings of Machine Learning Research*, 199–207.

Zhou, Y.; Chen, X.; and Li, J. 2014. Optimal PAC multiple arm identification with applications to crowdsourcing. In *ICML*, 217–225.