Online Recommendations for Agents with Discounted Adaptive Preferences

Arpit Agarwal[*] William Brown[†]

February 12, 2023

Abstract

For domains in which a recommender provides repeated content suggestions, agent preferences may evolve over time as a function of prior recommendations, and algorithms must take this into account for long-run optimization. Recently, Agarwal and Brown (2022) introduced a model for studying recommendations when agents' preferences are adaptive, and gave a series of results for the case when agent preferences depend *uniformly* on their history of past selections. Here, the recommender shows a k-item menu (out of n) to the agent at each round, who selects one of the k items via their history-dependent *preference model*, yielding a per-item adversarial reward for the recommender.

We expand this setting to *non-uniform* preferences, and give a series of results for γ*-discounted* histories. For this problem, the feasible regret benchmarks can depend drastically on varying conditions. In the "large γ" regime, we show that the previously considered benchmark, the "EIRD set", is attainable for any *smooth* model, relaxing the "local learnability" requirement from the uniform memory case. We introduce "pseudo-increasing" preference models, for which we give an algorithm which can compete against any item distribution with small uniform noise (the "smoothed simplex"). We show NP-hardness results for larger regret benchmarks in each case. We give another algorithm for pseudo-increasing models (under a restriction on the adversarial nature of the reward functions), which works for any γ and is faster when γ is sufficiently small, and we show a super-polynomial regret lower bound with respect to EIRD for general models in the "small γ" regime. We conclude with a pair of algorithms for the memoryless case.

1 Introduction

Today, recommendation systems are an integral part of online platforms for e-commerce, social networks, and content sharing. It has been well-documented that user preferences change over time in response to content recommendations (see Curmei et al. (2022) for an overview), which can lead to self-reinforcing patterns of content consumption and can have a variety of unintended consequences for the user, platform and society; including but not limited to loss of revenue for the platform, or creation of "filter-bubbles" and "echo-chambers" that drive polarization in society. Hence, it becomes important for recommendation systems to incorporate such self-reinforcing patterns into user modeling.

Motivated by this, Agarwal and Brown (2022) introduce a model for adaptive user preferences that depend on the history of interaction with the recommender. In this problem formulation, the recommender is faced with a multi-armed bandit task where items cannot be chosen directly, but rather are selected by an agent with an *adaptive preference model* over menus shown by the recommender. This interaction happens repeats for T rounds, and the reward for each item at any given round may be adversarial. The goal of the recommender is to minimize its regret with respect to a suitable benchmark; the authors identify a feasible regret benchmark (the "EIRD set", roughly corresponding to the set of high-entropy item distributions), while showing linear regret lower bounds for other benchmarks. In this model, agent preferences depend uniformly on the sequence of past interactions, i.e. recent selections are given the same weight as past selections. However, there is much evidence from psychology that humans are recency-biased, which has

[*] `arpit.agarwal@columbia.edu`
[†] `w.brown@columbia.edu`

long been a standard modeling practice in fields from economics to reinforcement learning (see e.g. Doyle (2012)).

We extend the model of Agarwal and Brown (2022) to include agent preferences which depend non-uniformly on the history of past interactions with the recommender, in which the influence of past interactions is discounted by a factor of $\gamma \leq 1$ after each round. Here, we study regret minimization for the online recommendations problem for adaptive agents across the landscape of memory horizons under this new model, including when the memory horizon can be $o(T)$ down to when agents are memoryless. We provide a series of algorithmic and barrier results for a variety of preference model classes and regret benchmarks. Below, we formally describe this problem setting and outline our main contributions.

1.1 Problem Description

In each round $t \in [T]$, the recommender must choose k out of n items to show in a menu to the agent, who will choose one item probabilistically according to their *preference model* M. Under this model, the choice probability of each item depends on a *memory vector* v which encodes the sequence of prior item selections. As in the classical adversarial multi-armed bandit problem, the sequences of rewards for each item may be adversarially chosen. However, a key difference in this setting is that the recommender can only suggest a menu of items to the agents, and the agent will then "pull" one of the arms (items) according to its preference model. As a result, in general it will not be possible to do as well as the best individual item in terms of reward, as an agent may never choose an item more than once every k rounds even if scores are uniform and the item is shown every round. Hence, we must be careful in selecting benchmarks with which we can feasibly compete (in the form of a sublinear regret bound). Notably, the feasibility of a benchmark can depend on the intricacies of how preferences are allowed to adapt over time.

Agarwal and Brown (2022) considered the EIRD set as benchmark for the case of uniform-memory preferences and showed that one can compete against the best item distribution in EIRD. Here, the EIRD set is a subset of item distributions which can be instantaneously realized regardless of the current memory vector via an appropriate randomized menu selection process. They also show negative results for the feasibility of other benchmarks, such as the the best stationary distribution over items resulting from a fixed menu distribution used in every round.

Our results consider both of these benchmarks, and we introduce a new benchmark which we term the "ϕ-smoothed simplex" $\Delta^{\phi}(n)$, which is the set obtained when each item distribution in the standard simplex $\Delta(n)$ is mixed with mass $\phi(1 + o(1))$ on the uniform distribution. In Section 3.1 we give additional characterization of the structure of the EIRD set which highlights the cases in which it can be restrictive, such as when it is impossible to persuade an agent to highly favor a particular item (which also yields an efficient sparse approximation algorithm for menu distributions). We identify a structural condition on preference models which precludes this restricted scenario, which we term the "pseudo-increasing" property, and which aligns with many common settings where exposure to an item tends to increase one's fondness for it (such as in Curmei et al. (2022)). Item scoring functions with this property can still be non-monotone, non-convex, and dependent on many items in intricate ways, yet are constrained to within some multiplicative factor of a linear increasing function of that item's weight in memory. When this property is satisfied, we that show Δ^{ϕ} becomes feasible as a benchmark; if the agent's "exploration" parameter λ is vanishing (e.g. at a $1/\operatorname{poly}(n)$ rate), we can take $\phi = O(\lambda)$ to be vanishing as well, in which case the distance between any item distribution in $\Delta(n)$ and some point in the ϕ-smoothed simplex every also vanishes.

1.2 Overview of Results

We give a series of algorithmic and barrier results for this problem under the γ-discounted setting. This coincides exactly with the uniform memory case of Agarwal and Brown (2022) when $\gamma = 1$; on the whole, our results illustrate the regret benchmarks against which one can compete may change drastically depending on the uniformity of memory and structural conditions on the preference model.

Our first key algorithmic contribution (in Section 3) considers the case when γ approaches 1 (as T grows large), and the "effective horizon" for memory is large (but still $o(T)$). Here, we show that the "local learnability" assumption (i.e. that preference models can be estimated accurately everywhere by only querying points in a small region) required by Agarwal and Brown (2022) is no longer needed. It suffices

Memory	**Preferences**	**Benchmark**
Uniform ($\gamma = 1$)	Locally Learnable	EIRD
$\gamma = 1 - \frac{1}{T^c}$	Smooth	EIRD
$\gamma = 1 - \frac{1}{T^c}$	Smooth + P-I	Δ^ϕ
$\gamma \in [0, 1)$	P-I	$\Delta^{\phi\ (*)}$
Memoryless	Fixed	Menus

Table 1: Sufficient conditions on memory rates and preference models, and their corresponding target benchmarks, for which where $o(T)$ regret is achievable. The first row is due to Agarwal and Brown (2022), and the remainder are contributions of this work. Here, $c < 1$ is a constant; "P-I" denotes that a preference model is pseudo-increasing, and Δ^ϕ denotes the ϕ-smoothed simplex, both of which are introduced in Section 3.3. $^{(*)}$ denotes a restriction on the adversarial nature of rewards, namely that the reward distribution cannot change more than once every $o(T)$ rounds.

for us to only assume that preference scoring functions are smooth, as we will always be able to maintain a preference model hypothesis which is locally accurate in the neighborhood of the memory vector, which cannot change too quickly. We give an algorithm which obtains $o(T)$ regret with respect to EIRD in this case (with a rate depending on γ); when the preference model satisfies the additional "pseudo-increasing" assumption we give a similar algorithm which can compete against Δ^ϕ. We also give a characterization of IRD sets which enables efficiently constructing sparse menu distributions. In each of these cases we also give a negative result that substantially larger regret benchmarks are infeasible, by reducing a NP-complete problem to the problem of computing the best stationary item distribution from fixed menu distributions.

On the other extreme, we consider the case where $\gamma \in [0, 1)$ is some fixed constant, and so the effective memory horizon is $O(1)$ with respect to T. This includes cases where the set of feasible memory vectors is essentially discrete, as there may be a finite grid of well-separated points which are all feasible vectors, such as when $\gamma = 0$ and the memory vector always has its entire mass on the most recently selected item. Despite this difficulty, we have that the pseudo-increasing condition is sufficient (even without smoothness) to compete with Δ^ϕ, and we give an algorithm for this case which obtains sublinear regret with respect to Δ^ϕ for $\gamma \in [0, 1)$, and with a faster rate than the previous algorithm when γ is sufficiently small. The presence of additional structural conditions (such as the pseudo-increasing property) on the scoring functions are necessary; we show a linear regret lower-bound with respect to EIRD for any algorithm when T is quasipolynomially bounded in n. To conclude, we analyze the "memoryless" case when preferences are fixed, and give an algorithm which obtains the optimal $\tilde{O}(\sqrt{T})$ rate yet depends exponentially on k, as well as a $\tilde{O}(T^{2/3})$ algorithm where dependence on other parameters is polynomial.

Our contributions range from proposing a model for non-uniform adaptive preferences and identifying suitable regret benchmarks, to understanding the landscape of feasibility for a range of values of γ that interpolate between the uniform case and the memoryless case via sublinear regret algorithms and barrier results. We summarize our results in Tables 1 and 2.

1.3 Related Work

Stochastic bandits with changing rewards: The stochastic multi-armed bandit problem has been studied in settings where the reward distributions change as a function of arm pulls (Gittins, 1979; Heidari et al., 2016; Levine et al., 2017; Kleinberg and Immorlica, 2018; Leqi et al., 2021; Laforgue et al., 2022; Awasthi et al., 2022; Papadigenopoulos et al., 2022). Most recent work on this problem has focused on specific models for reward evolution that are motivated by user satiation, user boredom, congestion etc. There are several crucial differences between our setting and this setting. Firstly, we consider adversarial rewards, whereas this setting considers stochastic rewards that evolve according to a fixed dynamics. Secondly, we consider a recommendation setting where the learner plays a menu of arms, and the user choice behavior over menus evolves over time. In contrast, these works consider a classic bandit setting where the learner plays a single

Memory	**Preferences**	**Barrier**
Uniform	Locally Learnable	Menus $^{(1)}$
Uniform	Non-L.L.	EIRD $^{(2)}$
$\gamma = 1 - \frac{1}{T^c}$	Smooth	Menus $^{(3)}$
$\gamma = 1 - o\frac{1}{T^c}$	Smooth + P-I	Menus $^{(3)}$
$\gamma \in (0, \frac{1}{2})$	Smooth	EIRD $^{(2)}$

Table 2: Conditions on memory rates and preference models for which formal barriers exist against competing with particular benchmarks for regret. The uniform barriers are due to Agarwal and Brown (2022), with the rest from this work. Here, $^{(1)}$ denotes a linear regret lower bound, $^{(2)}$ denotes a linear regret lower bound for quasipolynomial horizons, and $^{(3)}$ denotes NP-hardness.

arm.

Models of preference dynamics: There has also been substantial work in understanding preference dynamics in recommendation systems (Hazla et al., 2019; Gaitonde et al., 2021; Dean and Morgenstern, 2022). Hazla et al. (2019); Gaitonde et al. (2021) studied a model for preference dynamics where preferences are represented by vectors and these vectors drift towards the direction of agreement/disagreement on a randomly drawn issue. Dean and Morgenstern (2022) studied a similar model, but in the context of personalized recommendations of a single item. Even though these papers consider preference dynamics similar to our work, their goal is to study conditions under which polarization occurs. In contrast, our goal is to minimize regret for the platform under these dynamically evolving preferences. Restricted history has also been identified by Schneider and Vodrahalli (2022) as a useful property for modeling agent behavior with online learning algorithms.

Reinforcement Learning: There has also been work on using reinforcement learning for recommendations in order to maximize long-term rewards (Ie et al., 2019; Zhan et al., 2021; Chen et al., 2019). However, either these works do not consider models of preference evolution or lack rigorous theoretical guarantees on the achieved performance. We consider a very general model for preference evolution as compared to these works, and provide a rigorous treatment of achievability and hardness under various benchmarks. Moreover, it does not seem possible to frame our model as a Markov decision process (MDP) and simultaneously achieve good regret guarantees using off-the-shelf reinforcement learning techniques.

Dueling Bandits: The "dueling bandits" framework studies a recommendation problem similar to ours where the learner plays more than one arm in each trial and observes relative feedback between arms (Yue and Joachims, 2009; Yue et al., 2012; Agarwal et al., 2020; Rangi et al., 2021). However, in contrast to our setting, these works consider preference models which are fixed *a priori*, and do not change as a function of item history.

1.4 Organization

We introduce the problem setting and key definitions in Section 2. In Section 3, we study the "large γ" regime where memory changes slowly, and in Section 4 we consider the "small γ" regime where memory vectors may change quickly, giving algorithmic and barrier results in each case. In Section 5 we give algorithms for the memoryless case. We defer most proofs to the appendix.

2 Model And Preliminaries

2.1 Setting

We recall the standard setup from Agarwal and Brown (2022) for the online recommendations problem for an agent with adaptive preferences. At any time, there is some *memory vector* $v \in \Delta(n)$, which expresses some function of the prior selections of the agent. The *preference model* of an agent is a mapping $M : \Delta(n) \to [0, 1]^n$

which assigns scores $M(v)_i = f_i(v)$ according to preference functions $f_i : \Delta(n) \to [0,1]$ for each item. In an instance of this problem, there is a universe of n items, a menu size $k < n$, a preference model M, a memory update rule U, and a sequence of reward vectors $r_1, \ldots, r_T$ for the recommender. In each round $t \in \{1, \ldots, T\}$:

- The recommender chooses a menu K_t, consisting of k distinct items from $[n]$, which is shown to the agent;
- The agent selects one item $i_t \in K_t$, chosen at random according to the distribution given by:
$$p_{K_t,v_t,i} = \frac{f_i(v_t)}{\sum_{j \in K_t} f_j(v_t)};$$
- The memory vector is updated to $v_{t+1} = U(v_t, i_t, t)$ by the update rule;
- The recommender observes receives reward $r_t(e_{i_t}) \in [0,1]$ for the chosen item, where e_{i_t} is the unit vector for i_t.

The goal of the recommender is to minimize their regret over T rounds with respect to some benchmark set. For any preference model M and current memory vector v, the set of item choice distributions which can be instantaneously realized by showing the agent a menu sampled from some distribution is denoted $\text{IRD}(v, M)$, or simply the IRD set, and is given by

$$\text{IRD}(v, M) = \operatorname*{convhull}_{K \in \binom{n}{k}} p_{K,v},$$

where $p_{K,v}$ denotes agent's item selection distibution conditioned on being shown a menu K, given by

$$p_{K,v,i} = \frac{f_i(v)}{\sum_{j \in K} f_j(v)}$$

for each item i in K (and 0 otherwise), with f_i being the preference scoring function for item i. The set of item distributions which are instantaneoulsy realizable *everywhere* (i.e. from any memory vector), denoted EIRD, is given by

$$\text{EIRD}(M) = \bigcap_{v \in \Delta(n)} \text{IRD}(v, M).$$

To ensure that EIRD is non-empty, we typically assume that scoring functions f_i are in fact bounded in the range $[\lambda, 1]$ for some constant λ, sometimes with required lower bounds (e.g. we require $\lambda \geq \frac{k^2}{n}$ in Section 3.2 and $\lambda \geq \frac{\sigma^2 k}{n}$ in Section 4.1, but only that $\lambda > 0$ in Section 3.4).

2.2 Discounted Memory Agents

Throughout, with the exception of Section 5, we consider agents whose memory update rules are γ-*discounted.*

Definition 1 (Discounted Memory Updating)**.** Under the γ-discounted memory update rule U_γ, for some $\gamma \in [0,1]$, when an item i_t is selected at round t, the memory vector v_t is updated to $v^{t+1} = U_\gamma(v_t, i_t, t)$, with

$$v_i^{t+1} = \frac{\sum_{s=1}^{t} \gamma^{t-s} \cdot x_i^s}{\sum_{s=1}^{t} \gamma^{t-s}},$$

As in other problem settings involving discount factors (such as MDPs), we can think of larger values of γ as corresponding to having larger "effective horizons" for memory; the case where $\gamma = 1$ is equivalent to the uniform memory setting from Agarwal and Brown (2022).

2.3 Smooth Preference Models

Many of our results consider preference models with the property that each scoring function has Lipschitz gradients, in addition to being bounded above 0, which we refer to as *smooth* preference models.

Definition 2 (Smooth Preference Models)**.** A preference model M is (λ, L)-smooth if each scoring function f_i takes values in $[\lambda, 1]$, is L-Lipschitz over $\Delta(n)$, and if the scoring functions have a constant sum $\sum_i f_i(v) = C$ for any v.

This property allows for quite a broad class of functions, and is satisfied by each of the classes in Agarwal and Brown (2022) (e.g. low-degree polynomials) with appropriate parameters. Despite its generality, we show in Section 3 that this assumption alone is sufficient to enable us to always maintain an accurate *local* approximation of the model, provided that memory does not change too quickly, by periodically implementing a query rountine. The requirement on the sum of functions being constant is without loss of generality up to polynomial factors, and further is justified by psychological phenomena (see e.g. Curmei et al. (2022)) and is minimally restrictive, as it merely enforces normalization of scores; agents may gain interest in certain items based on recent selections, which typically corresponds with a drop in interest for other items.

2.4 Other Preliminaries

We assume the reader is familiar with basic terminology regarding bandit and online convex optimization; see e.g. Hazan (2019) for an overview. For a *regret benchmark* S, we define the regret with respect to S for an algorithm $\mathcal{A}$ (which chooses action a_t at time t) as:

$$\text{Reg}_S(\mathcal{A}; T) = \max_{x \in S} \sum_{t=1}^{T} r_t(x) - r_t(a_t).$$

We make use of the following result from Agarwal and Brown (2022), which extends the FKM bandit linear optimization algorithm Flaxman et al. (2004) to *contracting* action sets with perturbed actions, i.e. where the feasible action space in each round is a convex subset of that in the prior round, and where an adversary may perturb each action by $\epsilon = O(T^{-1/4})$.

Theorem 1 (Agarwal and Brown (2022))**.** There is an algorithm RC-FKM for bandit linear optimization which obtains $O(T^{3/4})$ regret against the best point in the final set in a length-T sequence of contracting convex action sets, with each action taken from the set revealed at each round and adversarially perturbed by at most $O(T^{-1/4})$.

Throughout, we use $\Delta(n)$ to denote the simplex over n items, $d_{TV}(v, v')$ to denote the total variation distance between distributions; we use the ℓ_2 norm unless denoted otherwise (e.g. $\|x\|_1$), and we use $B_\epsilon(x)$ to denote the norm ball of radius ϵ around the point x.

3 Agents with Long Memory Horizons

We begin by considering the case where γ is close to 1, and the effective horizon for memory is large, i.e. memory can only be truncated to some sequence of prior rounds with length at least $o(T)$ without drastically changing the resulting vector. This implies that memory vectors do not change too quickly, and further that every point in Δ^n is *feasible* up to small error by *some* appropriate sequence of item selections (if we were able to dictate the agent's item choices). In Section 3.1 we give a result on the structure of IRD sets which enables key subroutines of our algorithms in Sections 3.2 and 3.3. In Section 3.5 we show NP-hardness results for relaxing each benchmark from the previous two sections.

3.1 Characterizing IRD via Menu Times

We introduce a notion of the *menu time* required by each item in order to induce a particular item distribution x from a memory vector v, which enables us to more directly characterize the set of feasible item distributions, as well as avoid the exponential dependence on k in runtime from the linear programming construction of menu distributions in Agarwal and Brown (2022). The menu time quantities μ_i are normalized such that $\sum_i \mu_i = k$, and satisfy:

$$\mu_i \propto \frac{x_i}{f_i(v)}.$$

We show that an item distribution x can be realized from a memory vector v if and only if $\max_i \mu_i \leq 1$.

Lemma 1. An item distribution x belongs to $\mathrm{IRD}(v, M)$ if and only if we have that the menu time μ_i for each item is at most 1. If this condition holds, there is a $O(n/(k\epsilon))$ time algorithm for constructing a menu distribution z, with positive mass on at most $O(\frac{n}{k^2\epsilon})$ menus, which results in an induced item distribution $\hat{x}$ satisfying $\|x - \hat{x}\|_\infty \leq \epsilon$.

Proof Sketch. For any $x \in \mathrm{IRD}(v, M)$, we consider an arbitrary convex combination of the menu-conditional item distributions given by items' scores $f_i(v)$, and show that converting this menu distribution into menu times by "charging" each menu in proportion with the inverse of the sum of its item scores results in a menu time vector satisfying $\|\mu\|_1 = k$ and $\|\mu\|_\infty \leq 1$.

Given a menu time vector satisfying these conditions, we can construct such a distribution by greedily choosing the k items with highest remaining menu time, and decreasing the remaining menu time of the selected items by ϵ. The mass of each added menu in our final distribution z will be inversely proportional to the sum of scores of items in the menu. This allows cancellation of the terms for sums of menu scores, resulting in a menu distribution where the selection probability of an item is proportional to its score $f_i(v)$ and the number of stages in which it was added to the menu. As the number of stages in which an item is added to a menu is proportional to its menu time, and its menu time is proportional to $x_i/f_i(v)$, the resulting induced item distribution is simply proportional to x_i as desired. □

3.2 Targeting EIRD

Here, we describe our algorithm for obtaining sublinear regret with respect to EIRD for agents with smooth preference models. If given access to M exactly, RC-FKM could be run directly by computing a mapping between menu distributions and item distributions, as in Lemma 1. Our algorithm periodically estimates the value of M in the neighborhood around the current memory vector by conducting local queries to each learn scoring function. We alternate between "learning" and "optimizing" stages, running RC-FKM during the optimization stage and re-learning our model M^* once the memory vector moves more than a distance ρ from the center v^* of our previous learning stage. Following an initial burn-in period, the memory vector is sufficiently "concentrated" to ensure slow movement, and thus model estimates which maintain accuracy for a long time relative to the length of the learning stage, and so the regret-bounded optimization stages account for $T - o(T)$ steps in total.

Theorem 2. For an agent with a (λ, L)-smooth preference model M and γ-discounted memory for $\gamma = 1 - \frac{1}{T^c}$, where $c \in (0, 1)$, Algorithm 1 obtains regret

$$\mathrm{Reg}_{\mathrm{EIRD}}(\mathcal{A}_1; T) = \tilde{O}\left(T^{1-c/4} + T^c\right)$$

with respect to $\mathrm{EIRD}(M)$.

We show in Section 3.5 that it is impossible to strictly beat EIRD in polynomial time (assuming NP $\not\subseteq$ RP) without additional structural assumptions beyond smoothness; in Section 3.3 we explore such an assumption which enables us to run a similar algorithm, yet compete against a much larger set of item distributions.

3.3 Pseudo-Increasing Functions and $\Delta^\phi(n)$

A major motivation of Agarwal and Brown (2022) for considering EIRD was due to the difficulties of exploration under uniform memory, as the current history cannot be repeatedly "washed away" without requiring exponentially many rounds. Considering discount factors of $\gamma < 1$ introduces the possibility that

Algorithm 1 (Targeting EIRD for Smooth Models).

```
Run BurnIn($\gamma$) for $t_{\text{burn}} = T^c$ rounds
Initialize RC-FKM to run for $T - t_{\text{burn}}$ rounds
while $t < T$ do
    Run Query($v_t$) for $t_{\text{query}} = \tilde{O}(T^{-c/2})$ rounds
    Let $v^* = v_t$
    Let $M^* = \{\tilde{f}_i(v^*) : i \in [n]\}$ (from Query($v_t$))
    while $v_t \in B_\rho(v^*)$ do
        Get $x_t$ from RC-FKM
        Let $z_t = $ MenuDist($v_t, x_t, M^*$)
        Sample menu $K_t \sim z_t$, update RC-FKM
    end while
end while
```

we can efficiently explore the space of feasible vectors and compete against item distributions which lie outside of EIRD, i.e. item distributions which may require the memory vector to lie in a small region in order to be feasible. We identify a structural property which enables this, namely that scoring functions are *pseudo-increasing.*

Definition 3 (Pseudo-Increasing Scoring Functions)**.** A scoring function $f_i : [0,1]^n \to [\frac{\lambda}{\sigma}, 1]$ is (σ, λ, L)-pseudo-increasing for $\sigma \geq 1$ and $\lambda > 0$ if

$$\frac{(1-\lambda)v_i + \lambda}{\sigma} \leq f(v) \leq \sigma((1-\lambda)v_i + \lambda).$$

We say that a preference model M is pseudo-increasing if, for every item i, the scoring function f_i is pseudo-increasing. When this property is satisfied, it becomes possible to induce any item distribution which is in the neighborhood of the current memory vector v_t, provided that v_t is not too close to the boundary of the simplex. This motivates a target regret benchmark of all such points which are not too close to the boundary in any dimension, which we term the *ϕ-smoothed simplex.*

Definition 4 (ϕ-Smoothed Simplex)**.** Let b^i_ϕ denote the "smoothed basis vector" for item i, where $b^i_{\phi,i} = 1-\phi$ for $\phi \in (0,1)$ and $b^i_{\phi,j} = \frac{\phi}{n-1}$ for every other item j. We define the ϕ-smoothed simplex $\Delta^\phi(n)$ as the convex hull of the n smoothed basis vectors.

This is equivalent to the set which results when any point in the standard simplex is mixed with $\frac{\phi n}{n-1}$ uniform noise. We show that this enables a "local feasibility" property, namely that for a point $v \in \Delta^\phi(n)$, the neighborhood around v is contained in $\text{IRD}(v, M)$. Intuitively, this holds because scores cannot be significantly far from an item's current memory vector weight, and so the required "menu time" (as in Lemma 1) cannot be too large.

Lemma 2. *Let M be a (σ, λ)-pseudo-increasing preference model with $\sigma \leq \sqrt{4(n-1)/k}$, and let $v \in \Delta^\phi(n)$ be a vector in the ϕ-smoothed simplex, for $\phi \geq 4k\lambda\sigma^2$. Then, $x \in \text{IRD}(v, M)$ for any vector $x \in B_{\lambda\phi}(v) \cap \Delta^\phi(n)$.*

We no longer require explicit lower bounds on λ, and so we can take our regret benchmark to be approaching the entire simplex as λ goes to 0 by appropriately choosing $\phi = O(\lambda)$. This presents a stark constrast with the EIRD benchmark, as the pseudo-increasing property now suggests that it may be possible to persuade the agent to pick the best item in nearly every round, rather than in only a $O(k/n)$ fraction of rounds (which may occur in Section 3.2 $f_i = \lambda = O(k^2/n)$).

3.4 A No-Regret Algorithm for $\Delta^\phi(n)$

The key difference between Algorithm 1 and our approach here is that rather than taking a step with our copy of RC-FKM during every round of the optimization stage, we treat each optimization stage as a *single* gradient step, with our domain fixed as the smooth simplex rather than a contracting sequence of sets containing EIRD. We take advantage of the fact that, with appropriately calibrated step sizes, we can complete each step without exiting the ball in which our preference model estimate is ϵ-accurate (and in which the Lemma 2 guarantee holds). While we no longer need the "contracting" feature of RC-FKM, we still make use of it rather than other algorithms for bandit linear optimization due to its ability to tolerate imprecision in action specifications and rewards.

Theorem 3 (Pseudo-Increasing Discounted Regret Bound)**.** For any agent with a preference model M which is (σ, λ)-pseudo-increasing and $(\frac{\lambda}{\sigma}, L)$-smooth with $\sigma \leq \sqrt{4(n-1)/k}$, and with γ-discounted memory for $\gamma = 1 - \frac{1}{T^c}$, Algorithm 2 obtains regret

$$\mathrm{Reg}_{\Delta^{\phi}(n)}(\mathcal{A}_2; T) = \tilde{O}\left(T^{1-c/8} + T^c\right)$$

with respect to the ϕ-smoothed simplex, for any $\phi \geq 4k\lambda\sigma^2$.

Here, we assume that both ϕ and λ are constant with respect to T (e.g. $1/\operatorname{poly}(n)$); we note that the analysis can be modified to allow for both parameters to decay as T grows, at the cost of a slower asymptotic regret rate. Rates can also be further optimized for a given c by appropriately parameterizing t_{query} and t_{step}, which we discuss in the proof.

Algorithm 2 (Targeting Δ^{ϕ} for Pseudo-Inc. Models).

```
Run BurnIn($\gamma$) for $t_{\text{burn}} = T^c$ rounds
Initialize RC-FKM to run for $\frac{T - t_{\text{burn}}}{t_{\text{query}} + t_{\text{step}}}$ steps
while $t < T$ do
    Run Query($v_t$) for $t_{\text{query}} = \tilde{O}(T^{-c/2})$ rounds
    Let $v^* = v_t$
    Let $M^* = \{\tilde{f}_i(v^*) : i \in [n]\}$ (from Query($v_t$))
    Get $x^*$ from RC-FKM
    for $t_{\text{step}}$ rounds such that $v_t \in B_\rho(v^*)$ do
        Let $z_t$ = MenuDist($v_t, x^*, M^*$)
        Sample menu $K_t \sim z_t$
    end for
    Update RC-FKM with average reward from $x^*$
end while
```

3.5 Hardness for Alternate Benchmarks

We give a set of hardness results for both of the previous cases against the possibility of relaxing to larger benchmark sets. Each of these constructions proceeds by showing that an instance of the "Max Independent Set" problem, which is NP-hard to approximate, can be encoded in a smooth preference model for an agent, such that any strategy attaining a particular average reward corresponds to a solution to the problem.

Theorem 4. Unless RP $\supseteq$ NP, there is no algorithm which runs in time polynomial in n and T and which obtains regret $o(T)$ for any $T = O(\text{poly}(n))$ against either the ϕ-smoothed simplex for any $\phi < 1/2$, the set of all menu distributions, or against the set of all item distributions contained in their own IRD set, for agents with arbitrary (λ, L)-smooth memory models.

Additionally, for agents with arbitrary σ-pseudo-increasing and (λ, L)-smooth memory models there is no such algorithm which obtains regret $o(T)$ for any $T = O(\text{poly}(n))$ against the set of all menu distributions, or against the set of all item distributions contained in their own IRD set.

This suggests that competing with any benchmark that is strictly better than the best point in EIRD will require additional structural assumptions on the preference model, as otherwise it is possible for the unique memory states which can beat the best point in EIRD, even for fixed rewards, to be computationally difficult to find.

4 Agents with Short Memory Horizons

When the discount factor of the agent is small enough that memory vectors may move rapidly, we lose the precision required by the algorithms in Section 3 in order to implement local learning, and in fact the feasible state space may more closely resemble a discrete grid, with memory vectors encoding the sequence of items chosen over an effective horizon which is constant with respect to T. Nonetheless, for pseudo-increasing models we give an algorithm which we call EXP-ϕ, which obtains $o(T)$ regret with respect to $\Delta^{\phi}(n)$ for *any* value of $\gamma \in [0, 1)$ under an assumption about the restricted adversarial nature of rewards. Here, we assume that rewards are stochastic rather than adversarial for windows of length $o(T)$, but distributions may change adversarially between each window; we require a slightly larger lower bound on ϕ (yet still $O(\lambda)$).

4.1 A No-Regret Algorithm for Pseudo-Increasing Models

The idea behind EXP-ϕ is to view each vertex of the smoothed simplex as an action for a multi-armed bandit problem, where each "pull" corresponds to several rounds. When we "commit" to playing an item in the menu for a sufficiently long time, while otherwise playing items with the smallest weight in memory, the pseudo-increasing property will ensure that the selection frequency of that item gravitates towards its vertex in the smoothed simplex. Further, as we are no longer attempting to learn the preference model explicitly, we can relax the smoothness requirement, and so scoring functions may in fact be discontinuous.

Theorem 5. For any agent with a preference model M which is (σ, λ)-pseudo-increasing for which each $f_i(v) \in [\lambda, 1]$ for $\lambda \geq \frac{\sigma^2 k}{n}$, and with γ-discounted memory for $\gamma \in [0, 1)$, when losses are drawn from a distribution which changes at most once every $t_{\text{hold}} = \tilde{O}\left(\frac{T^{2/3}}{1-\gamma}\right)$ rounds, Algorithm 2 obtains regret at most

$$\text{Reg}_{\Delta^{\phi}(n)}(\mathcal{A}_3; T) = \tilde{O}(T^{5/6})$$

with respect to $\Delta^{\phi}(n)$, for any $\phi \geq 2\lambda k^3 \sigma^6$.

Algorithm 3 (EXP-ϕ).

```
Initialize EXP3 to run for $T/t_{\text{hold}}$ steps
while $t < T$ do
    Sample arm $i^*$ from EXP3
    for $O(1/(1-\gamma))$ rounds do
        Let $K_t = \{i^*\} + \text{argmin}_{j \neq i}^{k+1} v_j$
    end for
    Update EXP3 with average reward of $i^*$
end while
```

4.2 Barriers for General Models

If we cannot assume that preferences are pseudo-increasing, then it appears difficult to compete even against EIRD for arbitrary smooth models. We show a regret lower bound with respect to EIRD for any algorithm over a quasipolynomial time horizon by constructing preference models in which the optimal strategy depends delicately on the current memory vector, and which simultaneously induces fast exploration over a discrete state space.

Theorem 6. For any $\gamma \in (0, 1/2)$, there is a set of (λ, L)- smooth preference models $\mathcal{M}$ with $\lambda = O(1/n)$ and $L = \text{poly}(n)$ such that any algorithm must have expected regret $\Omega(T)$ for any $T \in O(n^{\log(n)})$ when the preference model M is sampled uniformly from $\mathcal{M}$.

Our approach is to observe that every feasible memory vector encodes a unique truncated history of length $O(\log n)$, resulting in an implicit state space of size $O(n^{\log(n)})$. We design preference models in which the optimal policy is implementable by inducing the uniform distribution at each round, which lies inside

EIRD, yet requires identifying a specific set of alternate items to place in the menu deterministically at each state to maximize the selection probability of item 1. We show that any competitive strategy also necessarily explores many states, and so any algorithm will frequently reach states where it cannot identify the optimal menu distribution (which is generated randomly).

5 Memoryless Preferences

Finally, to complete the landscape of potential memory horizons, we consider the "memoryless" case where agent preferences are fixed, and independent of past selections. This variant of the problem is notably simpler, as per-round item distributions are fixed as a linear function of the menu distribution used, which will allow us to compete against the benchmark of all fixed menus; however, it remains an interesting question as to whether the structure of the recommendation menu setting can be exploited to improve dependence on parameters other than T, as enumerating all menus is undesirable when k is large.

In the appendix, we give two $o(T)$ algorithms for this case. The first is a direct application of EXP3 to the set of all menus, which obtains optimal dependence on T but is exponential in k. We also give an algorithm which removes the dependence on k entirely in exchange for a weaker rate with respect to T, in which we estimate each preference score directly and conduct bandit linear optimization using the "menu time" distribution construction from Section 3.1.

Theorem 7. For an agent with fixed preferences, there is an algorithm which obtains regret $\tilde{O}(\sqrt{Tn^k})$, as well as an algorithm which obtains regret $\tilde{O}(T^{2/3}\,\mathrm{poly}(n))$ with polynomial time per-round computation.

6 Conclusion

We extend the setting from Agarwal and Brown (2022) beyond the uniform-memory case to encompass a spectrum of memory update rates which may be more realistic for a variety of scenarios, and we give a series of novel algorithmic results under varying conditions. In particular, we give sublinear algorithms for both EIRD and our "smooth simplex" benchmark under relaxed conditions, while simultaneously proving barrier results against further relaxation. Several key questions remain open, particularly in determining optimal regret rates in cases where sublinear regret algorithms exist, and identifying optimal tradeoffs between parameters.

References

Arpit Agarwal and William Brown. Diversified recommendations for agents with adaptive preferences. In *NeurIPS*, 2022.

Arpit Agarwal, Nicholas Johnson, and Shivani Agarwal. Choice bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18399–18410. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/d5fcc35c94879a4afad61cacca56192c-Paper.pdf`.

Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, USA, 1st edition, 2009. ISBN 0521424267.

Pranjal Awasthi, Kush Bhatia, Sreenivas Gollapudi, and Kostas Kollias. Congested bandits: Optimal routing via short-term resets. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 1078–1100. PMLR, 2022.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H. Chi. Top-k off-policy correction for a REINFORCE recommender system. In J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman, editors, *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 456–464. ACM, 2019. doi: 10.1145/3289600.3290999. URL `https://doi.org/10.1145/3289600.3290999`.

Mihaela Curmei, Andreas A. Haupt, Benjamin Recht, and Dylan Hadfield-Menell. Towards psychologically-grounded dynamic preference models. In Jennifer Golbeck, F. Maxwell Harper, Vanessa Murdock, Michael D. Ekstrand, Bracha Shapira, Justin Basilico, Keld T. Lundgaard, and Even Oldridge, editors, *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022*, pages 35–48. ACM, 2022.

Sarah Dean and Jamie Morgenstern. Preference dynamics under personalized recommendations. In David M. Pennock, Ilya Segal, and Sven Seuken, editors, *EC '22: The 23rd ACM Conference on Economics and Computation, Boulder, CO, USA, July 11 - 15, 2022*, pages 795–816. ACM, 2022.

John Doyle. Survey of time preference, delay discounting models. *Judgment and Decision Making*, 8, 04 2012. doi: 10.2139/ssrn.1685861.

Abraham Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *CoRR*, cs.LG/0408007, 2004.

Jason Gaitonde, Jon M. Kleinberg, and Éva Tardos. Polarization in geometric opinion dynamics. In Péter Biró, Shuchi Chawla, and Federico Echenique, editors, *EC '21: The 22nd ACM Conference on Economics and Computation, Budapest, Hungary, July 18-23, 2021*, pages 499–519. ACM, 2021.

John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41(2):148–164, 1979.

Elad Hazan. Introduction to online convex optimization. *CoRR*, abs/1909.05207, 2019. URL `http://arxiv.org/abs/1909.05207`.

Jan Hazla, Yan Jin, Elchanan Mossel, and Govind Ramnarayan. A geometric model of opinion polarization. *CoRR*, abs/1910.05274, 2019.

Hoda Heidari, Michael J. Kearns, and Aaron Roth. Tight policy regret bounds for improving and decaying bandits. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1562–1570. IJCAI/AAAI Press, 2016. URL `http://www.ijcai.org/Abstract/16/224`.

Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Tushar Chandra, and Craig Boutilier. Slateq: A tractable decomposition for reinforcement learning with recommendation sets. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 2592–2599. ijcai.org, 2019. doi: 10.24963/ijcai.2019/360. URL `https://doi.org/10.24963/ijcai.2019/360`.

Robert Kleinberg and Nicole Immorlica. Recharging bandits. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 309–319. IEEE Computer Society, 2018. doi: 10.1109/FOCS.2018.00037. URL `https://doi.org/10.1109/FOCS.2018.00037`.

Pierre Laforgue, Giulia Clerici, Nicolò Cesa-Bianchi, and Ran Gilad-Bachrach. A last switch dependent analysis of satiation and seasonality in bandits. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 971–990. PMLR, 28–30 Mar 2022.

Liu Leqi, Fatma Kilinç-Karzan, Zachary C. Lipton, and Alan L. Montgomery. Rebounding bandits for modeling satiation effects. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 4003–4014, 2021.

Nir Levine, Koby Crammer, and Shie Mannor. Rotting bandits. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3074–3083, 2017.

Orestis Papadigenopoulos, Constantine Caramanis, and Sanjay Shakkottai. Non-stationary bandits under recharging payoffs: Improved planning with sublinear regret. *CoRR*, abs/2205.14790, 2022. doi: 10.48550/arXiv.2205.14790. URL `https://doi.org/10.48550/arXiv.2205.14790`.

Anshuka Rangi, Massimo Franceschetti, and Long Tran-Thanh. Sequential choice bandits with feedback for personalizing users' experience, 2021. URL `https://arxiv.org/abs/2101.01572`.

Jon Schneider and Kiran Vodrahalli. History-restricted online learning, 2022. URL `https://arxiv.org/abs/2205.14519`.

Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1201–1208, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553527.

Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2011.12.028. URL `https://www.sciencedirect.com/science/article/pii/S0022000012000281`. JCSS Special Issue: Cloud Computing 2011.

Ruohan Zhan, Konstantina Christakopoulou, Ya Le, Jayden Ooi, Martin Mladenov, Alex Beutel, Craig Boutilier, Ed H. Chi, and Minmin Chen. Towards content provider aware recommender systems: A simulation study on the interplay between user and provider utilities. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia, editors, *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3872–3883. ACM / IW3C2, 2021. doi: 10.1145/3442381.3449889. URL `https://doi.org/10.1145/3442381.3449889`.

A OMITTED PROOFS FOR SECTION 3

A.1 Proof of Lemma 1

Lemma 1 (IRD Construction)**.** *An item distribution x belongs to* $\mathrm{IRD}(v, M)$ *if and only if we have that the menu time μ_i for each item is at most* 1*. If this condition holds, there is a $O(n/(k\epsilon))$ time algorithm for constructing a menu distribution z, with positive mass on at most $O(\frac{n}{k^2\epsilon})$ menus, which results in an induced item distribution $\hat{x}$ satisfying $\|x - \hat{x}\|_\infty \leq \epsilon$.*

Proof. We make use of a notion which we call the *menu time* required for each item, corresponding roughly to the relative frequency with which an item must be included in the menu. The total menu time will sum to k, and each item will need menu time at most 1. Let the menu time μ_i for each item be given by:

$$\mu_i := \frac{k \cdot \frac{x_i}{f_i(v)}}{\sum_{j=1}^{n} \frac{x_j}{f_j(v)}}.$$

We first show that any $x \in \mathrm{IRD}(v, M)$ results in μ_i at most 1 per item. For any $x \in \mathrm{IRD}(v, M)$, consider an arbitrary convex combination of the menu-conditional item distributions given by items' scores $f_i(v)$, with the probability of each menu given by p_K. Allocate "menu time units" μ_K to each menu K in proportion with $p_K \cdot \sum_{i \in K} f_i(v)$, such that $\sum_K \mu_K = k$, and further let $\mu_{K,i} = \mu_K / k$ for each $i \in K$. Observe that selection probability of an item i is given by:

$$\begin{aligned}
x_i &= \sum_{K : i \in K} p_K \cdot \frac{f_i(v)}{\sum_{j \in K} f_j(v)} \\
&= \frac{1}{Z} \sum_{K : i \in K} \frac{\mu_K}{k} \cdot f_i(v) \\
&= \frac{f_i(v)}{Z} \sum_{K : i \in K} \mu_{K,i}
\end{aligned}$$

where Z is a normalizing constant such that $\sum_K \mu_K = k$, and so we have that $\sum_K \mu_{K,i} \leq 1$ as each μ_K is positive. Further, we have that

$$\begin{aligned}
\sum_{K : i \in K} \mu_{K,i} &= Z \cdot \frac{x_i}{f_i} \\
&= \frac{x_i}{f_i} \cdot \frac{k}{\sum_{j=1}^{n} \frac{x_j}{f_j(v)}}
\end{aligned}$$

upon solving for Z such that $\sum_K \mu_K = k$, which gives us that

$$\sum_{K : i \in K} \mu_{K,i} = \mu_i,$$

yielding that $\mu_i \leq 1$ for each item.

Next, we show that menu times can be used to construct a menu distribution to realize any $x \in \mathrm{IRD}(v, M)$. We construct a menu distribution z_v over $\frac{1}{\tau}$ stages for some $\tau \geq 0$ (such that the resulting quantity is an integer), which approximately induces the item distribution x to arbitrary precision (as we take the limit of τ to 0) when v is the current memory vector. We add a single menu K_h in each stage $h \in \{1, \ldots, \frac{1}{\tau}\}$, with an assigned relative probability score $Z(K_h)$ determined by the weights and scores of its items, and the resulting distribution will be given by:

$$z_v(K_h) = \frac{Z(K_h)}{\sum_{j=1}^{\frac{1}{\tau}} Z(K_j)}$$

for each menu K_h. As x lies in $\mathrm{IRD}(v, M)$, we have that $\mu_i \leq 1$ for each item. To select each menu in our sequence, we first compute the initial menu time $\mu_i^1 = \mu_i$ for each item, which we will update to μ_i^{h+1} after choosing each menu K_h. We proceed greedily by selecting the items with the k largest values μ_i^h for each menu K_h, and we set the relative probability score as:

$$Z(K_h) = \sum_{i \in K_h} f_i(v).$$

We then set $\mu_i^{h+1} = \mu_i^h - \tau$ for each item $i \in K_h$, and $\mu_j^{h+1} = \mu_j^h$ for all other items. Observe that in each stage we decrement $\sum_i \mu_i^h$ by $k\tau$, and so after all $\frac{1}{\tau}$ stages we will have $\sum_i \mu_i^{1/\tau+1} = 0$. Some of these terms may be negative (e.g. if μ_i is not an integer multiple of τ), but we show that they are at least $-\tau$, and further that every $\mu_i^{1/\tau+1}$ is at most τ after the stages complete. For a quantity μ_i^h to drop below 0, there must be less than k items with remaining $\mu_i^h \geq \tau$. Items which have initial values μ_i^1 which is among the largest k without tie-breaking will be added to every menu until we reach a stage h where there are at least $k+1$ items within τ of the maximal value μ_i^h, as values drop by at most τ. This event must have occurred by the time stage $\frac{1}{\tau}$ completes, as the largest value initial μ_i value for an item is at most 1, and its remaining menu time will drop by τ for each stage that it is in the menu. Once this occurs, the top $k+1$ v items at any stage will always have remaining time values within τ of each other, as any in the top cannot drop more than τ below their prior value. As such, following a stage where some μ_i^h is below zero, we have that every item's value μ_i^h is at most τ. Further, as the final values $\mu_i^{1/\tau+1}$ sum to 0, a quantity cannot drop below $-\tau$, as this would imply that an item with a negative quantity is in the top k, and that the sum of the values μ_i^h is less than $k\tau$, which can only occur after the final stage completes. As such, we have that:

$$\mu_i - \left(\mu_i^1 - \mu_i^{1/\tau+1}\right) \in [-\tau, \tau],$$

and so the proportion of stages in which an item i is included in the menu is approximately equal to its menu time μ_i.

Taking the limit of τ to zero, we obtain $\mu_i^1 - \mu_i^{1/\tau+1} \approx \mu_i$ to arbitrary precision, for which the resulting menu distribution is given by:

$$z_v(K_h) = \frac{\sum_{i \in K_h} f_i(v)}{\sum_{s=1}^{\frac{1}{\tau}} \sum_{j \in K_s} f_j(v)}.$$

When a menu K is shown to the agent, their selection probability for each item i is given by:

$$\Pr\left[\text{Agent selects } i \mid K \text{ shown}\right] = \frac{f_i(v)}{\sum_{j \in K} f_j(v)}.$$

For each menu K_h that an item i is included in, the probability that both K_h is sampled from z_v and that an agent picks item i is:

$$\begin{aligned} \Pr\left[K_h \text{ shown and Agent selects } i\right] &= z_v(K_h) \cdot \frac{f_i(v)}{\sum_{j \in K_h} f_j(v)} \\ &= \frac{f_i(v)}{\sum_{s=1}^{\frac{1}{\tau}} \sum_{j \in K_s} f_j(v)}, \end{aligned}$$

which is independent of $Z(K_h)$ and the other scores of items in K_j, and so the selection probability of i under z_v depends only on the number of stages in which a menu containing i was added, which is proportional to μ_i. Thus, the probability that the agent chooses item i when a menu is sampled from z_v is:

$$\begin{aligned} \mathop{\mathbb{E}}_{K \sim z_v}\left[\frac{f_i(v)}{\sum_{j \in K} f_j(v)}\right] &= \frac{\mu_i}{\sum_{s=1}^{n} \mu_s} \cdot \frac{1}{\tau} \cdot \frac{f_i(v)}{\sum_{h=1}^{\frac{1}{\tau}} \sum_{j \in K_h} f_j(v)} \\ &= \frac{\frac{x_i}{f_i(v)}}{\sum_{j=1}^{n} \frac{x_j}{f_j(v)}} \cdot \frac{f_i(v)}{\frac{1}{\tau} \sum_{h=1}^{\frac{1}{\tau}} \sum_{j \in K_h} f_j(v)} \\ &= x_i. \end{aligned}$$

As such, we have that any any $x \in \mathrm{IRD}(v, M)$ can be induced by some menu distribution at the memory vector v. The last step of this calculation follows by symmetricity across all items, as the two denominator terms are normalizing to a probability distribution.

For τ bounded away from 0 this requires τ stages, $O(n)$ time for initially computing menu times μ_i, $O(k)$ time per stage, and results in an empirical item distribution where each item's menu time is within τ from μ_i. Further, note that $\sum_{j=1}^{n} \frac{x_i}{f_j(v)} \leq \frac{n}{k^2}$, so if $\tau \leq \frac{\epsilon k^2}{n}$ the runtime is $O(\frac{n}{k\epsilon})$ and the empirical error for each item in the distribution $\hat{x}$ is at most:

$$\begin{aligned} |x_i - \hat{x_i}| &\leq f_i(v) \cdot \frac{n}{k^2} \cdot \tau \\ &\leq \epsilon. \end{aligned}$$

□

A.2 Proof of Theorem 2

Proof. We show that the regret of Algorithm 1 can be bounded by:

$$\mathrm{Reg}_{\mathrm{EIRD}}(\mathcal{A}; T) \leq \mathrm{Reg}_{\mathrm{EIRD}}(\text{RC-FKM}; T) + t_{\text{burn}} + T \cdot \left(\frac{t_{\text{query}}}{t_{\text{query}} + t_{\text{valid}}} + p_{\text{fail}} \right)$$

where $t_{\text{burn}} = T^c$, $t_{\text{query}} = \tilde{O}(T^{1-c/2})$, $t_{\text{valid}} = \tilde{O}(T^{1-c/4})$, and $p_{\text{fail}} = O(1/T)$. From Theorem 1, RC-FKM obtains expected regret at most

$$\mathrm{Reg}_{\mathrm{EIRD}}(\text{RC-FKM}; T) = \tilde{O}\left(T^{3/4} + \epsilon T\right)$$

when the model estimate M^* is sufficiently accurate such that each induced expected item distribution is perturbed by at most ℓ_2 distance ϵ from its target (due to model imprecision), with respect to the true EIRD set. It remains for us to show that the above values of t_{burn}, t_{query}, t_{valid}, and p_{fail} suffice for yielding a model accuracy bound of $\epsilon = \tilde{O}(T^{-c/4})$ in each of the rounds where RC-FKM is used.

Lemma 3 (Burn-In)**.** After $t_{\text{burn}} = T^c$ rounds, we have that $\gamma^{t_{\text{burn}}} \leq \frac{1}{2}$.

Proof. For any $T^c \geq 1$, we have:

$$\gamma^{t_{\text{burn}}} \leq \left(1 - \frac{1}{T^c}\right)^{(T^c)} \leq \frac{1}{e} \leq \frac{1}{2}.$$

□

Following this burn-in stage, we can bound the maximum possible change to the memory vector in any given window.

Lemma 4 (Bounding Memory Drift)**.** For γ, t, w such that $\gamma^w \geq 1 - 2\beta$ and $\gamma^{t+w-1} \leq \frac{1}{2}$, $d_{TV}(v_t, v_{t+w})$ is at most β.

Proof. We bound the drift to a memory vector which can occur from w rounds:

$$\begin{aligned} v_{t+w,i} &= \frac{\sum_{s=t+1}^{t+w-1} \gamma^{t+w-s-1} \cdot x_i^s + \sum_{s=1}^{t-1} \gamma^{t+w-s-1} \cdot x_i^s}{\sum_{s=1}^{t+w-1} \gamma^{t+w-s-1}} \\ &= \frac{\Delta_w}{\sum_{s=1}^{t+w-1} \gamma^{t+w-s-1}} + \frac{\sum_{s=1}^{t-1} \gamma^{t+w-s-1} \cdot x_i^s}{\sum_{s=1}^{t-1} \gamma^{t-s-1}} \cdot \frac{\sum_{s=1}^{t-1} \gamma^{t-s-1}}{\sum_{s=1}^{t+w-1} \gamma^{t+w-s-1}} \\ &= \frac{\Delta_w}{\sum_{s=1}^{t+w-1} \gamma^{t+w-s-1}} + \gamma^w v_i^t \cdot \frac{\sum_{s=1}^{t-1} \gamma^{t-s-1}}{\sum_{s=1}^{t+w-1} \gamma^{t+w-s-1}} \\ &= \frac{(1-\gamma) \cdot \Delta_w}{1 - \gamma^{t+w-1}} + \frac{\gamma^w (1 - \gamma^{t-1}) \cdot v_i^t}{1 - \gamma^{t+w-1}} \end{aligned}$$

The magnitude of Δ_w is $\frac{1-\gamma^w}{1-\gamma}$ and so $d_{TV}(v_t, v_{t+w})$ is at most $\frac{1-\gamma^w}{1-\gamma^{t+w-1}}$. For γ, t, w such that $\gamma^{t+w-1} \leq \frac{1}{2}$, taking $\gamma^w \geq 1 - 2\beta$ yields a d_{TV} bound of β. □

We can use this to obtain an upper limit on w in terms of c such that this bound holds.

Lemma 5. For $\gamma = 1 - 1/T^c$, $t \geq T^c$, and $w \leq \beta \cdot T^c$, we have that $d_{TV}(v_t, v_{t+w}) \leq \beta$.

Proof. First, note that $2w/T^c \leq 2\beta$. Further, we have:

$$\log\left(\frac{1}{1-2\beta}\right) \geq 2\beta$$

and

$$\begin{aligned} \frac{2w}{T^c} &\geq w \log\left(\frac{1}{1-\frac{1}{T^c}}\right) && \left(\text{for } \frac{1}{T^c} \leq \frac{1}{2}\right) \\ &= \log\left(\frac{1}{\gamma^w}\right). \end{aligned}$$

As such, we have that $\log(\gamma^w) \geq \log(1-2\beta)$ and so $\gamma^w \geq 1 - 2\beta$ yielding the result via Lemma A.2. □

We now describe the subroutine $\texttt{Query}(v_t)$, which is analogous to a routine used in Agarwal and Brown (2022). For $w = t_{\text{query}}$ total rounds:

- Include item 1 in the menu in each round;
- Partition items 2 through n into sets of size $k-1$;
- Include each set in the menu $w' = \frac{w(k-1)}{(n-1)}$ rounds;
- Let $\hat{f}_1(v^*) = 1$, and let $\hat{f}_i(v^*)$ for each $i \neq 1$ be its selection frequency relative to item 1, conditioned on being in the menu;
- Let $\tilde{f}_i(v^*)$ be the scores resulting from normalizing the maximum $\hat{f}_i(v^*)$ to be equal to 1.

For a given item $i > 1$, let P_i be its expected probability of being seen, and let $X = \sum_{j=1}^{w'} X_i$ be the sum of Bernoulli trials for whether or not it was chosen in a given round it was included in the menu. By a Hoeffding bound, we have that

$$\begin{aligned} \Pr[|X - \mathbb{E}[X]| \geq \alpha w'] &\leq 2\exp(-2\alpha^2 w') \\ &\leq \delta/n \end{aligned}$$

given that $w' \geq \frac{1}{2\alpha^2}\log(\frac{2n}{\delta})$, and when this bound holds each item is seen with a frequency within α of P_i. Likewise, this holds for each set of trials for item 1, with all events occurring with probability $1-\delta$. Upon converting from observed frequencies to scores $\tilde{f}_i$ with a maximum of 1, this error is increased by a factor of at most k, as each P_i lies in the range $[\lambda/k, 1]$. Given the possible differences in each set of trials for item 1, as well as the drift of at most β in d_{TV} induced by the changing memory vector, this results in a maximum error of $2(k\alpha + L\beta)$ to each estimate $\tilde{f}_i(v^*)$ from $f_i(v^*)$. Further, for any point inside $B_\rho(v^*)$ this error is additionally increased by at most $L\rho$. As such, throughout each round of the optimization stage (where RC-FKM is used) we have that:

$$\left| f_i(v_t) - \tilde{f}_i(v_t) \right| \leq 2(k\alpha + L\beta) + L\rho$$

When considering the resulting menu distribution for a given target item distribution x_t (computed exactly), the error to each item frequency is increased by at most a factor of λ, as the sum of scores lies in $[\lambda k, k]$ for any menu. This yields a bound of ϵ on the resulting distribution in ℓ_2 distance if we have that

$$\sqrt{n} \cdot (2(k\alpha + L\beta) + L\rho) = \epsilon$$

as we have an ℓ_∞ bound of $\epsilon/\sqrt{n}$. We can efficiently approximate this menu in poly$(n)/\epsilon$ time using the construction from Lemma 1 for $\texttt{MenuDist}(v_t, x_t, M^*)$ to obtain an ℓ_∞ error bound of $\epsilon/\sqrt{n}$, yielding an ℓ_2

bound of at most 2ϵ on the resulting expected item distribution upon sampling a menu. Let $t_{\text{query}} = \beta \cdot T^c$; we then can solve for α as:

$$\alpha = \frac{\sqrt{\log(2n/\delta)/2}}{\beta^{1/2}T^{c/2}}.$$

The distribution error bound is then at most:

$$\epsilon = \frac{2k\sqrt{n\log(2n/\delta)/2}}{\beta^{1/2}T^{c/2}} + (2L\sqrt{n})\beta + L\sqrt{n}\rho.$$

Assuming maximal regret of 1 during each step of the learning stage, we incur additional regret of $T \cdot \frac{t_{\text{query}}}{t_{\text{query}}+t_{\text{valid}}} \leq T \cdot \frac{\beta}{\rho}$, as the memory vector remains inside $B_\rho(v^*)$ for at least $t_{\text{valid}} \geq \rho \cdot T^c$ steps by Lemma 5. By setting $\beta = \rho^2$, and $\rho = T^{-c/4}$ we have

$$\begin{aligned}\epsilon &= \frac{2k\sqrt{n\log(2n/\delta)/2}}{\rho T^{c/2}} + (2L\sqrt{n})\rho^2 + L\sqrt{n}\rho \\ &= \tilde{O}(T^{-c/4}),\end{aligned}$$

as we can take $\delta = O(1/T^2)$ which yields $p_{\text{fail}} = O(1/T)$. This yields an expected regret contribution from model error of $\tilde{O}(T^{1-c/4})$, as well as for the learning stages. Further, the possibility of failure of any event in any learning stage contributes at most $O(1)$ to expected regret. Noting that $\max(T^{1-c/4}, T^c) \geq 4/5$ for any $c \in (0, 1)$ and assuming worst-case regret T^c during the burn-in stage, this gives us the desired bound of:

$$\text{Reg}_{\text{EIRD}}(\mathcal{A}_1; T) = \tilde{O}\left(T^{1-c/4} + T^c\right).$$

□

A.3 Proof of Lemma 2

Proof. From Lemma 1, it suffices to show that the menu time μ_i for any such point x is at most 1. Given that v lies inside $\Delta^\phi(n)$ and that each f_i is pseudo-increasing, we have that

$$\frac{x_i}{((1-\lambda)v_i + \lambda)\sigma} \leq \frac{x_i}{f_i(v)} \leq \frac{\sigma x_i}{(1-\lambda)v_i + \lambda}.$$

Recall that the menu time μ_i for x is given by

$$\mu_i(x) = \frac{k \cdot \frac{x_i}{f_i(v)}}{\sum_{j=1}^n \frac{x_j}{f_j(v)}}.$$

First we show that this numerator is at most $k\sigma$. Let $\alpha = \lambda\phi$; we have that $x_i \leq v_i + \alpha$, and so

$$\frac{x_i}{f_i(v)} \leq \frac{\sigma(v_i + \alpha)}{(1-\lambda)v_i + \lambda}.$$

We have that $v_i + \alpha \leq (1-\lambda)v_i + \lambda = v_i + \lambda(1 - v_i)$, as $\alpha \leq \lambda\phi$, yielding that $k\frac{x_i}{f_i(v)} \leq k\sigma$. Next, we can also bound the denominator as at least $k\sigma$. Observe that:

$$\begin{aligned}\sum_{j=1}^n \frac{x_j}{f_j(v)} &\geq \sum_{j=1}^n \frac{x_i}{((1-\lambda)v_i + \lambda)\sigma} \\ &\geq \sum_{j=1}^n \frac{v_j - \rho}{((1-\lambda)v_i + \lambda)\sigma}\end{aligned}$$

Differentiating with respect to v_j for any term, we have:

$$\begin{aligned}
\frac{d}{dv_j}\frac{v_j-\rho}{((1-\lambda)v_i+\lambda)\sigma} &= \frac{((1-\lambda)v_i+\lambda)\sigma-(v_j-\rho)(1-\lambda)\sigma}{\sigma^2((1-\lambda)v_i+\lambda)^2}\\
&= \frac{\lambda\sigma+\rho(1-\lambda)\sigma}{\sigma^2((1-\lambda)v_i+\lambda)^2}
\end{aligned}$$

which is positive for any v_j, and so we can lower bound these terms using the minimal value for v_j where x_j remains in the smoothed simplex which occurs when $v_j-\rho=\phi/(n-1)$ (this further holds for gaps between x_j and v_j which are less than ρ). As such, the sum of terms is at least:

$$\sum_{j=1}^{n}\frac{x_j}{f_j(v)} \geq \frac{n-1}{n}\cdot\frac{\phi}{((1-\lambda)\left(\frac{\phi}{n-1}+\alpha\right)+\lambda)\sigma}$$

which is at least $k\sigma$ when each of the following is satisfied (for $n\geq 4$):

$$\begin{aligned}
\frac{\phi}{\phi/n-1} &\geq 4k\sigma^2;\\
\frac{\phi}{\alpha} &\geq 4k\sigma^2;\\
\frac{\phi}{\lambda} &\geq 4k\sigma^2.
\end{aligned}$$

The first holds given that $n-1\geq 4k\sigma^2$; the second holds given that $\alpha\leq\lambda\phi\leq\frac{\phi}{4k\sigma^2}$, as $\lambda\leq\frac{1}{4k\sigma^2}$ given that $\phi\leq 1$; the latter holds as $\phi\geq 4k\lambda\sigma^2$. As such, we have that

$$\frac{k\cdot\frac{x_i}{f_i(v)}}{\sum_{j=1}^{n}\frac{x_j}{f_j(v)}}\leq 1,$$

and so $x\in\mathrm{IRD}(v,M)$. □

A.4 Proof of Theorem 3

Proof. Much of the proof proceeds similarly to that for Theorem 2. From the analysis of RC-FKM in Agarwal and Brown (2022), the step size in each round can be taken to be $O(T^{-3/4})$, and so we can calibrate our learning rate to ensure that $x^*\in B_{\lambda\phi}(v^*)$ to allow us to apply Lemma 2. Again, we obtain an error bound of for the menu distribution using our estimated model M^* in place of M of at most

$$|\texttt{MenuDist}(v_t,x^*,M)-\texttt{MenuDist}(v_t,x^*,M^*)|\leq\sqrt{n}\cdot(2(k\alpha+L\beta)+L\rho)=\epsilon$$

provided that $t_{\text{query}}\leq\beta\cdot T^c$ and $t_{\text{step}}\leq\rho\cdot T^c$, where $\alpha=\frac{\sqrt{\log(2n/\delta)/2}}{\beta^{1/2}T^{c/2}}$; further, the resulting average item distribution induced over t_{step} will drift at most an additional $O(\alpha)$ distance due to sampling via another set of Hoeffding bounds, by a symmetric analysis to bounding the menu estimation sampling error (as $t_{\text{query}}<t_{\text{step}}$). Again we can take $p_{\text{fail}}=O(1/T)$, and so the remaining elements contributing to our total regret are:

- the average regret of RC-FKM, running for steps of size t_{step} (for the t_{step}-averaged reward functions);
- the perturbation error $O(\epsilon+\alpha)$ to each step (as handled by RC-FKM);
- the fraction of time spent in learning stages, i.e. $\frac{t_{\text{query}}}{t_{\text{query}}+t_{\text{step}}}$;
- the burn-in T^c.

Taking $t_{\text{query}} = \beta \cdot T^c$ and $t_{\text{step}} = \rho \cdot T^c$, this is bounded by:

$$\begin{aligned}
\text{Reg}_{\Delta^\phi(n)}(\mathcal{A}_2; T) &= \tilde{O}\left(t_{\text{step}} \cdot \left(\frac{T}{t_{\text{step}}} \right)^{3/4} + \left(\frac{T^{1-c/2}}{\beta^{1/2}} + \rho \cdot T \right) + \frac{t_{\text{query}}}{t_{\text{step}}} \cdot T + T^c \right) \\
&= \tilde{O}\left(T^{3/4} \cdot t_{\text{step}}^{1/4} + \frac{T}{t_{\text{query}}^{1/2}} + t_{\text{step}} \cdot T^{1-c} + \frac{t_{\text{query}}}{t_{\text{step}}} \cdot T + T^c \right).
\end{aligned}$$

Letting $\beta = T^{-y}$ and $\rho = T^{-z}$, we have $t_{\text{query}} = T^{c-y}$ and $t_{\text{step}} = T^{c-z}$, which yields:

$$\begin{aligned}
\text{Reg}_{\Delta^\phi(n)}(\mathcal{A}_2; T) &= \tilde{O}\left(T^{3/4+c/4-z/4} + T^{1-c/2+y/2} + T^{1-z} + T^{1-y+z} + T^c \right) \\
&= \tilde{O}\left(T^{1-z/4} + T^{1-c/2+y/2} + T^{1-z} + T^{1-y+z} + T^c \right) \qquad (c < 1) \\
&= \tilde{O}\left(T^{1-c/8} + T^{1-3c/16} + T^{1-c/2} + T^{1-c/8} + T^c \right) \\
&= \tilde{O}\left(T^{1-c/8} + T^c \right)
\end{aligned}$$

upon setting $y = 5c/8$ and $z = c/2$. Note that the values of y and z can be further optimized for any c by solving a small linear system of equations to minimize the maximum exponent. □

A.5 Proof of Theorem 4

Proof. We reduce to an instance of Maximum Independent Set (MIS). The MIS problem is Poly-APX-Hard (see e.g. Arora and Barak (2009)), and so there is no constant factor polynomial time approximation algorithm unless P = NP. Given a graph G with $n - 1$ vertices (which we can assume to have a unique maximum independent set S^*), we will construct a preference model M for n items where the optimal menu distribution (or item distribution contained in its IRD set). Our approach is to assume that item 1 yields a constant reward of 1, with all other items yielding a reward of 0. The objective is then to maximize the probability of item 1 being selected, but $f_1(v)$ will only be maximized when the memory mass not placed on item i is uniformly allocated across items corresponding to the maximum independent set (the other scoring functions can be constant at λ). Let $N = \{2, \dots, n\}$. For any memory vector v, let $V(S) = \sum_{j \in S} v_j$. We describe our preference scoring function for item 1 as a piecewise function, and note that it can be made smooth for sufficiently large L (e.g by linearly interpolating within an ϵ radius), chosen based on the discount factor, without qualitatively affecting our results. The function f_i is given as

- If $V(S) \geq V(N) - v_i - \epsilon$, $v_i \in [v_j - \epsilon, v_j + \epsilon]$ for each $i, j \in S$, and S is an independent set in G, then $f_i = \frac{|S| \cdot (1-\lambda)}{|N|} + \lambda$;
- Else; $f_i = \lambda$.

Here we can see that the optimal item distribution is that which places maximal mass on item 1 (as allowed by f_i) while splitting the remainder of its mass uniformly across the maximum independent set S^*. Further, holding constant the menu distribution which always includes item 1 and chooses the other items uniformly from S^* will converge to a stationary point almost surely for appropriate ϵ, via martingale concentration to ensure that the distribution on the items in S^* does not drift more than ϵ from uniform. The reward obtained by either of these approaches is bounded away from any item distribution or menu stationary distribution which does not satisfy the condition for S^*, and so sublinear regret implies identifying S^* in order to maximize the selection probability of item 1.

We can implement oracle access to this function in polynomial time by first checking if a set exists which satisfies the condition (by removing elements with less than ϵ weight, then checking weight similarity, then checking the graph for independence), then computing f_1, enabling efficient reduction. Suppose there existed an algorithm for obtaining $o(T)$ regret with $T = \text{poly}(n)$ with respect to either the best stationary menu distribution, or the best item distribution. Running this algorithm on our preference model, via an oracle simulation using G, must result in item 1 being chosen with probability approaching that from the optimal item distribution. Observing the frequency of other items played would then uniquely identify S^*, as f_1 must have been maximized in almost every round.

To extend this to hardness for pseudo-increasing models, it suffices to scale the requisite scoring functions such that the frequency of the best item is more than $1 - \phi$, with the remaining probability mass allocated accordingly.

□

B OMITTED PROOFS FOR SECTION 4

B.1 Proof of Theorem 5

Proof. The key element of our analysis is to analyze the convergence of item frequencies during windows of length t_{hold} when a fixed target item i is held constant in the menu. For a given such window of length $t_{\text{hold}} = O(\frac{1}{\alpha^4(1-\gamma)})$, we can ensure that the accumulated reward approaches its expectation under the current reward distribution to within α. As we choose the $k - 1$ smallest weights in memory, the total weight of items in memory other than i is at most $\frac{(k-1)(1-v_i)}{n}$; given a current memory vector v, the probability of selecting item i from a menu K_t is given by:

$$\begin{aligned}\Pr[i^* \text{ selected}] &= \frac{f_i(v)}{\sum_{K_t} f_j(v_i)} \\ &\geq \frac{(1-\lambda)v_i + \lambda}{(1-\lambda)v_i + \lambda + \left(\frac{(1-v_i)(1-\lambda)}{n} + \lambda\right)(k-1)\sigma^2}\end{aligned}$$

by the pseudo-increasing property. Our approach will be to analyze the expectation of $v_{t,i}$ over time, with $E_t = \mathbb{E}[v_{t,i}]$, and show that it approaches $1 - \phi$. A challenge is that, given a current expectation, there are many possible allocations of probabilities to values of $v_{t,i}$ which yield E_t. A second derivative test shows that the above probability function is concave for positive v_i; note that both the numerator and denominator are positive and increasing in v_i, and that the numerator is always smaller but grows faster in v_i. As such, we can apply Jensen's inequality and restrict our consideration to the extremal case where the expectation E_t is entirely composed of trials in which $v_{t,i} = 0$ or $v_{t,i} = 1$, which indeed occurs $\gamma = 0$. We can define P_0 and P_1 as lower bounds on selection probabilities for each case:

$$\begin{aligned}\Pr[i^* \text{ selected}|v_{t,i} = 1] &\geq \frac{1}{1 + (k-1)\lambda\sigma^2} \\ &\geq 1 - k\lambda\sigma^2 \\ &:= P_1;\end{aligned}$$

$$\begin{aligned}\Pr[i^* \text{ selected}|v_{t,i} = 0] &\geq \frac{\lambda}{\lambda + (\frac{1-\lambda}{n} + \lambda)(k-1)\sigma^2} \\ &= \frac{1}{1 + \left(\frac{1-\lambda}{n\lambda} + 1\right)(k-1)\sigma^2} \\ &\geq \frac{1}{1 + (k\sigma^2 + 1)(k-1)\sigma^2} \\ &\geq \frac{1}{2\sigma^2 k^4} \\ &:= P_0.\end{aligned}$$

As such, we have that

$$\begin{aligned}E_{t+1} &= \mathbb{E}[v_{t+1}|E_t] \\ &\geq (1-\gamma)\left(E_t \cdot \Pr[i^* \text{ selected}|v_{t,i} = 1] + (1 - E_t) \cdot \Pr[i^* \text{ selected}|v_{t,i} = 0]\right) + \gamma E_t \\ &\geq (1-\gamma)\left(E_t \cdot P_1 + (1 - E_t) \cdot P_0\right) + \gamma E_t.\end{aligned}$$

We can solve for E_t^* such that $E_{t+1} = E_t$, i.e. where $E_t \cdot P_1 + (1 - E_t) \cdot P_0 = E_t$, as:

$$\begin{aligned} E_t^* &= \frac{1}{1 + 2\lambda\sigma^6 k^3} \\ &\geq 1 - 2\sigma^6 k^3 \lambda, \end{aligned}$$

and further for a value E_t^α such that $E_{t+1} \geq (1 - \gamma)(E_t + \alpha) + \gamma E_t$ as:

$$E_t^\alpha = E_t^* - 2\sigma^4 k^2 \alpha.$$

Note that the rate of growth of E_{t+1} is decreasing in t, and eventually reaches a fixed point; given that the rate of growth of E_t is linear in α when within $O(\alpha)$ of E_t^*, the cumulative number of rounds required to reach $E_t^* - O(\alpha)$ is at most $O(\frac{1}{\alpha(1-\gamma)})$. If we continue after for $O(\frac{1}{\alpha^2(1-\gamma)})$ rounds, these first rounds contribute at most α to the total summed expectation for the fraction of rounds in which item i is selected is at least $E_t^* - \alpha$; the fraction of each other item played also quicly approaches $\frac{1-E_t^*}{n-1}$ in expectation.

Treat each such batch of $O(\frac{1}{\alpha^2(1-\gamma)})$ rounds as a trial, and repeat for $\tilde{O}(1/\alpha^2)$ trials, resulting in a total of $t_{\text{hold}} = O(\frac{1}{\alpha^4(1-\gamma)})$ steps. We can treat each trial as independent, as resetting the memory vector to some lower value of v_i can only decrease expected reward. By a Hoeffding bound, we have that the reward is within α from the expectation under the current distribution and the "arm" of the ϕ-smoothed simplex corresponding to i. To complete the analysis, observe that our total regret (using the $\tilde{O}(T^{1/2})$ bound for EXP3) is given by:

$$\begin{aligned} \text{Reg}_{\Delta^\phi(n)}(\mathcal{A}_3; T) &= \tilde{O}(t_{\text{hold}} \cdot \left(\frac{T}{t_{\text{hold}}}\right)^{1/2} + \alpha T) \\ &= \tilde{O}(\frac{T^{1/2}}{\alpha^2} + \alpha T) \\ &= \tilde{O}(T^{5/6}) \end{aligned}$$

upon setting $\alpha = O(T^{-1/6})$.

□

B.2 Proof of Theorem 6

Proof. We consider a set of models and reward functions where item 1 yields a reward of 1 in each round, with all other items yielding a reward of 0. For $\gamma = \frac{1}{2^c}$ for some constant $c > 1$, note that the weight of any step in memory is larger than the sum of weights of all preceding steps, and thus a memory vector v_t exactly encodes the history of item selections for the first $t - 1$ rounds. Let $h = \log_{2^c}(n)$; For t sufficiently larger than h, the sum of weights of steps 1 through $t - h$ is $\Theta(1/n)$. We will consider states s which are subsets of the space of memory vectors corresponding to each possible history truncated to the previous h steps, and which are bounded apart by a distance of at least $O(1/n)$. We will abuse notation and represent each memory vector v_t as its rounded state s_t. The behavior of the memory model is constant over each state, and smoothly interpolates between states; the model can be defined arbitrarily for infeasible memory vectors to satisfy $L = \text{poly}(n)$ Lipschitzness. Our process for generating $\mathcal{M}$ is as follows:

- Let $k = n/2$;
- Let $\lambda = \frac{1}{n-k+1}$;
- For each $s \in [n]^h$, let G_s be a set of $k - 2$ items sampled uniformly at random from $\{2, \ldots, n\}$;
- Let $f_i(s) = \lambda$ if $i = 1$ or $i \in G_s$, and $f_i(s) = 1$ otherwise.

Observe that the optimal strategy π^* at s is to include item 1, each item in G_s, and any arbitrary final item. Note that each of the $k - 1$ items with score λ is selected with probability

$$\begin{aligned} \Pr[i \text{ chosen} | f_i(s) = \lambda, \pi^* \text{ played}] &= \frac{\lambda}{1 + (k-1)\lambda} \\ &= \frac{1}{n}, \end{aligned}$$

and so the expected reward per round is 1 as well. Note that π^* is consistent with a menu distribution which chooses the final item (after 1 and G_s) uniformly at random, which generates the uniform distribution. As such, the uniform distribution lies inside EIRD (t is straightforward to define scores for infeasible memory vectors such that feasibility holds for any $v \in \Delta(n)$). We can also see that any menu inconsistent with π^* has expected reward at most:

$$\frac{\lambda}{2+(k-2)\lambda} = \frac{\frac{1}{n-k+1}}{\frac{2n-2k+2}{n-k+1} + \frac{k-2}{n-k+1}} \\ = \frac{3}{4n},$$

as some item not in G_s must be included. To lower bound the regret of any algorithm, consider an arbitrary time t and history of item selections. By time t, at most distinct states have been observed thus far. Consider the following cases:

- The algorithm plays a menu consistent with π^* in every step from t to $t+h-1$;
- The algorithm plays a menu inconsistent with π^* at some step from t to $t+h-1$.

Suppose t is less than $T = \frac{1}{2} \cdot (\frac{n}{2})^h = O(n^{\log(n)})$. In the former case, there is a uniform distribution over $\frac{n}{2}$ items chosen by the agent at each round, and so the maximum probability of any given state is at most $\frac{n}{2})^h$, and so a new state w that a Given that the set G_s is generated independently at random for each state, an algorithm has no information about G_s for unvisited states, and thus cannot improve expected reward beyond that obtained by choosing a random hypothesis for G_s, which incurs $O(\frac{1}{n})$ regret at round $t+h$. In the latter case, the step in which a menu inconsistent with π^* is played additionally incurs $O(\frac{1}{n})$ regret. Each event occurs once at least once in expectation every h rounds while $t < T$, and thus any algorithm must have $O(\frac{1}{nh})$ expected regret per round up to T.

□

C OMITTED PROOFS FOR SECTION 5

C.1 Proof of Theorem 7

Proof. The first algorithm is a direct application of the well-known EXP3 algorithm. The EXP3 algorithm achieves $\tilde{O}(\sqrt{TK})$ regret for the adversarial multi-armed bandit instance with K arms and time-horizon T. The proof of the $\tilde{O}(\sqrt{Tn^k})$ result follows from the fact that $K = O(n^k)$ in our setting and the loss for items can be transformed into the loss for menus.

The second algorithm is divided into two phases: (1) score estimation, (2) regret minimization. We perform score estimation for $O(n \cdot T^{2/3})$ rounds. We chose any arbitrary item i and partition the remaining items into $(n-1)/(k-1)$ menus of size $k-1$. We then add item i to each of these menus. We play each of these menus for $T^{2/3}$ times and for each item j obtain unbiased estimates $\hat{w}_{ji}$ of $w_{ji} = s_j/s_i$ based on the choice probabilities. Using the standard Chernoff bound we can show that these estimates $\hat{w}_{ji}$ are within $\text{poly}(n) \cdot T^{-1/3}$ of the true scores w_{ji}. Since the preference model is scale invariant, we can scale these estimates such that these estimates are within $[\lambda, 1]$, and use them as a proxy for the true scores s_j's. Once we have these estimates for the true scores, we can use the argument in Section 3.1 for the construction of distributions over menus that induces an IRD target distribution over items. We then run the SCRIBLE algorithm where the arm set is the set of items. Given a distribution p_t over arms at round t, we can efficiently compute a distribution over menus that can induce the same distribution over items using Lemma 1. Given that there is $\text{poly}(n) \cdot T^{-1/3}$ error in each score estimate, we can effectively play a distribution $\hat{p}_t$ such that $\|p_t - \hat{p}_t\|_1 \leq \text{poly}(n) \cdot T^{-1/3}$. The total regret is then the regret of SCRIBLE which is $\tilde{O}(\sqrt{Tn})$ in addition to the regret due to playing $\hat{p}_t$ instead of p_t which is $\tilde{O}(\text{poly}(n) \cdot T^{-1/3})$ in each trial. This gives a total regret upper bound of $\tilde{O}(\text{poly}(n) \cdot T^{2/3})$.

□