Diversified Recommendations for Agents with Adaptive Preferences

Arpit Agarwal*
Department of Computer Science
Columbia University
New York, NY 10027
arpit.agarwal@columbia.edu

William Brown†
Department of Computer Science
Columbia University
New York, NY 10027
w.brown@columbia.edu

Abstract

When an Agent visits a platform recommending a menu of content to select from, their choice of item depends not only on immutable preferences, but also on their prior engagements with the platform. The Recommender's primary objective is typically to encourage content consumption which optimizes some reward, such as ad revenue, but they often additionally aim to ensure that a sufficiently wide variety of content is consumed by the Agent over time. We formalize this problem as an adversarial bandit task. At each step, the Recommender presents a menu of k (out of n) items to the Agent, who selects one item in the menu according to their unknown *preference model*, which maps their history of past items to relative selection probabilities. The Recommender then observes the Agent's selected item and receives bandit feedback of the item's (adversarial) reward. In addition to optimizing reward from the selected items at each step, the Recommender must also ensure that the total distribution of chosen items has sufficiently high entropy.

We define a class of preference models which are *locally learnable*, i.e. behavior over the entire domain can be estimated by only observing behavior in a small region; this includes models representable by bounded-degree polynomials as well as functions with a sparse Fourier basis. For this class, we give an algorithm for the Recommender which obtains $\tilde{O}(T^{3/4})$ regret against all item distributions satisfying two conditions: they are sufficiently diversified, and they are *instantaneously realizable* at any history by some distribution over menus. We show that these conditions are closely connected: all sufficiently high-entropy distributions are instantaneously realizable at any history of selected items. We also give a set of negative results justifying our assumptions, in the form of a runtime lower bound for non-local learning and linear regret lower bounds for alternate benchmarks.

1 Introduction

Suppose you manage an online platform that repeatedly provides menus of recommended content to visitors, such as sets of videos to watch or items to purchase, aiming to display options which agents will engage favorably with and yield you high rewards (in the form of ad revenue, watch time, purchases, or other metrics). In many settings, the preferences of agents are not fixed *a priori*, but rather can *change* as a function of their consumption patterns—the deeper one goes down a content "rabbit hole", the further one might be likely to keep going. This "rabbit hole" effect can lead to (unforeseen) loss of revenue for the platform, as advertisers may later decide that they are not willing to pay as much for this "rabbit hole" content as they would for other content. The scope of negative

*http://www.columbia.edu/ aa4931/
†wibrown.github.io

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

effects emerging from these feedback loops is large, ranging from the emergence of "echo chambers" [14] and rapid political polarization [21] to increased homogeneity which can decrease agent utility [9], amplify bias [18], or drive content providers to leave the platform [20]. These are harms which many platforms aim to avoid, both for their own sake and out of broader societal concerns.

Hence, the evolving preferences of the Agent can be directly at odds with the Recommender's objectives of maximizing revenue and ensuring diverse consumption patterns in this *dynamic* environment. Our goal is to study such tensions between the interaction of these two players: the Recommender that recommends menus based on past choices of the Agent so as to maximize its reward (subject to diversity constraints), and the Agent whose preferences evolve as a function of past recommendations.

To this end, we consider a stylized setting where the Recommender is tasked with providing a *menu* of k recommended items (out of n total) every round to an Agent for T sequential rounds. In each round, the Agent observes the menu, then selects one of the items according to their *preference model* M, which the Recommender does not know in advance. The preference model M takes as input the Agent's *memory vector* v, which is the normalized histogram of their past chosen items, and assigns relative selection probabilities to each item. The selected item at each round results in a reward for the Recommender, specified by an adversarial sequence of reward vectors, which the Recommender receives as bandit feedback, in addition to observing which item was selected. The Recommender must choose a sequence of menus to maximize their reward (or minimize regret), subject to a *diversity constraint*, expressed as a minimum entropy for the empirical item distribution.

However, any regret minimization problem is incomplete without an appropriate benchmark for comparing the performance of a learner. An entropy constraint alone is insufficient to define a such benchmark. Due to intricacies of the Agent's preference model, there may be item distributions which are impossible to induce under any sequence of menus (e.g. they may strongly dislike the most profitable content). Adding to the challenge is the fact that the preference model is initially unknown and must be *learned*, and the set of item distributions which are *instantaneously realizable* by sampling a menu from some distribution can shift each round as well. Several immediate proposals are infeasible: it is impossible to obtain sublinear regret against the best fixed menu distribution, or even against the best item distribution realizable from the uniform memory vector. We propose a natural benchmark for which regret minimization becomes possible: the set of item distributions which are *everywhere* instantaneously realizable (the $\texttt{EIRD}(M)$ set), i.e. item distributions such that, at any memory vector, there is always *some* menu distribution which induces them. We show that this set is also closely related to entropy constraints: when M is sufficiently *dispersed* (a condition on the minimum selection probability for each item), $\texttt{EIRD}(M)$ contains all sufficiently high-entropy distributions, and so regret minimization can occur over the entire high-entropy set.

1.1 Our Results

We give an algorithm which, for a minimum entropy set H_c and preference model M, allows the Recommender to obtain $\tilde{O}(T^{3/4})$ regret against the best distribution in the intersection of H_c and $\texttt{EIRD}(M)$, provided that M satisfies λ-*dispersion* and belongs to a class $\mathcal{M}$ which is *locally learnable*. A λ-dispersed preference model M assigns a preference score of at least $\lambda > 0$ to every item, ensuring a minimum positive probability of selection to each item in a menu. Dispersion is a natural assumption, given our restriction to $\texttt{EIRD}(M)$, as items which only have positive selection probability in part of the domain cannot be induced everywhere. The local learnability condition for a model class enforces that the behavior of any particular model can be predicted by observing behavior only in a small region. This is essentially necessary to have any hope of model estimation in this setting: we show that if learning a class from *exact* queries requires making queries to many points which are pairwise well-separated, exponentially many rounds are required to implement query learning. Despite this restriction, we show that several rich classes of preference models are indeed locally learnable, including those where preference scoring functions are expressed by bounded-degree multivariate polynomials, or by univariate functions with a sparse Fourier basis.

Our algorithm is explicitly separated into learning and optimization stages. The sole objective for the learning stage is to solve the *outer problem*: recover an accurate hypothesis for the preference model. We select sequences of menus which move the Agent's memory vector to various points near the uniform distribution, enabling us to implement local learning and produce a model hypothesis $\hat{M}$. We then shift our focus to the *inner problem* for the Recommender, which is natural to view as a bandit linear optimization problem over the set of distributions in consideration, as we can use $\hat{M}$ to identify a distribution of menus which generates a particular item distribution. However, representing the $\texttt{EIRD}(\hat{M})$ set explicitly is impractical, as the functions which generate feasible sets from the history can be highly non-convex. Instead, we operate over the potentially larger set where intersections are taken only over the sets $\texttt{IRD}(v, \hat{M})$ of instantaneously realizable distributions we have observed thus far. This precludes us from using off-the-shelf bandit linear optimization algorithms as a black box, as they typically require the decision set to be specified in advance. We introduce a modification of the FKM algorithm [13], RC-FKM, which can operate over contracting decision sets, and additionally can account for the imprecision in $\hat{M}$ when generating menu distributions. This enables the Recommender to guide the Agent to minimize regret on their behalf via the sequence of menus they present.

1.2 Summary of Contributions

Briefly, our main contributions are:

1. We formulate the dynamic interaction between a Recommender and an Agent as an adversarial bandit task. We show that no algorithm can obtain $o(T)$ regret against the best menu distribution, or against the best item distribution in the IRD set of uniform vector. We then consider $\texttt{EIRD}(M)$ and argue that it is a natural benchmark for regret as it also contains all sufficiently high entropy distributions over items.
2. We define a class of *locally learnable* functions, which are functions that can be learned only using samples from a small neighborhood. We show a number of rich classes of functions where this is possible, and further we show that any class which is *not* locally learnable cannot be learned quickly by any algorithm which fits a hypothesis using queries.
3. We give an algorithm for the Recommender that achieves $\widetilde{O}(T^{3/4})$ regret against $\texttt{EIRD}(M)$ for locally learnable classes of preference models that are λ-dispersed, which implements local learning to obtain a sufficiently accurate hypothesis for use in optimizing menu distributions. As a component of this, we develop a new algorithm for bandit linear optimization which can operate over contracting decision sets, and which can account for bounded adversarial imprecision in the played action.

Overall, by considering this stylized setting we are able to provide several insights into the dynamic interaction between an Agent and a Recommender. While our algorithm is a useful tool for a Recommender who is already committed to providing diversified recommendations, we also view our results as presenting an intrinsic argument for incorporating such constraints. When preferences adapt over time, and Agents may be prone to venturing down content "rabbit holes", restricting attention to recommendation patterns which are not too concentrated on small sets of items can in fact make the regret minimization problem tractable by discouraging consumption patterns which may be difficult to draw the Agent back from. This suggests a synergy between the goal of regret minimization and showing diverse content to the user.

1.3 Related Work

Feedback loops in user preferences have received significant attention in the recommender systems literature, particularly for models with multiple agents which make use of collaborative filtering methods, and with explicit adaptivity models which are less flexible than those we consider [7, 9, 18, 26, 20]. Within the online learning literature, our formalization bears some resemblance to bandit problems where multiple arms can be pulled simultaneously, which have received much recent attention [28, 27, 8, 3]. Our results also share similarities with work on optimization from revealed preferences, where a mapping to a nested convex problem must be learned [25, 12]; with the performative prediction literature, where actions induce a distribution shift which impacts instantaneous reward potential [22, 17]; and more broadly, with repeated game problems against adaptive agents [5, 11, 10]. Further related work is discussed in Appendix A.

1.4 Organization

In Section 2, we introduce our setting and key definitions, analyze the local learnability of several classes of preference models, and give a series of negative and structural results. In Section 3 we introduce a bandit linear optimization algorithm for contracting sets, which we use as a subroutine for our main algorithm in Section 4. We discuss the intuition for our proof techniques throughout, with full proofs deferred to the appendix.

2 Model and Preliminaries

The central object of our setting is the *preference model* of the Agent, which dictates their relative item preferences based on their selection history and expresses their adaptivity over time.

Definition 1 (Preference Models)**.** *A preference model is a mapping $M : \Delta(n) \to [0, 1]^n$ which maps memory vectors v to a preference score vector $s_v = M(v)$.*

We assume that any input $v \notin \Delta(n)$ to M (such as the empty history at $t = 1$) results in the uniform score vector where $M(v)_i = 1$ for all i. A constraint on our sequence of interactions with the Agent is that the resulting item distribution must have sufficiently high entropy.

Definition 2 (Diversity Constraints)**.** *A diversity constraint $H_c \subset \Delta(n)$ is the convex set containing all item distributions $v \in \Delta(n)$ with entropy at least c, i.e. v is in H_c if and only if:*

$$H(v) = -\sum_{i=1}^{n} v_i \log(v_i) \geq c.$$

We say that a constraint H_c is ϵ*-satisfied* by a distribution v if we have that $\min_{x \in H_c} d_{TV}(x, v) \leq \epsilon$, where d_{TV} is the total variation distance between probability distributions.

Our algorithmic results can be extended to any convex constraint set which contains a small region around the uniform distribution, but we focus on entropy constraints as they are quite natural and have interesting connections to our setting which we consider in Section 2.3.

2.1 Recommendation Menus for Adaptive Agents

An instance of our problem consists of an item set $N = [n]$, a menu size k, a preference model M for the Agent, a constraint H_c, a horizon length of T rounds, and a sequence of linear reward functions $\rho_1, \ldots, \rho_T$ for the Recommender. In each round $t \in \{1, \ldots, T\}$:

- The Recommender chooses a menu $K_t \subset N$ with $|K_t| = k$.
- The Agent chooses item $i \in K_t$ with probability

  $$p_{K_t,v_t,i_t} = \frac{s_{v_t,i_t}}{\sum_{j \in K_t} s_{v_t,j}}$$

  and updates its memory vector to the normalized histogram

  $$v_{t+1} = \frac{e_i}{t+1} + \frac{t \cdot v_t}{t+1},$$

  where e_i is the ith standard unit vector.
- The Recommender observes receives reward $\rho_t(e_i)$ for the chosen item.

The goal of the Recommender is to maximize their reward over T rounds subject to v_T satisfying H_c. It might seem to the reader that the Recommender can 'manipulate' the Agent to achieve any preference score vector over time; however, this is not true as many score vectors might not be achievable depending on the preference model.

2.2 Realizability Conditions for Item Distributions

For any memory vector v, we define the feasible set of item choice distributions for Agent in the current round, each generated by a distribution over menus which the Recommender samples from.

Definition 3 (Instantaneously-Realizable Distributions at v)**.** *Let $p_{K,v} \in \Delta(n)$ be the item distribution selected by an Agent presented with menu K at memory vector v, given by:*

$$p_{K,v,i} = \frac{s_{v,i}}{\sum_{j \in K} s_{v,j}}.$$

The set of instantaneously-realizable distributions at v is given by:

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

For any $x \in \mathtt{IRD}(v, M)$, any menu distribution $z \in \Delta(\binom{n}{k})$ specifying a convex combination of menu score vectors $p_{K,v}$ which sum to x will generate the item distribution x upon sampling.

One might hope to match the performance of the best menu distribution, or perhaps the best realizable item distribution from the uniform vector. Unfortunately, neither of these are possible.

Theorem 1. *There is no algorithm which can obtain $o(T)$ regret against the best item distribution in the* `IRD` *set for the uniform vector, or against the best menu distribution in $\Delta\left(\binom{n}{k}\right)$, even when the preference model is known exactly and is expressible by univariate linear functions.*

We give a separate construction for each claim, with the full proof deferred to Appendix . The first is a case where the optimal distribution from the uniform vector cannot be played every round, as it draws the the memory vector into `IRD` sets where the reward opportunities are suboptimal. The second considers menu distributions where obtaining their late-round performance requires committing early to an irreversible course of action. Instead, our benchmark will be the set of distributions which are realizable from *any* memory vector.

Definition 4 (Everywhere Instantaneously-Realizable Distributions)**.** *For a preference model M, the set of everywhere instantaneously-realizable distributions is given by:*

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

This is the set of distributions $x \in \Delta(n)$ such that from any memory vector v, there is some menu distribution z such that sampling menus from d induces a choice distribution of x for the agent.

Note that the set $\mathtt{EIRD}(M)$ is convex, as each $\mathtt{IRD}(v, M)$ is convex by construction.

2.3 Conditions for Preference Models

The algorithm we present in Section 4 requires two key conditions for a class of preference models: each model in the class must be *dispersed*, and the class must be *locally learnable*. This enforces that the Agent is always willing to select every item in the menu they see with some positive probability, and that the behavior at any memory vector can be estimated by observing behavior in a small region.

Definition 5 (Dispersion)**.** *A preference model M is λ-dispersed if $s_{v,i} \geq \lambda$ for all $v \in \Delta(n)$ and for all i, i.e. items always have a score of at least λ at any memory vector.*

The dispersion condition plays an important role in the analysis of our algorithm by enabling efficient exploration, but it additionally coincides with diversity constraints in appropriate regimes.

Theorem 2 (High-Entropy Containment in `EIRD`)**.** *Consider the diversity constraint H_c for $c = \log(n) - \gamma$, and let $\tau \geq \exp(-\gamma)$. Let M be a λ-dispersed preference model with $\lambda \geq \frac{k^2 \exp(\gamma/\tau)}{n}$. For any vector $v \in H_c$, there is a vector $v' \in \mathtt{EIRD}(M)$ such that $d_{TV}(v, v')$ is at most $O(\tau)$.*

The key step here, proved in Appendix B.2, is that $\mathtt{EIRD}(M)$ contains the uniform distribution over any large subset of items, and taking mixtures of these can approximate any high-entropy distribution.

Next, for a class of models to be locally learnable, one must be able to accurately estimate a model's preference scores everywhere when only given access to samples in an arbitrarily small region.

Definition 6 (Local Learnability)**.** *Let $\mathcal{M}$ be a class of preference models, and let*

$$\mathtt{EIRD}(\mathcal{M}) = \bigcap_{M \in \mathcal{M}} \mathtt{EIRD}(M).$$

Let v^* be a point in $\mathtt{EIRD}(\mathcal{M})$, and V_α be the set of points within distance α from v^*, for α such that $V_\alpha \subseteq \mathtt{EIRD}(\mathcal{M})$. $\mathcal{M}$ is h-locally learnable if there is some v^* and an algorithm $\mathcal{A}$ which, for any $M \in \mathcal{M}$ and any $\alpha > 0$, given query access to normalized score estimates $\hat{s}_v$ where $\|\hat{s}_v - M(v)/M^*_v\|_\infty \leq \beta$ for any $v \in V_\alpha$ (where $M^*_v = \sum_i M(v)_i$) and for some β, can produce a hypothesis model $\hat{M}$ such that $\left\|\hat{M}(x)/\hat{M}^*_x - M(x)/M^*_x\right\| \leq \epsilon$ for any $x \in \Delta(n)$ and $\epsilon = \Omega(\beta)$.

The local learnability condition, while covering many natural examples shown in Section 2.4, is indeed somewhat restrictive. In particular, it is not difficult to see that classes of piecewise functions, such as neural networks with ReLU activations, are not locally learnable. However, this appears to be essentially a necessary assumption for efficient learning, given the cumulative nature of memory in our setting. We show a runtime lower bound for any algorithm that hopes to learn an estimate $\hat{M}$ for the preference model M via *queries*. Even a Recommender who can force the Agent to pick a particular item each round, and exactly query the preference model for free at the current memory vector, may require exponentially many rounds to learn $\hat{M}$ if the points it must query are far apart.

Theorem 3 (Query Learning Lower Bound)**.** *Suppose the Recommender can force the Agent to select any item at each step t, and can query $M(v_t)$ at the current memory vector v_t. Let $\mathcal{A}_S$ be an algorithm which produces a hypothesis $\hat{M}$ by receiving queries $M(v)$ for each $v \in S$. For points v and v', let $d_{\max}(v, v') = \max_i v_i - v'_i$. Then, any sequence of item selections and queries by the Recommender requires at least*

$$T \geq \min_{\sigma \in \pi(S)} \prod_{i=1}^{|S|-1} (1 + d_{\max}(\sigma(i), \sigma(i+1)))$$

rounds to run $\mathcal{A}(S)$, where $\pi(S)$ is the set of permutations over S and $\sigma(i)$ is the ith item in σ.

We prove this in Appendix B.3. Notably, this implies that if S contains m points which, for any pair (v, v') have both $d_{\max}(v, v') \geq \gamma$ and $d_{\max}(v', v) \geq \gamma$, at least $O\left((1+\gamma)^m\right)$ rounds are required.

2.4 Locally Learnable Preference Models

There are several interesting examples of model classes which are indeed locally learnable, which we prove in Appendix C. In general, our approach is to query a grid of points inside the radius α ball around the uniform vector, estimate each function's parameters and show that the propagation of over the entire domain is bounded. Note that the normalizing constants for each query we observe may differ; for univariate functions, we can handle this by only moving a subset of values at a time, allowing for renormalization. For multivariate polynomials, we directly estimate the *ratio* of each the n functions in isolation to their sum, which requires a somewhat more involved error propagation analysis. Each local learning result we prove involves an algorithm which makes queries near the uniform vector. We later show in Lemma 4 that taking $\lambda \geq k^2/n$ suffices to ensure that these queries can indeed be implemented via an appropriate sequence of menu distributions for any M in such a class.

2.4.1 Bounded-Degree Univariate Polynomials

Let $\mathcal{M}_{BUP}$ be the class of *bounded-degree univariate polynomial* preference models where:

- For each i , $M(v)_i = f_i(v_i)$, where f_i is a degree-d univariate polynomial which takes values in $[\lambda, 1]$ over the range $[0, 1]$ for some constant $\lambda > 0$.

Univariateness captures cases where relative preferences for an item depend only on the weight of that item in the agent's memory, i.e. there are no substitute or complement effects between items.

Lemma 1. *$\mathcal{M}_{BUP}$ is $O(d)$-locally learnable by an algorithm $\mathcal{A}_{BUP}$ with $\beta \leq O(\epsilon\lambda^2 \cdot (\frac{\alpha}{nd})^d)$.*

2.4.2 Bounded-Degree Multivariate Polynomials

Let $\mathcal{M}_{BMP}$ be the class of *bounded-degree multivariate polynomial* preference models where:

- For each i , $M(v)_i = f_i(v)$, where f_i is a degree-d polynomial which takes values in $[\lambda, 1]$ over $\Delta(n)$ for some constant $\lambda > 0$.

This captures quite a large variety of adaptivity patterns for preferences, including those where the score for each item can depend on observed frequencies for many other items simultaneously. In particular, it can capture relatively intricate "rabbit hole" effects, in which some subsets of items are mutually self-reinforcing, and where their selection can discourage future selection of other subsets.

Lemma 2. *$\mathcal{M}_{BMP}$ is $O(n^d)$-locally learnable by an algorithm $\mathcal{A}_{BMP}$ with $\beta \leq O(\frac{\epsilon}{\text{poly}(nd/(\alpha\lambda))^d})$.*

2.4.3 Univariate Functions with Sparse Fourier Representations

We can also allow for classes of functions where the minimum allowable α depends on some parameter. Functions with sparse Fourier representations are such an example, and naturally capture settings where preferences are somewhat cyclical, such as when an Agent goes through "phases" of preferring some type of content for a limited window. We say that a function $f : \mathbb{R} \to \mathbb{R}$ is ℓ*-sparse* if $f(x) = \sum_{i=1}^{\ell} \xi_i e^{2\pi \mathbf{i} \eta_i x}$ where $\eta_i \in [-F, F]$ denotes the i-th frequency and ξ_i denotes the corresponding magnitude. We say that an ℓ-sparse function f is $\hat{\alpha}$-separable when $\min_{i \neq j} |\eta_i - \eta_j| > \hat{\alpha}$. Let $\mathcal{M}_{SFR}(\hat{\alpha})$ be the class of univariate *sparse Fourier representation* preference models where:

- For each i , $M(v)_i = f_i(v_i)$, where f_i is a univariate ℓ-sparse and $\hat{\alpha}$-separable function which, over $[0, 1]$, is L-Lipschitz and takes values in $[\lambda, 1]$ for some constant $\lambda > 0$.

Lemma 3. *$\mathcal{M}_{SFR}(\hat{\alpha})$ is $\widetilde{O}(n\ell)$-locally learnable by an algorithm $\mathcal{A}_{SFR}$ with $\beta \leq O(\frac{\epsilon\lambda\alpha}{\sqrt{n\ell}})$ and any $\alpha \geq \tilde{\Omega}(1/\hat{\alpha})$.*

3 Bandit Linear Optimization with Contracting Sets

The inner problem for the Recommender can be viewed as a bandit linear optimization problem over $H_c \cap \texttt{EIRD}(M)$. However, representing $\texttt{EIRD}(M)$ is challenging even if we know M exactly, as it involves an intersection over infinitely many sets (generated by a possibly non-convex function), and a net approximation would involve exponential dependence on n. Instead, our approach will be to operate over the larger set $H_c \cap (\bigcap_t \texttt{IRD}(v_t, M))$ for the memory vectors v_t we have seen thus far, where representing each IRD has exponential dependence only on k (from enumerating all menus).

The tradeoff is that we can no longer directly use off-the-shelf bandit linear optimization algorithms for a known and fixed decision set such as FKM [13] or SCRIBLE [1] as a subroutine, as our decision set is contracting each round. We introduce an algorithm for bandit linear optimization, a modification of the FKM algorithm we call Robust Contracting FKM (RC-FKM), which handles this issue by projecting to our estimate of the contracted decision set at each step. Additionally, RC-FKM can handle the imprecision resulting from our model estimation step, which can be represented by small adversarial perturbations to the action vector in each round; we modify the sampling rule to ensure that our target action remains in the true decision set even when perturbations are present. We prove the regret bound for RC-FKM in Appendix D.

Algorithm 1 (Robust Contracting FKM).

Input: sequence of contracting convex decision sets $\mathcal{K}_1, \ldots \mathcal{K}_T$ containing $\mathbf{0}$, perturbation vectors $\xi_1, \ldots, \xi_T$ where $\|\xi_t\| \leq \epsilon$, parameters δ, η.
Set $x_1 = \mathbf{0}$
for $t = 1$ to T **do**
  Draw $u_t \in \mathbb{S}_1$ uniformly at random, set $y_t = x_t + \delta u_t + \xi_t$
  Play y_t, observe and incur loss $\phi_t \in [0, 1]$, where $\mathbb{E}[\phi_t] = f_t(y_t)$
  Let $g_t = \frac{n}{\delta} \phi_t u_t$
  Let $\mathcal{K}_{t+1,\delta,\epsilon} = \{x | \frac{r}{r-\delta-\epsilon} x_t \in \mathcal{K}_{t+1}\}$
  Update $x_{t+1} = \Pi_{\mathcal{K}_{t+1,\delta,\epsilon}}[x_t - \eta g_t]$
end for

Theorem 4 (Regret Bound for Algorithm 1)**.** *For a sequence of G-Lipschitz linear losses $f_1, \ldots, f_T$ and a contracting sequence of domains $\mathcal{K}_1, \ldots, \mathcal{K}_T$ (with $\mathcal{K}_j \subseteq \mathcal{K}_i$ for $j > i$, each with diameter at most D, and where a ball of radius $r > \delta + \epsilon$ around $\mathbf{0}$ is contained in $\mathcal{K}_T$), and adversarially chosen unobserved vectors $\xi_1, \ldots, \xi_T$ with $\|\xi_t\| \leq \epsilon$ which perturb the chosen action at each step, with parameters $\eta = \frac{D}{nT^{3/4}}$ and $\delta = \frac{1}{T^{1/4}}$, Algorithm 1 obtains the expected regret bound*

$$\sum_{t=1}^{T} \mathbb{E}[\phi_t] - \min_{x \in \mathcal{K}_T} \sum_{t=1}^{T} f_t(x) \leq nGDT^{3/4} + \frac{GDT^{3/4}}{r} + \frac{2\epsilon GDT}{r}.$$

4 Recommendations for Adaptive Agents

Our main algorithm begins with an explicit learning phase, after which we conduct regret minimization, and at a high level works as follows:

- First, we learn an estimate of the preference model $\hat{M}$ in $t_0 = \tilde{O}(T^{3/4})$ steps by implementing local learning with a set of points close to the uniform memory vector, which suffices to ensure high accuracy of our representation with respect to M.
- For the remaining $T - t_0$ steps, we implement RC-FKM by using the learned model $\hat{M}$ at each step to solve for a menu distribution which generates the desired item distribution from the current memory vector, then contracting the decision set based on the memory update.

Theorem 5 (Regret Bound for Algorithm 2)**.** *Algorithm 2 obtains regret bounded by*

$$\text{Regret}_{C \cap \texttt{EIRD}(M)}(T) \leq \tilde{O}\left(t_0 + nGT^{3/4} + \frac{(\delta+\epsilon)GT}{r} + \epsilon GT\right) = \tilde{O}(T^{3/4})$$

where $t_0 = \tilde{O}(1/\epsilon^3)$, $r = O(k^2/n)$, and $\epsilon, \delta = O(r \cdot T^{-1/4})$, and results in an empirical distribution such that H_c is $O(\epsilon)$-satisfied with probability at least $1 - O(T^{-1/4})$.

Algorithm 2 A no-regret recommendation algorithm for adaptive agents.

Input: Item set $[n]$, menu size k, Agent with λ-dispersed memory model M for $\lambda \geq \frac{k^2}{n}$, where M belongs to an S-locally learnable class $\mathcal{M}$, diversity constraint H_c, horizon T, G-Lipschitz linear losses $\rho_i, \ldots, \rho_T$.
Let $t_{\text{pad}} = \tilde{\Theta}(1/\epsilon^3)$
Let $t_{\text{move}} = \tilde{\Theta}(1/\epsilon^3)$
Let $t_{\text{query}} = \tilde{\Theta}(1/\epsilon^2)$
Let $\alpha = \Theta(\frac{k}{n^2 S})$
Get set of S points in the α-ball around uniform vector x_U to query from $\mathcal{A}_{\mathcal{M}}$
Let $t_0 = t_{\text{pad}} + S(2 \cdot t_{\text{move}} + t_{\text{query}})$
Run `UniformPad` for t_{pad} rounds
for x_i in S **do**
    Run `MoveTo`(x_i) for t_{move} rounds
    Run `Query`(x_i) for t_{query} rounds, observe result $\hat{q}(x_i)$
    Run `MoveTo`(x_U) for t_{move} rounds
end for
Estimate model $\hat{M}$ using $\mathcal{A}_{\mathcal{M}}$ for $\beta = \Theta(\epsilon)$
Let v_{t_0} be the empirical item distribution of the first items t_0 items
Let $\mathcal{K}_{t_0} = H_c$ (in $n-1$ dimensions, with $x_{t,n} = 1 - \sum_{i=1}^{n-1} x_{t,i}$, and s.t. x_U translates to $\mathbf{0}$)
Initialize RC-FKM to run for $T^* = T - t_0 - 1$ rounds with $r = O(k^2/n)$, $\delta, \epsilon = \frac{r}{T^{*1/4}}$
for $t = t_0 + 1$ to T **do**
    Let x_t be the point chosen by RC-FKM
    Use `PlayDist`(x_t) to compute menu distribution z_t
    Sample $K_t \sim z_t$, show K_t to Agent
    Observe Agent's chosen item i_t and reward $\rho_t(e_{i_t})$
    Update RC-FKM with $\rho_t(e_{i_t})$
    Let $v_t = \frac{t-1}{t} v_{t-1} + \frac{1}{t} \cdot e_{i_t}$
    Update the decision set to $\mathcal{K}_{t+1} = \mathcal{K}_t \cap \texttt{IRD}(v_t, \hat{M})$
end for

4.1 Structure of `EIRD`(M)

The key tool which enables us to implement local learning is a construction for generating any point near the uniform via an adaptive sequence of menu distributions, provided λ is sufficiently large.

Lemma 4. *For any λ-dispersed M where $\lambda \geq \frac{k^2}{n}$, `EIRD`$(M)$ contains all points $x \in \Delta(N)$ satisfying*

$$\|x - x_U\|_\infty \leq \frac{k-1}{n(n-1)},$$

where x_U is the uniform $\frac{1}{n}$ vector.

We give an algorithmic variant of this lemma which is used directly by Algorithm 2, as well as a variant for uniform distributions over smaller subsets as λ grows, which we use to prove Theorem 2.

4.2 Subroutines

Our algorithm makes use of a number of subroutines for navigating the memory space, model learning, and implementing RC-FKM. We state their key ideas here, with full details deferred to Appendix E.

`UniformPad`:

- In each round, include the k items with smallest counts, breaking ties randomly.

`MoveTo`(x):

- Apply the same approach from `UniformPad` to the difference between the current histogram and x.

`Query`(x):

- Play a sequence of $O(n/k)$ partially overlapping menus which cover all items, holding each constant long enough for concentration, and compute relative probabilities of each item.

`PlayDist`(x):

- Given an item distribution x, we solve a linear program to compute a menu distribution z_x using $\hat{M}(v)$ which induces x when a menu is sampled and the Agent selects an item.

The intuition behind our learning stage is that each call to `Query`(x) can be accurately estimated by bounding the "drift" in the memory vector while sampling occurs, as the number of samples per query is small compared to the history thus far. Each call to `MoveTo`(x) for a point within the α-ball can be implemented by generating an empirical distribution corresponding to a point in `EIRD`(M) for sufficiently many rounds.

The resulting model estimate $\hat{M}$ yields score estimates which are accurate for any memory vector. To run RC-FKM, we translate to an $n-1$ dimensional simplex representation, and construct a menu distribution to implement any action x_t via a linear program (`PlayDist`(x)). The robustness guarantee for RC-FKM ensures that the loss resulting from imprecision in $\hat{M}$ is bounded, and further ensures that the resulting expected distribution remains inside H_c (and that H_c is approximately satisfied with high probability by the empirical distribution). We contract our decision set in each step with the current space `IRD`($v_t, \hat{M}$), which will always contain `EIRD`($\hat{M}$), the best point in which is competitive with the best point in `EIRD`(M).

5 Conclusion and Future Work

Our work formalizes a bandit setting for investigating online recommendation problems where agents' preferences can adapt over time and provides a number of key initial results which highlight the importance of diversity in recommendations, including lower bounds for more "ambitious" regret benchmarks, and a no-regret algorithm for the `EIRD` set benchmark, which can coincide with the high-entropy set under appropriate conditions. Our results showcase a tradeoff between the space of strategies one considers and the ability to minimize regret. Crucially, our lower bound constructions illustrate that we cannot hope to optimize over the set of recommendation patterns which may send agents down "rabbit holes" that drastically alter their preferences, whereas it is indeed feasible to optimize over the space of sufficiently diversified recommendations.

There are several interesting directions which remain open for future investigation, including additional characterizations of the `EIRD` set, discovering more examples or applications for local learnability, identifying the optimal rate of regret or dependence on other parameters, settings involving multiple agents with correlated preferences, and consideration of alternate models of agent behavior which circumvent the difficulties posed by uniform memory.

References

[1] Jacob D. Abernethy, Elad Hazan, and Alexander Rakhlin. Interior-point methods for full-information and bandit online learning. *IEEE Transactions on Information Theory*, 58(7):4164–4175, 2012.

[2] Arpit Agarwal, Shivani Agarwal, Sepehr Assadi, and Sanjeev Khanna. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In Satyen Kale and Ohad Shamir, editors, *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 39–75. PMLR, 07–10 Jul 2017.

[3] 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.

[4] Jason Altschuler and Kunal Talwar. Online learning over a finite action set with limited switching. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 1569–1573. PMLR, 06–09 Jul 2018.

[5] Mark Braverman, Jieming Mao, Jon Schneider, and S. Matthew Weinberg. Selling to a no-regret buyer. *CoRR*, abs/1711.09176, 2017.

[6] Sébastien Bubeck, Bo'az Klartag, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. *Chasing Nested Convex Bodies Nearly Optimally*, pages 1496–1508. 2020.

[7] Keith Burghardt and Kristina Lerman. Emergent instabilities in algorithmic feedback loops. *CoRR*, abs/2201.07203, 2022.

[8] Róbert Busa-Fekete, Eyke Hüllermeier, and Adil El Mesaoudi-Paul. Preference-based online learning with dueling bandits: A survey. *CoRR*, abs/1807.11398, 2018.

[9] Allison J. B. Chaney, Brandon M. Stewart, and Barbara E. Engelhardt. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. *CoRR*, abs/1710.11214, 2017.

[10] Constantinos Daskalakis and Vasilis Syrgkanis. Learning in auctions: Regret is hard, envy is easy. *CoRR*, abs/1511.01411, 2015.

[11] Yuan Deng, Jon Schneider, and Balasubramanian Sivan. Strategizing against no-regret learners. *CoRR*, abs/1909.13861, 2019.

[12] Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. Strategic classification from revealed preferences. *CoRR*, abs/1710.07887, 2017.

[13] 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.

[14] Yingqiang Ge, Shuya Zhao, Honglu Zhou, Changhua Pei, Fei Sun, Wenwu Ou, and Yongfeng Zhang. Understanding echo chambers in e-commerce recommender systems. *CoRR*, abs/2007.02474, 2020.

[15] Taha Hassan. Trust and trustworthiness in social recommender systems. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, page 529–532, New York, NY, USA, 2019. Association for Computing Machinery.

[16] Elad Hazan. Introduction to online convex optimization. *CoRR*, abs/1909.05207, 2019.

[17] Meena Jagadeesan, Tijana Zrnic, and Celestine Mendler-Dünner. Regret minimization with performative feedback. *CoRR*, abs/2202.00628, 2022.

[18] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. Feedback loop and bias amplification in recommender systems. *CoRR*, abs/2007.13019, 2020.

[19] Celestine Mendler-Dünner, Juan Perdomo, Tijana Zrnic, and Moritz Hardt. Stochastic optimization for performative prediction. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4929–4939. Curran Associates, Inc., 2020.

[20] Martin Mladenov, Elliot Creager, Omer Ben-Porat, Kevin Swersky, Richard S. Zemel, and Craig Boutilier. Optimizing long-term social welfare in recommender systems: A constrained matching approach. *CoRR*, abs/2008.00104, 2020.

[21] Derek O'Callaghan, Derek Greene, Maura Conway, Joe Carthy, and Pádraig Cunningham. Down the (white) rabbit hole: The extreme right and online recommender systems. *Social Science Computer Review*, 33(4):459–478, 2015.

[22] Juan C. Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. Performative prediction. *CoRR*, abs/2002.06673, 2020.

[23] Eric Price and Zhao Song. A robust sparse fourier transform in the continuous setting. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 583–600. IEEE Computer Society, 2015.

[24] Anshuka Rangi, Massimo Franceschetti, and Long Tran-Thanh. Sequential choice bandits with feedback for personalizing users' experience, 2021.

[25] Aaron Roth, Jonathan R. Ullman, and Zhiwei Steven Wu. Watch and learn: Optimizing from revealed preferences feedback. *CoRR*, abs/1504.01033, 2015.

[26] Ayan Sinha, David F. Gleich, and Karthik Ramani. Deconvolving feedback loops in recommender systems. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3251–3259, Red Hook, NY, USA, 2016. Curran Associates Inc.

[27] 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. JCSS Special Issue: Cloud Computing 2011.

[28] 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.

[29] Xing Zhao, Ziwei Zhu, and James Caverlee. Rabbit holes and taste distortion: Distribution-aware recommendation with evolving interests. In *Proceedings of the Web Conference 2021*, WWW '21, page 888–899, New York, NY, USA, 2021. Association for Computing Machinery.

[30] Tijana Zrnic, Eric V. Mazumdar, S. Shankar Sastry, and Michael I. Jordan. Who leads and who follows in strategic classification? In *NeurIPS*, 2021.

A Further Related Work

A.1 Empirical Investigation Of Recommendation Feedback Loops

A substantial body of evidence has emerged in recent years indicating that recommendation systems can create feedback loops which drive negative social consequences. [21] observed that users accessing videos with extreme political views are likely to get caught in an "ideological bubble" in just a few clicks, and [15] explore the role of recommendation algorithms in creating distrust and amplifying political polarization on social media platforms. By investigating a real-world e-commerce dataset, [14] study the way in which recommendation systems drive agents' self-reinforcing preferences and lead them into "echo chambers" where they are separated from observing a diversity of content. [29] conduct a meta-analysis over many datasets which focuses specifically on the "rabbit hole" problem by means of exploring "taste distortion" of agents who observe recommendations which are more extreme than their current preferences. Such results motivate investigating these dynamics from game-theoretic and learning-theoretic foundations.

A.2 Modeling Feedback Loops in Recommendation Systems

A number of recent works from the recommendation systems literature have explored the role of collaborative filtering algorithms for various models of agent behavior, aiming to understand how feedback loops in recommendation patterns emerge, the harms they cause, and how they can be corrected [7, 26]. A common theme is homogenization of recommendations across a *population* of users, which can lead to exacerbation of biased utility distributions for minority groups [18], long-run utility degradation [9], and a lack of traffic to smaller content providers which results in them being driven to exit the platform [20]. Our work indirectly addresses this phenomenon by encouraging diverse recommendations, but our primary focus is from the perspective of a single agent, who may be led down a "rabbit hole" by an algorithm which optimizes for their immediate engagement.

A.3 Dueling Bandits

The "dueling bandits" problem, initially proposed as a model for similar recommendation systems challenges [28, 27], and which has been generalized for sets larger than two [3, 24], considers a similar setting in which bandit optimization is conducted with respect to the preference model of an agent, occasionally represented via an explicit parametric form. Here, one presents a set of choices to an agent, then receives only *ordinal* feedback about the relative rewards of the choices, and must optimize recommendations with regret measured against the best individual choice. In contrast to our setting, these works consider preferences which are fully determined *a priori*, and do not change as a function of item history or exhibit preference feedback loops.

A.4 Online Stackelberg Problems

A number of works in recent years explore online problems where an agent responds to the decision-maker's actions, influencing their reward. The performative prediction setting, introduced in [22], captures settings in which a deployed classifier results in changes to the distribution itself, in turn affecting performance. This work has been extended to handle stochastic feedback [19] and notably, to a no-regret variant [17] which involves learning mapping between classifiers and distribution shifts, which bears some conceptual similarities to our procedure for locally learning an agent's preference model. The "revealed preferences" literature involves a similar requirement of learning a mapping between actions and agent choices [25, 12]. Some features of our setting resemble elements of other well-studied online problems, including the restricted exploration ability for limited switching problems (e.g. [4]), and the contracting target set for chasing nested convex bodies (e.g. [6]).

A.5 Strategizing Against Adaptive Agents

Some recent work has begun to explore the problem of designing optimal strategies in a repeated game against agents who *adapt* their strategies over time using a no-regret algorithm. In auction problems, [5] study the extent to which an auction designer can extract value from bidders who use different kinds of no-regret algorithms. More generally, [11] connect this line of investigation to Stackelberg equilibria for normal-form games. In strategic classification problems, [30] study the behavior when using a learning rate which is either much faster or much slower than that of the agents which one aims to classify, and draw connections to equilibrium concepts as well. Our work extends this notion of strategizing against adaptive agents to recommendation settings, with novel formulations of adaptivity and regret to suit the problem's constraints.

B Omitted Proofs for Sections 2

B.1 Proof of Linear Regret Lower Bounds (Theorem 1)

We give a separate lower bound construction for the uniform IRD item distribution benchmark and the menu distribution benchmark, yielding the theorem.

Lemma 5. *There is no algorithm which can obtain $o(T)$ regret against the best item distribution in the* IRD *set for the uniform vector, even when the preference model is known exactly and is expressible by univariate linear functions.*

Proof. First we give an example for which obtaining $o(T)$ regret against $\texttt{IRD}(v, M)$ for the uniform vector v_U is impossible. Consider the memory model M where:

- $M(v)_1 = \lambda + 0.5 + \frac{n}{n-1} \cdot (v_1 - \frac{1}{n}) \cdot (0.5 - \lambda)$;
- $M(v)_2 = \lambda + 0.5(1 - v_1 + \frac{1}{n})$;
- $M(v)_i = 0.5 + \lambda$ for $i > 2$.

Observe that at the uniform distribution where $v_1 = \frac{1}{n}$, all items have a score of $0.5 + \lambda$. If $v_1 = 1$, we have that:

- $M(v)_1 = 1$, and
- $M(v)_2 = \lambda + \frac{0.5}{n}$.

If $v_1 = 0$, we have that:

- $M(v)_1 = \lambda + 0.5 - \frac{0.5-\lambda}{n-1}$, and
- $M(v)_2 = \lambda + 0.5 \cdot (1 + \frac{1}{n})$

As scores linearly interpolate between these endpoints for any v_1, M is λ-dispersed, and scores lie in $[\lambda, 1]$. Let $k = 2$. Consider reward functions which give reward $\alpha > 0$ for item 1 in each round up to $t^* = T/2$, giving reward 0 to each other item; after t^*, a reward of $\beta > 0$ is given for item 2 while the rest receive a reward 0. The distribution which assigns probability $1/2$ each to item 1 and 2, with all other items having probability 0, is contained in $\texttt{IRD}(v_U, M)$, as one can simply play the menu with both items. This distribution yields a total expected reward of

$$R_v = \frac{\alpha t^*}{2} + \frac{\beta t^*}{2}$$

over T steps. Consider the performance of any algorithm $\mathcal{A}$ which results in item 1 being selected with an empirical probability p over the first t^* rounds. At $t = t^*$, we have $v_{t^*,1} = p$; its total reward over the first t^* rounds is $\alpha p t^*$. For sufficiently large n and small λ, the score for item 2 is approximated by $M(v)_2 = 0.5(1 - p)$ up to any desired accuracy. In future rounds $t \geq t^*$, the value $v_{t,s1}$ is at least $\frac{pt^*}{t}$, and so the score for item 2 is at most

$$M(v)_2 = 0.5(1 - \frac{pt^*}{t}).$$

Each other item has a score of at least 0.5, yielding an upper bound on the probability that item 2 can be selected even if it is always in the menu, as well as a maximum expected per-round reward of

$$R_t = \beta \cdot \left(\frac{0.5(1 - \frac{pt^*}{t})}{1 - 0.5\frac{pt^*}{t}} \right).$$

At time $T = 2t^*$, the instantaneous reward is at most

$$R_T = \beta \cdot \left(\frac{2-p}{4-p}\right),$$

which is also a per-round upper bound for each $t \geq t^*$. This bounds the total reward for $\mathcal{A}$ by

$$R_{\mathcal{A}} = \alpha p t^* + \beta t^* \cdot \frac{2-p}{4-p}.$$

We can now show that for any p, there exists a β such that $R_v - R_{\mathcal{A}} = \Theta(T)$. For any $p \leq \frac{1}{3}$, we have

$$R_{\mathcal{A}} \leq \frac{\alpha t^*}{3} + \frac{\beta t^*}{2},$$

and for any $p > \frac{1}{3}$ we have:

$$R_{\mathcal{A}} \leq \alpha t^* + \frac{5\beta t^*}{11}.$$

In the first case, we immediately have $R_v - R_{\mathcal{A}} \geq T\alpha/6$ for any β. In the second case, let $\beta \geq 22\alpha$. We then have:

$$\begin{aligned} R_v - R_{\mathcal{A}} &\geq \frac{\beta t^*}{22} - \frac{\alpha t^*}{2} \\ &\geq T\alpha/4. \end{aligned}$$

The value of β can be determined adversarially, and so there is no algorithm $\mathcal{A}$ which can obtain $o(T)$ regret against $\texttt{IRD}(v, M)$.

□

Next we show a similar impossibility result for regret minimization with respect to the set of all menu distributions.

Lemma 6. *There is no algorithm which can obtain $o(T)$ regret against the best menu distribution in $\Delta\left(\binom{n}{k}\right)$, even when the preference model is known exactly and is expressible by univariate linear functions.*

Proof. Let M be the λ-dispersed memory model where the functions for items (a, b, c), and every other item i, are given by:

- $M(v)_a = \lambda + (1-\epsilon)(1 - v_b)$;
- $M(v)_b = \lambda + (1-\epsilon)v_b$;
- $M(v)_c = \lambda + (1-\epsilon)v_c$;
- $M(v)_i = \lambda + (1-\epsilon)(1 - v_b)$ for $i \notin \{a, b, c\}$;

for some $\lambda > 0$ and $\epsilon > \lambda$. Let $k = 2$. Consider a sequence of rewards $\{f_t\}$ which yields reward α to items (a, b) for each round $t \leq t^*$ and 0 to the rest, then in each step after t^*, yields a reward of β for item c, a reward of 0 for item b, and reward of $-\beta$ for every other item. Note the total expected reward for the following distributions:

$$\begin{aligned} R_{(a,b)}(T) &= \alpha t^* - \beta(T - t^*)/2; \\ R_{(b,c)}(T) &= \alpha t^*/2 + \beta(T - t^*)/2; \end{aligned}$$

The bound for $R_{(a,b)}(t^*)$ follows from symmetricity of the resulting stationary distribution, given by the unique solution $v_a = 0.5$ to the recurrence:

$$v_b = \frac{\lambda + (1-\epsilon)v_b}{2\lambda + (1-\epsilon)}$$

which is approached in expectation for large T regardless of initial conditions for any constant λ. Symmetricity also results in balanced expectations for each item in $R_{(b,c)}$.

Consider the distribution p_{t^*} played by an algorithm $\mathcal{A}$ over the first t^* rounds, where t^* is large enough to ensure concentration. If $p_{t^*,a} + p_{t^*,b} \leq 1 - \delta$ for some constant δ, then for $\beta = 0$ the algorithm has regret $\delta \alpha t^* = \Theta(T)$ for any $t^* = \Theta(T)$. Further, if regret is not bounded, the menu (a, b) must be played in nearly every round, as other item placed in the menu has positive selection probability. As such, the empirical probability of b must be close to $1/2$.

After t^*, the algorithm cannot obtain a per-round utility which matches that of (b, c) up to δ until a round t where either:

$$\frac{\lambda + (1-\epsilon)p_{t,c}}{2\lambda + (1-\epsilon)(p_{t,b} + p_{t,c})} \geq 1/2 - \delta$$

or

$$\frac{\lambda + (1-\epsilon)p_{t,c}}{2\lambda + (1-\epsilon)(1 - p_{t,b} + p_{t,c})} \geq 1/2 - \delta,$$

which requires the total number of rounds in which c is chosen to approach $t^*/2 - C \cdot \delta t^*$, where C is a constant depending on ϵ and λ. Let $T = 3t^*/2$, and so this cannot happen for small enough constant δ, resulting in a regret of $\delta\beta T/3 - \alpha T/3$ with respect to (b, c), which is $\Theta(T)$ when $\delta\beta > \alpha$.

□

B.2 Proof of High-Entropy Containment of EIRD (Theorem 2)

Proof. By Lemma 13, for a λ-dispersed preference model M with $\lambda \geq \frac{Ck^2}{n}$, any uniform distribution over n/C items lies inside $\texttt{EIRD}(M)$. We make use of a lemma from [2], which we restate here.

Lemma 7 (Lemma 8 in [2])**.** *For a random variable A over $[n]$ with $H(A) \geq \log n - \gamma$, there is a set of $\ell + 1 = O(\gamma/\tau^3))$ distributions ψ_i for $i \in \{0, \dots, n\}$ over a partition of the support of A which can be mixed together to generate A, where ψ_0 has weight $O(\tau)$, and where for each $i \geq 1$:*

1. $\log |\text{supp}(\psi_i)| \geq \log n - \gamma/\tau$.
2. *ψ_i is within total variation distance $O(\tau)$ from the uniform distribution on its support.*

Using this, we can explicitly lower bound the support of each ψ_i:

$$\begin{aligned} \log |\text{supp}(\psi_i)| &\geq \log(n) - \gamma/\tau \\ &= \log(n) - \log(\exp(\gamma/\tau)) \\ &= \log\left(\frac{n}{\exp(\gamma/\tau)}\right). \end{aligned}$$

As such:

$$|\text{supp}(\psi_i)| \geq \frac{n}{\exp(\gamma/\tau)}.$$

Each uniform distribution over $\text{supp}(\psi_i)$ lies inside $\texttt{EIRD}(M)$ for $\lambda \geq \frac{Ck^2}{n}$, provided that $C \geq \exp(\gamma/\tau)$. The $O(\tau)$ bound on total variation distance is preserved under mixture, as well as when redistributing the mass of ψ_0 arbitrarily amongst the uniform distributions.

□

B.3 Proof of Query Learning Runtime Lower Bound (Theorem 3)

Proof. For any permutation σ, we can lower bound the steps required to move between any two vectors adjacent in the ordering in terms of $d_{\max}$ and the number of rounds elapsed thus far.

Lemma 8. *Consider two vectors v and v', where v is the current empirical item distribution after t steps. Reaching an empirical distribution of v' requires at least $t \cdot d_{\max}(v, v')$ additional steps.*

Proof. Let x be the histogram representation of v with total mass t, and let $j^* = \arg\max_j v_j - v'_j$, where $v_j - v'_j = d_{\max}(v, v')$. Let $x' = t' \cdot v'$ be the histogram representation of v' with total mass t', such that $x_{j^*} = x'_{j^*}$. Note that t' is the smallest total mass (or total number of rounds) where a histogram can normalize to v', as any subsequent histogram must have $x'_{j^*} \geq x_{j^*}$. As such, we must have that $t' \cdot v'_{j^*} \geq t \cdot v_{j^*}$, implying that:

$$\begin{aligned}\frac{t'}{t} &\geq \frac{v_{j^*}}{v'_{j^*}} \\ &= \frac{v'_{j^*} + d_{\max}(v, v')}{v'_{j^*}} \\ &\leq 1 + d_{\max}(v, v').\end{aligned}$$

□

At least one round is required to reach the first vector in a permutation, and we can use the above lemma to lower-bound the rounds between any adjacent vectors in the ordering. Taking the minimum over all permutations gives us the result. □

C Proofs of Local Learnability for Section 2.4

Each proof gives a learning algorithm which operates in a ball around the uniform vector, which is contained in $\texttt{EIRD}(M)$ whenever $\lambda \geq \frac{k^2}{n}$ by Lemma 4.

C.1 Proof of Univariate Polynomial Local Learnability

Proof. Query the uniform vector v_U where each $v_i = \frac{1}{n}$. Let $Z = \frac{\sqrt{nd/6}}{\alpha}$. Consider three sets each of $d/2$ memory vectors where the items with indices satisfying $i \mod 3 = z$ each have memory values $\frac{1}{n} + \frac{j}{Z}$, items satisfying $i \mod 3 = z + 1$ have values $\frac{1}{n} - \frac{j}{Z}$, and the remainder have $\frac{1}{n}$ (for $z \in \{0, 1, 2\}$, and for $1 \leq j \leq \frac{d}{2}$). All such vectors lie in V_α, as $2n/3 \cdot (d/(2Z))^2 \leq \alpha^2$. Query each of the $3d/2$ vectors. For each query, let R_v be the sum of all scores of the items held at $\frac{1}{n}$, divided by the sum of those same items' scores in the uniform query. Divide all scores by R_v. Let R_v^* be the be the corresponding ratio of these sums of scores under $\{f_i\}$; each sum is within $[\frac{\lambda}{3}, 1]$ at each vector, and the sums of observed scores have additive error at most $n\beta/3$. As such, R_v has additive error at most $\frac{2n\beta}{\lambda}$ from R_v^*. This gives us estimates for $d + 1$ points of $\hat{f}_i(x_j) = \hat{y}_j$ for each polynomial, up to some universal scaling factor. We can express this d-degree polynomial $\hat{f}_i$ via Lagrange interpolation:

$$L_{d,j}(x) = \prod_{k \neq j}^{d} \frac{x - x_k}{x_j - x_k};$$

$$\hat{f}_i(x) = \sum_{j=0}^{d} \hat{y}_j L_{n,j}.$$

Note that $\sum_i \hat{f}_i(v_U) = 1$ as the scores coincide exactly with our query results at the uniform vector. To analyze the representation error, let $\{f_i^*\}$ be the set of true polynomials f_i rescaled to sum to 1 at the uniform vector; this involves dividing by a factor $S \in [n\lambda, n]$, and produces identical scores at every point. Consider the difference $\left|\hat{y}_j - y_j^*\right|$ for each $y_j^* = f_i^*(x_j)$. The query error for $\hat{y}_j$ prior to rescaling is at most β; rescaling by R_v^* would increase this to at most $3\beta/\lambda$, which is amplified to at most

$$\left|\hat{y}_j - y_j^*\right| \leq \frac{3\beta}{\lambda} + \frac{2n\beta}{\lambda} \leq \frac{3n\beta}{\lambda}$$

as each query score is at most 1 (and our setting is trivial for $n \leq 2$). The magnitude of each of the $d+1$ Lagrange terms can be bounded by:

$$\begin{aligned} |L_{d,j}(x)| &\leq \prod_{j=1}^{d/2} \frac{Z^2}{j^2} \\ &\leq \frac{Z^d}{((d/2)!)^2} \end{aligned}$$

for any $x \in [0,1]$, and so for any function $\hat{f}_i(x)$ we can bound its distance from $f_i^*(x)$ by:

$$\begin{aligned} \left| f_i^*(x) - \hat{f}_i(x) \right| &= (d+1) \cdot \frac{3n\beta Z^d}{\lambda((d/2)!)^2} \\ &\leq \frac{(d+1)3n\beta Z^d}{\lambda 2^{d/2}}. \end{aligned}$$

This holds simultaneously for each $\hat{f}_i$ which, using the fact that the true ratio is at least λ/n and the per-function bound applies to each denominator term, gives us a total bound on the score estimates we generate:

$$\begin{aligned} \left| \frac{\hat{f}_i(x)}{\sum_{j=1}^{x} \hat{f}_j(x)} - \frac{f_i(x)}{\sum_{j=1} f_j(x)} \right| &\leq \left(1 + \frac{(d+1)3n\beta Z^d}{\lambda 2^{d/2}} \right) \cdot \frac{(d+1)3n^3\beta Z^d}{\lambda^2 2^{d/2}} \\ &\leq \frac{7n^3 d\beta Z^d}{\lambda^2 2^{d/2}} \\ &\leq \frac{3 \cdot (6nd)^{d/2+2}\beta}{\alpha^d \lambda^2 2^{d/2}} \\ &= \frac{(3nd)^{d/2+2}\beta}{\alpha^d \lambda^2}. \end{aligned}$$

Taking $\beta \leq \frac{\epsilon \alpha^d \lambda^2}{(3nd)^{d/2+2}}$ gives us an absolute error of at most ϵ per item score, satisfying a Euclidean bound of ϵ from any true score vector $M(w)/M_w^*$ for our hypothesis $\hat{M}(v) = \{\hat{f}_i(v_i) : i \in [n]\}$. □

C.2 Proof of Multivariate Polynomial Local Learnability

Lemma 9. *$\mathcal{M}_{BMP}$ is $O(n^d)$-locally learnable by an algorithm $\mathcal{A}_{BMP}$ with $\beta \leq O(\frac{\epsilon}{\text{poly}(nd/(\alpha\lambda))^d})$.*

Proof. Consider the set of polynomials where each v_n term is reparameterized as $1 - \sum_{i=1}^{n-1} v_i$, then translated so that the uniform vector appears at the origin (i.e. with $x_n = -\sum_{i=1}^{n-1} x_i$). Our approach will be to learn a representation of each polynomial normalized their sum, which is unique up to a universal scaling factor. Let f_i^* be the representation of f_i in this translation. Consider the $B = \sum_{j=0}^{d} \binom{n-1}{j}$-dimensional basis where each variable in a vector x corresponds to a monomial of variables in v with degree at most d, with the domain constrained to ensure mutual consistency between monomials; observe that f_i^* is a linear function in this basis. Let $q_i(x) = M(v)_i/M_v^*$ denote the normalized score for item i at v, where v translates to x in the new basis. For we any x we have:

$$\frac{f_i^*(x)}{\sum_{j=1}^{n} f_j^*(x)} = q_i(x),$$

and let $\hat{q}_i(x)$ denote the analogous perturbed query result, both of which sum to 1 over each i. We are done if we can estimate the vector $q(x)$ up to distance ϵ for any x.

With $f_i^*(x) = \langle a, x \rangle + a_0$ and $\sum_{i=1}^{n} f_i^*(x) = \langle b, x \rangle + b_0$, our strategy will be to estimate the ratio of each coefficient with b_0, for each f_i^*, in increasing order of degree. While our parameterization does not include item n, we will explicitly estimate b separate from each a, which we can then use to estimate $f_n^*(x) = \langle b, x \rangle + b_0 - \sum_{i=1}^{n-1} f_i^*(x)$. For a monomial m of degree j, we can estimate its coefficient for all f_i^* simultaneously by moving the values for variables it contains simultaneously from the $\mathbf{0}$ vector, and viewing the restriction to its subset monomials as a univariate polynomial of degree j. We will use a single query to the $\mathbf{0}$ vector, and $2j+1$ additional queries for each degree-j monomial (which can be used for learning that monomial's coefficient in all f_i^* simultaneously), resulting in a total query count of:

$$1 + \sum_{j=1}^{d} (2j+1) \cdot \binom{n}{j} = 1 + \sum_{j=1}^{d} (2j+1) \frac{n!}{j!(n-j)!}$$
$$= O(n^d).$$

Querying $\mathbf{0}$ gives us an estimate for each additive term:

$$\frac{\hat{a}_0^i}{b_0} = \hat{q}_i(\mathbf{0})$$

which sum to 1 over all items (and we will take $\hat{b}_0 = 1$). We now describe our strategy for computing higher-order coefficients in terms of lower-order coefficients under the assumption of *exact* queries, after which we conduct error propagation analysis. For a monomial m of degree j, let $x_{(h,m)}$ be the point where $x_{(h,m),i} = hZ$ if an item i belongs to m and 0 otherwise, with higher degree terms satisfying the basis constraints (i.e. $(hZ)^3$ for a degree-3 subset of m, and $(hZ)^j$ for m), which also results in the term for a monomial containing any item not in m being set to zero. Query $x_{(h,m)}$ for $2j+1$ distinct values h in $\{\pm 1, \dots, \pm(j+1)\}$. For $Z = \alpha/(2d(d+1))$ all queries lie in the α-ball, as the ℓ_1 norm of the positive coefficients, as well as the negative offset for item n, are both bounded by $\alpha/2$ in the original simplex basis. Suppose all coefficients up to degree $j-1$ are known. The result of such a query is equivalent to:

$$q(x_{(h,m)}) = \frac{a_m z^j + f_a(z)}{b_m z^j + f_b(z)}$$

where f_a and f_b are $(j-1)$-degree univariate polynomials, where each coefficient of some degree $k \le j-1$ is expressed by summing the coefficients for degree-k monomials which are subsets of m, for a and b respectively. Rearranging, we have:

$$a_m = q_i(x_{(h,m)}) \cdot b_m + \frac{q(x_{(h,m)}) \cdot f_b(z) - f_a(z)}{z^j}.$$

This gives us a linear relationship between a_m and b_m in terms of known quantities after just one query where $z \neq 0$. Suppose we could make *exact* queries; if we observe two distinct linear relationships, we can solve for a_m and b_m. If each query gives us the same linear relationship, i.e. $q_i(x_{(h,m)}) = q_i(x_{(h',m)})$ for every query pair (h, h'), then equality also holds for each of the $(q_i(x_{(h,m)}) \cdot f_b(z) - f_a(z))/z^j$ terms. If the latter term is truly a constant function c:

$$\frac{q_i(x_{(h,m)}) \cdot f_b(z) - f_a(z)}{z^j} = c$$

then we also have:

$$(a_m z^j + f_a(z)) \cdot f_b(z) - (b_m z^j + f_b(z)) \cdot f_a(z) = cz^j (b_m z^j + f_b(z)).$$

Each side is a polynomial with degree at most $2j$, and thus cannot agree on $2j+1$ points unless equality holds. However, if equality does hold, we have that either $c = 0$ or $b_m = 0$, as the left side has degree at most $2j-1$, and both z^j and $b_m z^j + f_b(z)$ are bounded away from 0 for any $z \neq 0$. If $c \neq 0$, then we have that $b_m = 0$ and $a_m = c$. If $c = 0$, then we have

$$a_m z^j f_a(z) f_b(z) - b_m z^j f_a(z) f_b(z) = 0,$$

which implies $a_m = b_m$, as $f_a(z) f_b(z)$ cannot be equal to 0 everywhere due to each a_0^i and b_0 being positive. Our answer to $q(x_{(h,m)})$ will be bounded above 0 and below 1, allowing for us to solve for both a_m and b_m as

$$a_m = b_m = \frac{q_i(x_{(h,m)}) \cdot f_b(z) - f_a(z)}{(1 - q_i(x_{(h,m)})) z^j}.$$

To summarize, if given exact query answers for $2j+1$ distinct points, we must be in one of the following cases:

- We observe at least two distinct linear relationships between a_m and b_m from differing query answers;
- We observe a non-zero constant $\frac{q_i(x_{(h,m)}) \cdot f_b(z) - f_a(z)}{z^j} = c$ for each query, and have $a_m = c$;
- We observe $\frac{q_i(x_{(h,m)}) \cdot f_b(z) - f_a(z)}{z^j} = 0$ for each query, and can solve for $a_m = b_m$.

To begin our error analysis for perturbed queries, we first show a bound on the size of the coefficients for a polynomial which is bounded over a range.

Lemma 10. *Each degree-d' coefficient of f_i^* is at most ${d'}^{2d'}$.*

Proof. First note that the constant coefficient and the coefficient for each linear term have magnitude at most 1, as the function is bounded in $[\lambda, 1]$ over the domain (which includes $\mathbf{0}$). For a degree-d' monomial m, consider the univariate polynomial corresponding to moving each of its variables in synchrony while holding the remaining variables at 0, whose degree-d' coefficient is equal to a_m. Consider the Lagrange polynomial representation of this polynomial

$$L_{d',j}(x) = \prod_{k \neq j}^{d'} \frac{x - x_k}{x_j - x_k};$$
$$\hat{f}_i(x) = \sum_{j=0}^{d'} \hat{y}_j L_{n,j}.$$

for $d' + 1$ evenly spaced points in the range $[-1/n, 1/d' - 1/n]$, which are all feasible under the simplex constraints (corresponding to $v_i \in [0, 1/d']$ in the original basis, for each $i \in m$). Each pair of points is separated by a distance of at least $1/({d'}^2)$, and so the leading coefficient of each Lagrange term is at most ${d'}^{2(d'-1)}$. Each $\hat{y}_j$ is in $[\lambda, 1]$ and so we have

$$\begin{aligned} a_m &\leq (d'+1){d'}^{2(d'-1)} \\ &\leq {d'}^{2d'} \end{aligned}$$

for each $d' > 1$. □

As we estimate coefficients for monomials of increasing degree, we will maintain the invariant that each degree-j coefficient of a and b is estimated up to additive error ϵ_j, with respect to the normalization where $b_0 = 1$. Immediately we have $\epsilon_0 = \beta$ for the estimates $\hat{a}_0$ from our query to the $\mathbf{0}$ vector. We will also let β_j denote the error of a polynomial $\hat{f}_a$ restricted to terms for subsets of a j-degree monomial m

For a monomial m, suppose we receive 2 queries $\hat{q}_i(x_{(h,m)})$ and $\hat{q}_i(x_{(h',m)})$ for some h and h' where

$$\left| \hat{q}_i(x_{(h,m)}) - \hat{q}_i(x_{(h',m)}) \right| \geq F_j$$

for some quantity F_j. Then we have:

$$\begin{aligned}\hat{a}_m &= \hat{q}_i(x_{(h,m)})\hat{b}_m + \frac{\hat{q}_i(x_{(h,m)}) \cdot \hat{f}_b(hZ) - \hat{f}_a(hZ)}{(hZ)^j} \\ &= \hat{q}_i(x_{(h',m)})\hat{b}_m + \frac{\hat{q}_i(x_{(h,'m)}) \cdot \hat{f}_b(h'Z) - \hat{f}_a(h'Z)}{(h'Z)^j} \\ \hat{b}_m &= \frac{\hat{a}_m}{\hat{q}_i(x_{(h,m)})} + \frac{\frac{\hat{f}_a(hZ)}{\hat{q}_i(x_{(h,m)})} - \hat{f}_b(hZ)}{(hZ)^j}; \\ &= \frac{\hat{a}_m}{\hat{q}_i(x_{(h',m)})} + \frac{\frac{\hat{f}_a(h'Z)}{\hat{q}_i(x_{(h',m)})} - \hat{f}_b(h'Z)}{(h'Z)^j}; \\ \frac{\hat{a}_m}{\hat{q}_i(x_{(h',m)})} - \frac{\hat{a}_m}{\hat{q}_i(x_{(h,m)})} &= \frac{\frac{\hat{f}_a(hZ)}{\hat{q}_i(x_{(h,m)})} - \hat{f}_b(hZ)}{(hZ)^j} - \frac{\frac{\hat{f}_a(h'Z)}{\hat{q}_i(x_{(h',m)})} - \hat{f}_b(h'Z)}{(h'Z)^j}; \\ \hat{a}_m &= \frac{\frac{\hat{q}_i(x_{(h',m)})\hat{f}_a(hZ)}{\hat{q}_i(x_{(h,m)})} - \hat{q}_i(x_{(h',m)})\hat{f}_b(hZ)}{\left(1 - \frac{\hat{q}_i(x_{(h',m)})}{\hat{q}_i(x_{(h,m)})}\right) \cdot (hZ)^j} - \frac{\hat{f}_a(h'Z) - \frac{\hat{f}_b(h'Z)}{\hat{q}_i(x_{(h',m)})}}{\left(1 - \frac{\hat{q}_i(x_{(h',m)})}{\hat{q}_i(x_{(h,m)})}\right) \cdot (h'Z)^j}; \\ \hat{b}_m &= \frac{\frac{\hat{q}_i(x_{(h,m)}) \cdot \hat{f}_b(hZ) - \hat{f}_a(hZ)}{(hZ)^j} - \frac{\hat{q}_i(x_{(h,'m)}) \cdot \hat{f}_b(h'Z) - \hat{f}_a(h'Z)}{(h'Z)^j}}{\hat{q}_i(x_{(h',m)}) - \hat{q}_i(x_{(h',m)})};\end{aligned}$$

where $\hat{f}_a$ and $\hat{f}_b$ are the univariate polynomials from summing the lower-order coefficient estimates for each degree up to $j-1$. The additive error to each $\hat{f}_a(hZ)$ and $\hat{f}_a(hZ)$ can be bounded by:

$$\beta + \sum_{k=1}^{j-1} \binom{n}{k} (hZ)^k k^{2k} \epsilon_k = \beta + \sum_{k=1}^{j-1} \binom{n}{k} (k^2 hZ)^k \epsilon_k.$$

Further, the magnitude of each $\hat{f}_a(hZ)$ and $\hat{f}_b(hZ)$ is at most $1 + \sum_{k=1}^{j-1} \binom{n}{k} (k^2 hZ)^k$. We can bound the error of other terms as follows:

- Each $\hat{q}_i(x_{(h',m)}) - \hat{q}_i(x_{(h',m)})$ has magnitude at least F_j and at most 1, and additive error at most 2β;
- Each $\hat{q}_i(x_{(h',m)})$ has value at least $\frac{\lambda}{n}$ and at most 1, and additive error at most β;
- Each $\frac{\hat{q}_i(x_{(h',m)})}{\hat{q}_i(x_{(h,m)})}$ term is either greater than $\frac{1}{1-F_j}$ or at most $1 - F_j$; the true ratio between the numerator and denominator is at least λ/n most n/λ, with additive error up to β in both.
- Each $1 - \frac{\hat{q}_i(x_{(h',m)})}{\hat{q}_i(x_{(h,m)})}$ term, is either greater than F_j or at most $1 - \frac{1}{1-F_j}$;
- Each $(hZ)^j$ has magnitude at least Z^j;

The error in the numerator of $\hat{a}_m$, and the fractional terms in the numerator of $\hat{b}_m$ is dominated by multiplying the functions of $\hat{q}_i$ with the polynomials themselves. As such, we can bound the error to a_m and b_m by ϵ_j if we have that:

$$\begin{aligned}\epsilon_j &\geq O\left(\frac{n\beta}{\lambda F_j Z^j} \cdot \left(1 + \sum_{k=1}^{j-1} \binom{n}{k} (k^2 hZ)^k\right)\right) \\ &= O\left(\frac{\beta \cdot \text{poly}(nd/\alpha)^j}{\lambda \alpha^j F_j}\right).\end{aligned}$$

for some polynomial. Now suppose all pairs of query answers we see are separated by less than F_j. The additive error to each estimate of the quantity

$$\hat{c}_{(h,m)} = \frac{\hat{q}_i(x_{(h,m)}) \cdot \hat{f}_b(z) - f_a(z)}{(hZ)^j}$$

is $\mathcal{E}_j = O\left(\frac{\beta}{Z^j} \cdot \left(1 + \sum_{k=1}^{j-1} \binom{n}{k}(k^2hZ)^k\right)\right) = O\left(\beta \cdot \text{poly}(nd/\alpha)^j\right)$. If each such quantity has value at most $\mathcal{E}_j$, we assume this quantity is zero and solve for $a_m = b_m$. If some are larger, we must be in the case where $\hat{b}_m \approx 0$ and so we set $a_m = \hat{c}_{(h,m)}$ for any query result. For sufficiently small F_j and $\mathcal{E}_j$, these both yield a bound of ϵ_j, as the resulting values of $\hat{a}_m$ and $\hat{b}_m$ will be close together regardless of which case we are in. For $\beta \leq O(\frac{\epsilon}{\text{poly}(nd/(\alpha\lambda))^d})$, we can take each F_j small enough to ensure a bound of ϵ_j for each coefficient, for values ϵ_j small enough to give a total Euclidean bound of ϵ for any vector of score estimates.

□

C.3 Proof of SFR Local Learnability

We now prove that functions with local sparse Fourier transformation are locally learnable. Recall that a function $f(x)$ has a ℓ-sparse Fourier transform if it can be written as

$$f(x) = \sum_{i=1}^{\ell} \xi_i e^{2\pi \mathbf{i} \eta_i x},$$

where η_i is the i-th frequency, ξ_i is the corresponding magnitude, and $\mathbf{i} = \sqrt{-1}$.

We will use the following result about learning sparse Fourier transforms [23].

Theorem 6 ([23]). *Consider any function $f(x) : \mathbb{R} \to \mathbb{R}$ of the form*

$$f(x) = f^*(x) + g(x),$$

where $f^*(x) = \sum_{i=1}^{\ell} \xi_i e^{2\pi \mathbf{i} \eta_i x}$ with frequencies $\eta_i \in [-F, F]$ and frequency separation $\hat{\alpha} = \min_{i \neq j} |\eta_i - \eta_j|$, and $g(x)$ is the arbitrary noise function. For some parameter $\delta > 0$, we define the noise-level *over an interval $I = [a, b] \subseteq \mathbb{R}$ as*

$$\mathcal{N}^2 = \frac{1}{|I|} \int_I |g(x)|^2 dx + \delta \sum_{i=1}^{\ell} |\xi_i|^2 .$$

There exists an algorithm that takes samples from the interval I with length $|I| > O(\frac{\log(\ell/\delta)}{\hat{\alpha}})$ and returns a set of ℓ pairs $\{(\xi'_i, \eta'_i)\}$ such that for any $|\xi_i| = \Omega(\mathcal{N})$ we have for an appropriate permutation of the indices

$$|\eta_i - \eta'_i| = O\Big(\frac{\mathcal{N}}{|I||\xi_i|}\Big), \qquad |\xi_i - \xi'_i| = O(\mathcal{N}), \forall i \in [\ell] .$$

The algorithm takes $O(\ell \log(F|I|) \log(\frac{\ell}{\delta}) \log(\ell))$ samples and $O(\ell \log(F|I|) \log(\frac{F|I|}{\delta}) \log(\ell))$ and succeeds with probability at least $1 - 1/k^c$ for any arbitrary constant c.

Furthermore, the algorithm used in the above theorem uses samples of the form $x_0, x_0 + \sigma \cdots x_0 + \ell \log(\ell/\delta)\sigma$ for randomly chosen x_0 and $\sigma = O(|I|/\ell \log(\ell/\delta))$.

We will use the above theorem to learn the sparse Fourier representation of the preference model. Recall that for a memory vector v and item $i \in [n]$, $M(v)_i = f_i(v_i)$.

Proof. Let v_{unif} denote the uniform memory vector. We will learn each function f_i separately. Fix $i \in [n]$. We will set the interval I to be $[1/n - Z, 1/n + Z]$ for some sufficiently small $\frac{\log(\ell/\delta)}{\hat{\alpha}} \leq Z \leq \alpha/2$ where $\hat{\alpha}$ is the frequency separation, where $\alpha = \tilde{\Omega}(1/\hat{\alpha})$ so that Z is defined. Let $S = \{x_j\}_{j=1}^{\tilde{O}(\ell)}$ for $x_j \in [-Z, Z]$ be a set of points such that the Fourier learning algorithm queries $1/n + x$ for each $x \in S$. For each point $x \in S$, we define the memory vector $v^x = v_{\text{unif}} + xe_i - xe_j$

where j is a fixed randomly chosen other index. All such vectors lie in V_α, as $2(\alpha/2)^2 \le \alpha^2$. We query all vectors v^x for $x \in S$, along with v_{unif}. Recall that $\hat{s}_v$ is the empirical score vector at a memory vector v. For each vector v, let R_v be the sum of all scores of all the $n-2$ items held at $\frac{1}{n}$, divided by the sum of those same items' scores in the uniform vector v_{unif}. For each vector v^x we multiply the score $\hat{s}_{v^x,i}$ of item i by R_{v^x} to obtain a noisy sample of $f_i(1/n+x)$. For $i \in \widetilde{O}(\ell)$, let the i-th sample be denoted by $\hat{y}_i$ and the true value $f_i(1/n+x_i)$ be denoted by y_i. We then pass all these samples to the Fourier learning algorithm in Theorem 6 in order to get an estimate $\hat{f}$ of f.

We now analyze the error in the samples. Let R_v^* be the corresponding ratio of these sums of scores under $\{f_i\}$; each sum is within $[\frac{\lambda}{3}, 1]$ at each vector, and the sums of observed scores have additive error at most $2n\beta$. As such, R_v has additive error at most $\frac{2n\beta}{\lambda}$ from R_v^*. For each vector v^x we have that $\hat{s}_{v^x,i}/(\sum_j \hat{s}_{v^x,j})$ is within a β error from $s_{v^x,i}/(\sum_j s_{v^x,j})$. Hence, the total error in each sample is bounded as:

$$|\hat{y}_i - y_i| \le \frac{7n\beta}{\lambda}\,.$$

Using this we can bound the total noise term by $\mathcal{N} = 8n\beta/\lambda$ using our choice of $\delta = (\beta n)/(\lambda \sum_{i=1}^{\ell} |\xi_i|)$. The algorithm will return a set of $\{(\hat{\eta}_i, \hat{\xi}_i)\}$ such that

$$|\eta_i - \eta_i'| = O\Big(\frac{1}{\alpha}\Big), \qquad |\xi_i - \xi_i'| = O(\frac{\beta n}{\lambda}), \forall i \in [\ell]\,.$$

So for function $\hat{f}_i(x)$ we can bound its distance from $f_i(x)$ by:

$$\begin{aligned} \Big|f_i(x) - \hat{f}_i(x)\Big| &= \left|\sum_{i=1}^{\ell} \xi_i e^{2\pi \mathbf{i} \eta_i x} - \sum_{i\in[\ell]} \hat{\xi}_i e^{2\pi \mathbf{i} \hat{\eta}_i x}\right| \\ &\le \sum_{i\in[\ell]} \left|\xi_i e^{2\pi \mathbf{i} \eta_i x} - \hat{\xi}_i e^{2\pi \mathbf{i} \hat{\eta}_i x}\right| \\ &\le \sum_{i\in[\ell]} \left|\xi_i - \hat{\xi}_i\right| |\eta_i - \hat{\eta}_i| \\ &\le O(\frac{\ell n \beta}{\lambda \alpha}), \end{aligned}$$

since we normalize the above estimates to get a score estimate, the total bound on the score estimates can be bounded as:

$$\left|\frac{\hat{f}_i(x)}{\sum_{j=1}^{x} \hat{f}_j(x)} - \frac{f_i(x)}{\sum_{j=1} f_j(x)}\right| \le O(\frac{\ell \beta n}{\alpha \lambda}).$$

Taking $\beta \le \frac{\epsilon \lambda \alpha}{\sqrt{n}\ell}$ gives us an error of at most $\epsilon\sqrt{n}$, satisfying a Euclidean bound of ϵ from any true score vector $M(w)/M_w^*$ for our hypothesis model $\hat{M}(v) = \{\hat{f}_i(v_i) : i \in [n]\}$. □

D Omitted Proofs for Section 3

D.1 Proof of Theorem 1

Proof. First observe that $y_t \in \mathcal{K}_t$ every round, as

For $x^* = \arg\min_{x\in\mathcal{K}_T} \sum_{t=1}^{T} f_t(x)$, let $x^*_{\delta,\epsilon} = \Pi_{\mathcal{K}_{T,\delta,\epsilon}}(x^*)$. By linearity and properties of projection, we also have that $x^*_{\delta,\epsilon} = \arg\min_{x\in\mathcal{K}_{T,\delta,\epsilon}} \sum_{t=1}^{T} f_t(x)$, and that $\left\|x^*_{\delta,\epsilon} - x^*\right\| \le (\delta+\epsilon)\frac{D}{r}$. For G-Lipschitz losses $\{f_t\}$ we have

$$\begin{aligned} \sum_{t=1}^{T} \mathbb{E}[\phi_t] - \sum_{t=1}^{T} f_t(x^*) &= \sum_{t=1}^{T} \mathbb{E}[f_t(y_t)] - \sum_{t=1}^{T} f_t(x^*) \\ &\le \sum_{t=1}^{T} \mathbb{E}[f_t(y_t)] - \sum_{t=1}^{T} f_t(x^*_{\delta,\epsilon}) + \delta T G \frac{D}{r} + \epsilon T G \frac{D}{r}. \end{aligned}$$

Let $\hat{f}_t(x) = \mathbb{E}_{u\sim\mathbb{B}}[f(x+\delta u+\xi_t)] = f_t(x+\xi_t)$ by linearity. Then we can bound the regret by:

$$\begin{aligned}
\sum_{t=1}^{T}\mathbb{E}[\phi_t] - \sum_{t=1}^{T} f_t(x^*) &\le \sum_{t=1}^{T}\mathbb{E}[f_t(y_t)] - \sum_{t=1}^{T} f_t(x^*_{\delta,\epsilon}) + \frac{\delta TGD}{r} + \frac{\epsilon TGD}{r} \\
&= \sum_{t=1}^{T}\mathbb{E}[\hat{f}_t(x_t)] - \sum_{t=1}^{T} f_t(x^*_{\delta,\epsilon}) + \frac{\delta TGD}{r} + \frac{\epsilon TGD}{r} \\
&\le \sum_{t=1}^{T}\mathbb{E}[\hat{f}_t(x_t)] - \sum_{t=1}^{T} \hat{f}_t(x^*_{\delta,\epsilon}) + \frac{\delta TGD}{r} + \epsilon TG\left(\frac{D}{r}+1\right) \\
&\le \sum_{t=1}^{T}\mathbb{E}[\hat{f}_t(x_t)] - \sum_{t=1}^{T} \hat{f}_t(x^*_{\delta,\epsilon}) + \frac{\delta TGD}{r} + \frac{2\epsilon TGD}{r}
\end{aligned}$$

Next, we prove a series of lemmas — an analysis of online gradient descent for contracting decision sets, and a corresponding bandit-to-full-information reduction — which allow us to view the remaining summation terms involving $\{x_t\}$ as the expected regret of stochastic online gradient descent for the loss function sequence $\{\hat{f}_t\}$ with respect to $\mathcal{K}_{T,\delta,\epsilon}$.

When modifying online gradient descent to project into smaller sets each round, the analysis is essentially unchanged.

Algorithm 3 Contracting Online Gradient Descent.

Input: sequence of contracting convex decision sets $\mathcal{K}_1, \dots \mathcal{K}_T$, $x_1 \in \mathcal{K}_1$, step size η
Set $x_1 = \mathbf{0}$
for $t = 1$ to T **do**
  Play x_t and observe cost $f_t(x_t)$
  Update and project:

$$\begin{aligned}
y_{t+1} &= x_t - \eta\nabla\ell_t(x_t) \\
x_{t+1} &= \Pi_{\mathcal{K}_{t+1}}(y_{t+1})
\end{aligned}$$

end for

Lemma 11. *For a sequence of contracting convex decision sets $\mathcal{K}_1, \dots \mathcal{K}_T$, $x_1 \in \mathcal{K}_1$ each with diameter at most D, a sequence of G-Lipschitz losses $\ell_1, \dots, \ell_T$, and $\eta = \frac{D}{G\sqrt{T}}$, the regret of Algorithm 3 with respect to $\mathcal{K}_t$ is bounded by*

$$\sum_{t=1}^{T}\ell_t(x_t) - \min_{x^*\in\mathcal{K}_T}\sum_{t=1}^{T}\ell_t(x^*) \le GD\sqrt{T}.$$

Proof. Let $x^* = \arg\min_{x\in\mathcal{K}_T}\sum_{t=1}^{T}\ell_t(x)$, and let $\nabla_t = \nabla\ell_t(x_t)$. First, note that

$$\ell_t(x_t) - \ell_t(x^*) \le \nabla_t^\top(x_t - x^*)$$

by convexity; we can then upper-bound each point's distance from x^* by:

$$\|x_{t+1} - x^*\| = \left\|\Pi_{\mathcal{K}_{t+1}}(x_t - \eta\nabla\ell_t(x_t))\right\| \le \|x_t - \eta\nabla_t - x^*\|,$$

using projection properties for convex bodies. Then we have

$$\|x_{t+1} - x^*\|^2 \le \|x_t - x^*\|^2 + \eta^2\|\nabla_t\|^2 - 2\eta\nabla_t^\top(x_t - x^*)$$

and

$$\nabla_t^\top(x_t - x^*) \le \frac{\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2}{2\eta} + \frac{\eta\|\nabla_t\|^2}{2}.$$

We can then conclude:

$$\begin{aligned}
\sum_{t=1}^T \ell_t(x_t) - \sum_{t=1}^T \ell_t(x^*) &\le \sum_{t=1}^T \nabla_t^\top (x_t - x^*) \\
&\le \sum_{t=1}^T \frac{\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^T \|\nabla_t\|^2 \\
&\le \frac{\|x_T - x^*\|^2}{2\eta} + \frac{\eta}{2}\|\nabla_t\|^2 \\
&\le \frac{D^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^T \|\nabla_t\|^2 \\
&= GD\sqrt{T} \qquad \text{(when } \eta = \frac{D}{G\sqrt{T}}\text{)}
\end{aligned}$$

□

The bandit-to-full-information reduction is fairly standard as well, with a proof equivalent to that of e.g. Lemma 6.5 in [16], modified for a full-information algorithm $\mathcal{A}$ for over contracting sets.

Lemma 12. *Let u be a fixed point in $\mathcal{K}_T$, let $\{\ell_t : \mathcal{K}_t \to \mathbb{R} \mid t \in [T]\}$ be a sequence of differentiable loss functions, and let $\mathcal{A}$ be a first-order online algorithm that ensures a regret bound $Regret_{\mathcal{K}_T}(\mathcal{A}) \le B_{\mathcal{A}}(\nabla\ell_1(x_1), \dots, \nabla\ell_T(x_T))$ in the full-information setting for contracting sets $\mathcal{K}_1, \dots, \mathcal{K}_T$. Define the points $\{x_t\}$ as $x_1 \leftarrow \mathcal{A}(\emptyset)$, $x_t \leftarrow \mathcal{A}(g_1, \dots, g_{t-1})$, where g_t is a random vector satisfying*

$$\mathbb{E}[g_t | x_1, \ell_1, \dots, x_t, \ell_t] = \nabla\ell_t(x_t).$$

Then for all $u \in \mathcal{K}_T$:

$$\mathbb{E}[\sum_{t=1}^T \ell_t(x_t)] - \sum_{t=1}^T \ell_t(u) \le E[B_{\mathcal{A}}(g_1, \dots, g_T)] \tag{1}$$

Proof. Let $h_t : \mathcal{K}_t \to \mathbb{R}$ be given by:

$$h_t(x) = \ell_t(x) + \psi_t^\top x, \text{ where } \psi_t = g_t - \nabla\ell_t(x_t).$$

Note that $\nabla h_t(x_t) = g_t$, and so deterministically applying a first order algorithm $\mathcal{A}$ on $\{h_t\}$ is equivalent to applying $\mathcal{A}$ on stochastic first order approximations of $\{f_t\}$. Thus,

$$\sum_{t=1}^T h_t(x_t) - \sum_{t=1}^T h_t(u) = \le B_{\mathcal{A}}(g_1, \dots, g_T).$$

Using the fact that the expectation of each ψ_t is 0 conditioned on history, and expanding, we get that

$$\begin{aligned}
\mathbb{E}[h_t(x_t)] &= \mathbb{E}[\ell_t(x_t)] + \mathbb{E}[\psi_t^\top x_t] \\
&= \mathbb{E}[\ell_t(x_t)] + \mathbb{E}[\mathbb{E}[\psi_t^\top x_t | x_1, \ell_1, \dots, x_t, \ell_t]] \\
&= \mathbb{E}[\ell_t(x_t)] + \mathbb{E}[\mathbb{E}[\psi_t | x_1, \ell_1, \dots, x_t, \ell_t]^\top x_t] \\
&= \mathbb{E}[\ell_t(x_t)],
\end{aligned}$$

and we can conclude by taking the expectation of Equation 1 for any point $u \in \mathcal{K}_T$. □

The key remaining step is to observe that each g_t is an unbiased estimator of $\nabla\hat{f}_t(x_t)$:

$$\begin{aligned}
\mathbb{E}[g_t | x_1, \hat{f}_1, \dots, x_t, \hat{f}_t] &= \frac{n}{\delta}\mathbb{E}[\phi_t u_t | x_t, \hat{f}_t] \\
&= \frac{n}{\delta}\mathbb{E}[\mathbb{E}[\phi_t | x_t, \hat{f}_t, u_t] \cdot u_t | x_t, \hat{f}_t] \\
&= \mathbb{E}[f_t(x_t + \delta u_t + \xi_t) u_t | x_t, \hat{f}_t] \\
&= \mathbb{E}[\hat{f}_t(x_t + \delta u_t) u_t] \\
&= \nabla\hat{f}_t(x_t),
\end{aligned}$$

where the final line makes use the sphere sampling estimator for linear functions (as in e.g. Lemma 6.7 in [16]). This allows us to apply Lemma 12 to Algorithm 3:

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}[\phi_t]-\sum_{t=1}^{T}f_t(x^*) &\leq \sum_{t=1}^{T}\mathbb{E}[\hat{f}_t(x_t)]-\sum_{t=1}^{T}\hat{f}_t(x^*_{\delta,\epsilon})+\frac{\delta TGD}{r}+\frac{2\epsilon TGD}{r}\\ &\leq \text{Regret}_{COGD}\left(g_1,\ldots,g_T|\{\hat{f}_t\}\right)+\frac{\delta TGD}{r}+\frac{2\epsilon TGD}{r}\\ &\leq \frac{D^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\|g_t\|^2+\frac{\delta TGD}{r}+\frac{2\epsilon TGD}{r}\\ &\leq \frac{D^2}{2\eta}+\eta\frac{n^2}{2\delta^2}T+\frac{\delta TGD}{r}+\frac{2\epsilon TGD}{r} \qquad \text{(def. of } g_t,\phi\leq 1)\\ &\leq nGDT^{3/4}+\frac{GDT^{3/4}}{r}+\frac{2\epsilon TGD}{r} \qquad (\eta=\frac{D}{nT^{3/4}},\delta=\frac{1}{T^{1/4}}).\end{aligned}$$

□

E Omitted Proofs for Section 4

E.1 Proof of Lemma 4

Proof. Consider any memory vector $v \in \Delta(n)$. We can show constructively that there is some distribution of menus z_U which induces the all-$\frac{1}{n}$ vector.

We construct z_U in $\frac{1}{\tau}+1$ stages for some $\tau > 0$, through a process where we continuously add weight a_{z_j} to a sequence of distributions $\{z_j | j \geq 1\}$ over menus until the total weight $\sum_j a_{z_j}$ sums to 1. The uniform-inducing menu distribution z_U will then be defined by taking the mixture of the menu distributions z_j where each is weighted by a_{z_j}.

Consider the uniform distribution over all menus; continuously add weight to this distribution until some item (the one with the largest score in M) has selection weight τ/n (its selection probability under M at memory vector v in each distribution of menus z_j considered thus far, weighted by a_{z_j}). While there are at least k items with selection weight τ/n, continuously add weight to the uniform distribution over all menus containing only items with weight below τ/n.

Once there are fewer than k items with selection weight at most τ/n, we terminate stage 1. In general, for stage i, we always include every item with weight below $\tau i/n$ in the menu, with all others chosen uniformly at random.

Inductively, we can see that every item starts stage i with at least weight $\tau(i-1)/n$ and at most $\tau i/n$, with at most $k-1$ items having weight less than $\tau(i-1)/n$. Crucially, any item with weight less than $\tau i/n$ at the start of stage i will reach weight $\tau i/n$ before any item starting at weight $\tau i/n$ reaches weight $\tau(i+1)/n$. Such an item is included in every menu until this occurs, resulting in a selection probability of at least $\frac{\lambda}{k}$ in each menu distribution considered, whereas any other item is only included in the menu with probability $\frac{k}{n}$, which bounds its selection probability in the menu distribution. As $\frac{\lambda}{k} \geq \frac{k}{n}$, the selection weight of items beginning stage i below $\tau i/n$ reaches $\tau i/n$ no later than when the stage terminates.

After stage $\frac{1}{\tau}$, every item has weight at most $\frac{1}{n}$ and at least $\frac{1}{n}-\frac{\tau}{n}$. We continue for one final stage until the sum of weights is 1, at which point every item has a final weight $p_{z_U} \in [\frac{1}{n}-\frac{\tau}{n}, \frac{1}{n}+\frac{\tau}{n}]$. Taking the limit of τ to zero gives us that x_U is in $\texttt{IRD}(v, M)$ for any v, and hence x_U is in $\texttt{EIRD}(M)$ as well.

Further, there is a distribution of menus z_{b_i} where i has probability $p_{b_i,i} = k/n$ and every other item j has probability

$$p_{b_i,j} = \frac{1}{n} - \frac{k-1}{n(n-1)}$$

Here, we include i in every menu and run the previous approach over the remaining $n-1$ items for menus of size $k-1$, which we then augment with i. The required bound on λ still holds for any $\lambda < 1$, as $\frac{k^2}{n} \geq \frac{(k-1)^2}{n-1}$ (for any $k \leq \sqrt{n}$, which holds as $\lambda < 1$). The selection probability of i will be at least $\frac{\lambda}{k} \geq \frac{k}{n}$; we can take a mixture of this menu distribution with z_U such that $p_{b_i,i} = \frac{k}{n}$ exactly.

The convex hull of each p_{b_i} is thus contained in $\texttt{EIRD}(M)$, as any point $p \in \text{convhull}\{p_{b_i} | i \in [n]\}$ can be generated by taking the corresponding convex combination of menu distributions z_{b_i}. Any point $x \in \Delta(n)$ where $\|x_U - x\|_\infty \leq \frac{k-1}{n(n-1)}$ can then be induced by taking mixtures of the z_{b_i} menu distributions.

□

E.2 Subset-Uniform Distributions in EIRD

Lemma 13. *For any λ-dispersed M where $\lambda \geq \frac{Ck^2}{n}$, $\texttt{EIRD}(M)$ contains the uniform distribution over any $\frac{n}{C}$ items.*

Proof. The proof of Lemma 4 carries through directly for a universe with only $\frac{n}{C}$ items. □

E.3 Implementing Near-Uniform Vectors

Lemma 14. *For any λ-dispersed M where $\lambda \geq \frac{k^2}{n}$, for any point $x \in \Delta(N)$ satisfying*

$$\|x - x_U\|_\infty \leq \frac{k-1}{n(n-1)},$$

there is an adaptive strategy for selecting a sequence of menus over t^* rounds, resulting in a t^*-round empirical distribution $\hat{x}$ such that $\|x - \hat{x}\|_\infty \leq \gamma(t^* - t_0)$ with probability at least $1 - 2n\exp(-\gamma^2(t^* - t_0)/8)$, where $t_0 \leq \frac{2(k-1)}{(n-k)}t^*$, for any γ.

Proof. For each item i, let $V_i = t^* \cdot x_i$ target number of rounds where i is selected over the window; for any $t \leq t^*$ let $\hat{V}_{t,i}$ denote the empirical number of rounds in which i has been selected. Our strategy will essentially correspond to the construction in Lemma 4, which shows that our vector is indeed in $\texttt{EIRD}(M)$.

We describe a process for menu generation where every pair of items has an expected count difference at most 2. For round $t = 1$ to t:

- Let B_t be the bottom k items, tiebreaking randomly amongst items sharing the kth smallest count. Include all of B_t in the menu.
- Let A_t be the remainder, which are not included in the menu.

Let $V_{t,i} = \mathbb{E}[\hat{V}_{t,i}]$ under this process. After round $n-k$, each item in B_t has a count of 0, and each in A_t has a count of 1. Consider the expected count of an item $V_{t,i=} \mathbb{E}[\hat{V}_{t,i}]$ as stages progress. At round $n-k$, every item is within 1 of the maximal count, which we denote $V_t^* = \max_i \hat{V}_{t,i}$. We maintain the invariant throughout that $\mathbb{E}[V_t^* - \hat{V}_{t,i}] \leq 2$ for all i. Consider a round in which the maximal count has just increased to V_t^*. If $\hat{V}_{t,i} \leq V_t^* - 2$, then $\hat{V}_{t,i}$ will be in B_t in every stage until it increases, as there must have been at least $n-k+1$ items with a count of $V_t^* - 1$ in the previous round for the maximal count to increase. While i is in the menu, its count will increase in each round with a probability at least λ/k, and so the expected number of rounds until it increases is at most $\frac{k}{\lambda} \leq \frac{n}{k}$, and the probability of this occurring within $\frac{n}{k}$ rounds is at least $1 - e^{-1}$, as $(1 - \lambda/k)^{n/k} \leq (1 - k/n)^{n/k} \leq e^{-1}$. Note that the maximum count cannot increase for at least $n-k+1$ rounds, and so its expected time to increase must be larger. For any $k < n$ this results in $\hat{V}_{t,i}$ increasing first in expectation. When viewing the entire sequence of rounds, we can consider every instance where V_t^* increases to some value above $\hat{V}_{t,i} + 1$ as the start of a random trial over whether V_t^* or $\hat{V}_{t,i}$ increases next, in which $\hat{V}_{t,i}$ wins the majority. As such, $\mathbb{E}[V_{t^*}^*]$ cannot be larger than $V_{t,i}+2$ for any i.

Given a set of empirical counts $\{\hat{V}_{t,i} : i \in [n]\}$, the process is fully defined at a given round. Let $X_{t,i} = \Pr[i \text{ increases} | \{\hat{V}_{t-1,i}\}]$. For this process, we can now view each quantity $Y_{t,i} = (\hat{V}_{t,i} - \hat{V}_{t-1,i})$ as a Bernoulli random variable with mean $X_{t,i}$. Then we can define $Z_{t,i} = \sum_{h=1}^{t} Y_{h,i} - X_{h,i}$ as a martingale, where $\mathbb{E}[Z_{t,i}] = Z_{t-1,i}$ and $|Z_{t,i} - Z_{t-1,i}| \leq 2$. Note that $\mathbb{E}[Z_{t^*,i}]$ is simply equal to $V_{t^*,i}$. We can then apply Azuma's inequality to get:

$$Pr\left[\left|\hat{Z}_{t^*,i} - V_{t^*,i}\right| \geq \gamma t^*\right] \leq 2\exp\left(\frac{-\gamma^2 t^*}{8}\right).$$

This allows us to implement the uniform distribution with high probability. To handle other vectors in the allowed ℓ_∞ ball, consider each item's desired mass above $\frac{1}{n} - \frac{k-1}{n(n-1)}$. We can run a deterministic process only selecting items where this is positive, removing from consideration when their target mass above $\frac{1}{n} - \frac{k}{n(n-1)}$ is reached, until there are at most $k-1$ items with remaining target mass above the threshold, each of which is at most $\frac{2(k-1)}{n(n-1)}$. Call this set E_0, whose fraction of the remaining total required mass is at most:

$$\frac{\frac{2(k-1^2)}{n(n-1)}}{1 - \frac{k-1}{n-1}} = \frac{2(k-1)^2}{n(n-k)}.$$

Let t_0 be the number of steps elapsed thus far, which is at most $\frac{2(k-1)}{(n-k)}t^*$. We can replicate the procedure from above, initializing each item i in E_0 to have the appropriate negative count $\hat{V}_{t_0,i}$. It suffices to show that each $i \in E_0$ has a sufficiently large $V_{t^*,i}$. Each round t, let $\mathbb{E}_t$ be the remaining items from E_0 which have never left B_t. If we were to never remove items from E_t, each would see its count increase at least every $\frac{\lambda}{k}$ rounds in expectation, and the maximum count increases at most every $n-k+1$ rounds. Again, we can view each round in which V_t^* increases and $\hat{V}_{t,i} \leq V_t^* - (k-1)$ as the beginning of a random trial, where i increases its count by at least $(n-k+1)\lambda/k \geq k - \frac{k^2}{n} + 1/n \geq k-1$, with analogous trials for count differences between 2 and $k-1$ which yield closing the gap to 1. For every trial, we can decompose $E[\hat{V}_{t,i}]$ by its distance from V_t^* at the beginning of a trial, and by whether it has left E_t. If it remains in E_t, the expected gap between $\hat{V}_{t,i}$ and V_t^* either shrinks by $k-1$ or to within 1, which will catch up to $V_t^* - 2$ before we reach t^*. If it does not, then we are simply in the case from the uniform distribution. As such, we maintain an expected count of at least $V_t^* - 2$ for each item, and the martingale analysis is symmetric over $t^* - t_0$ rounds, yielding:

$$Pr\left[\left|\hat{Z}_{t^*,i} - V_{t^*,i}\right| \geq \gamma(t^* - t_0)\right] \leq 2\exp\left(\frac{-\gamma^2(t^* - t_0)}{8}\right).$$

□

E.4 Proof of Theorem 5

Proof. Let:

- $F_{LL} = f_{LL}(\lambda, \alpha, n, \mathcal{M})$ s.t. $\mathcal{A}_\mathcal{M}$ with β/F_{LL} results in $\epsilon_{LL} = \frac{\epsilon\lambda k}{n}$;
- $F_Q = \frac{8L\sqrt{n}k}{\lambda}F_{LL}$;
- $t_{\text{query}} = \frac{2n}{k-1}\left(\frac{F_{LL}}{\beta}\right)^2 \log\left(\frac{2nkS}{(k-1)\delta_{\text{query}}}\right) = \tilde{\Theta}(1/\epsilon^2)$;
- $t_{\text{pad}} = \max\left(\frac{2F_Q t_{\text{query}}}{\beta}, \frac{32n^2F_Q^2\log(2/\delta_{\text{pad}})}{\beta^2}\right) = \tilde{\Theta}(1/\epsilon^3)$;
- $t_{\text{move}} = \max\left(\frac{n(n-1)t_{\text{query}}}{k-1}, \frac{32n^2F_Q^2\log(4S/\delta_{\text{move}})}{(1-4k/n)\beta^2}, t_{\text{pad}}\right) = \tilde{\Theta}(1/\epsilon^3)$;

- $t_0 = t_{\text{pad}} + S(2 \cdot t_{\text{move}} + t_{\text{query}}) = \tilde{\Theta}(1/\epsilon^3)$.

After running `UniformPad` via the first Lemma 14 construction for t_{pad} steps, our empirical memory vector is within ℓ_∞ distance $\frac{\beta}{nF_Q}$ of x_U with probability at least $1 - \delta_{\text{pad}}$. We maintain the invariant that when calling `MoveTo`(x) to reach some non-uniform vector x from x_U, the ℓ_∞ distance between x and x_U is at most α, and that after calling `Query`(x) the current vector x' (accounting for drift during sampling) has ℓ_∞ distance at most α from x_U.

At any time $t < t_0$ when `MoveTo` is called, the proportion of steps which the current invocation will contribute to the total history is at least:

$$R_{\text{move}} = \frac{t_{\text{move}}}{t_{\text{pad}} + S(t_{\text{move}} + t_{\text{query}})} = O(1/S)$$

Let $\alpha = \frac{k-1}{2n(n-1)} \cdot R_{\text{move}}$ denote the radius of the ℓ_2 ball around x_U in which we permit queries for local learning. Any point x within the α-ball around the uniform vector can reach (or be reached from) the uniform vector with one call to `MoveTo`(x), as their ℓ_∞ distance is at most α, so some difference vector exists with mass R_{move} and which satisfies the required norm bound. For each input x, called from x_t, `MoveTo`(x) applies the construction from Lemma 14 for the mass t_{move} vector $y = x \cdot (t_{\text{move}}) - x_t \cdot t$. This results in a total error of at most $\frac{\beta}{2nF_Q} \cdot t_{\text{move}} + 1 \leq \frac{\beta}{nF_Q} \cdot t_{\text{move}}$ per item count with probability at least $1 - \delta_{\text{move}}$, as

$$t_{\text{move}} \geq \frac{32n^2 F_Q^2 \log(4S/\delta_{\text{move}})}{(1 - 4k/n)\beta^2}.$$

This yields a total variation distance within $\frac{\beta}{2F_Q}$ for the entire memory vector when appended to the current history.

To run `Query`(x), consider a set of $\frac{n}{k-1}$ menus, where item 1 appears in every menu and every other item appears in exactly one. Over the following t_{query} rounds, play each menu $t_{\text{query}} \cdot \frac{k-1}{n}$ times and note the proportion of each item observed relative to item 1 when its menu was played. Each scoring function $f_i \in M$ is L-Lipschitz; we run `Query`(x) for t_{query} rounds, which can introduce a drift of at most $\beta/(2F_Q)$ in total variation distance given the bound on t_{query} in terms of t_{pad}. This drift results in a vector which remains within ℓ_∞ distance 2α from x_U, and so x_U can still be reached again in a single `MoveTo`(x_U) call.

The empirical average memory vector over all menu queries (for any item) is within β/F_Q total variation distance from x, and so the expected distribution of items differs from that at x by at most $\beta/F_Q \cdot \frac{4L\sqrt{n}k}{\lambda} = \beta/(2F_{LL})$ in ℓ_∞ distance. Each point's observed frequency differs from that expectation by at most $\beta/(2F_{LL})$ with high probability. For an item i in the menu at a given round, we view whether or not it was chosen as a Bernoulli random variable, with mean equal to its relative score among items in the menu. Let $\bar{s}_{v,K,i}$ be the expected frequency of observing an item when the menu K containing it is played, given the empirical sequence of memory vectors during those rounds $t_{\text{query}} \cdot \frac{k-1}{n}$, and let $\hat{s}_{v,K,i}$ be the true observed frequency. We then have:

$$\begin{aligned} Pr\left[|\bar{s}_{v,K,i} - \hat{s}_{v,K,i}| \geq \frac{\beta}{2F_{LL}}\right] &\leq 2e^{\left(-2(\beta/2F_{LL})^2 t_{\text{query}}(k-1)/n\right)} \\ &= 2e^{\left(-(\beta/F_{LL})^2 t_{\text{query}}(k-1)/(2n)\right)} \\ &\leq \frac{\delta_{\text{query}}(k-1)}{nkS}, \end{aligned}$$

given that

$$t_{\text{query}} \geq \frac{2n}{k-1}\left(\frac{F_{LL}}{\beta}\right)^2 \log\left(\frac{2nkS}{(k-1)\delta_{\text{query}}}\right).$$

For item 1 take the average over all menus, and rescale such that all scores sum to 1 (using the frequency of item i relative to the frequency of item 1 when both were in the menu). Each score, and its error bound, will only shrink under the rescaling. This gives us score vector estimates $\hat{s}_x$ for each

$x \in S$ with additive error at most $\frac{\beta}{F_{LL}}$ relative to the true frequency of item 1, and thus overall, where $F_{LL} = f_{LL}(\lambda, \alpha, n, \mathcal{M})$. This holds for every query simultaneously with probability $1 - \delta_{\text{query}}$.

By the local learnability guarantee for $\mathcal{M}$, running $\mathcal{A}_{\mathcal{M}}$ our results in a hypothesis $\hat{M}$ which has ℓ_2 error at most $\epsilon_{LL} = \frac{\epsilon \lambda k}{n}$ for any $x \in \Delta(n)$. In each round, the model and memory vector defines a space of feasible item distributions. This allows us to run RC-FKM for perturbations up to ϵ. We can represent each set $\texttt{IRD}(v_t, \hat{M})$ explicitly as the convex hull of normalized score estimates for every menu.

We implement $\texttt{PlayDist}(x)$ using current score estimates $\hat{M}(v_t)$ to generate a menu distribution which approximately induces the instantaneous item distribution x. Taking the convex hull over every menu's score vector under $\hat{M}$ yields a polytope representation of $\texttt{IRD}(v_t, \hat{M})$, which will contain our chosen action at each step.

Lemma 15. *Let x be a point in $\texttt{IRD}(v, M)$, and let $z \in \Delta(\binom{n}{k})$ be a non-negative vector such that $\sum_{j \in \binom{n}{k}} z_j \cdot p_{K_j, v} = x$, where K_j is the jth menu in lexicographic order. If the Recommender randomly selects a menu K to show the Agent with probability according to z, then the Agent's item selection distribution is x.*

Proof. The probability that the Agent selects item i is obtained by first sampling a menu, then selecting an item proportionally to its score:

$$\Pr[\text{Agent selects } i] = \sum_{j \in \binom{n}{k}} z_j \cdot p_{K_j, v, i} = x_i.$$

□

Lemma 16. *Given $\hat{M}$ satisfying $\frac{\epsilon \lambda k}{n}$-accuracy and a target vector $x_t \in \texttt{IRD}(v_t, \hat{M})$ generated by* RC-FKM*, there is a linear program for computing a menu distribution z_t such that the induced item distribution p_{z_t} satisfies*

$$\|p_{z_t} - x_t\| \leq \epsilon.$$

Proof. We can define a linear program to solve for z with:

- variables for $z_j \in [0, 1]$, where $\sum_{j \in \binom{n}{k}} z_j = 1$,
- estimated induced distributions for each menu $\hat{p}_{K_j}$, and
- a constraint for each $i \in [n]$:

$$\sum_{j=1}^{\binom{n}{k}} z_j \cdot \hat{p}_{K_j, i} = x_{t,i}.$$

If $\left\| \hat{M}(x)/\hat{M}^* - M(x)/M^*_x \right\| \leq \frac{\epsilon \lambda k}{n}$, then for any menu distribution z, we have that:

$$\|p_{z,v} - \hat{p}_{z,v}\| \leq \epsilon.$$

Consider some menu K. The ℓ_2 distance of score vectors restricted to the menu is at most $\frac{\epsilon \lambda k}{n}$, and each vector has mass at least $\frac{k\lambda}{n}$ by dispersion. Rescaling vectors to have mass 1 yields a bound of ϵ, which is preserved under mixture (which is the induced distribution by Lemma 15), as well as when projecting into the $n-1$ dimensional space for RC-FKM, and so there is some perturbation vector ξ_t with norm at most ϵ such that z induces $x_t + \xi_t$. □

Note that the losses for RC-FKM can be $2G$-Lipschitz after the reparameterization where $x_{t,n} = 1 - \sum_{i=1}^{n-1} x_{t,i}$. Any point satisfying within radius $r = \frac{k-1}{n(n-1)}$ from the uniform distribution in n dimensions, feasible by Lemma 4, is within distance r under the reparameterization as well, as we simply drop the term for x_n. The required radius surrounding $\mathbf{0}$ for RC-FKM of r is thus satisfied, and we have that $\epsilon + \delta \leq r/T^{1/4} \leq r$. Further, the diameter of the simplex is bounded by $D = 2$. We can directly apply the regret bound of RC-FKM for these quantities, which holds with respect to $H_c \cap \texttt{EIRD}(\hat{M})$. By Lemma 16, for any point $x \in \texttt{EIRD}(\hat{M})$, there is a point $x' \in \texttt{EIRD}(M)$ such that $\|x - x'\| \leq \epsilon$. Projecting both points into H_c cannot increase their distance by convexity, and so the optimality gap between the two sets is at most ϵGT. Our total regret is at most the sum of:

- Maximal regret for the learning runtime $G \cdot t_0$;
- The regret of RC-FKM over $T - t_0$ rounds;
- The gap between $\texttt{EIRD}(\hat{M})$ and $\texttt{EIRD}(M)$; and
- The union bound of each event's failure probability.

We can bound this by:

$$\begin{aligned}
\text{Regret}_{C \cap \texttt{EIRD}(M)}(T) &\leq G \cdot t_0 + 4nGT^{3/4} + \frac{4(\delta + 2\epsilon)GT}{r} + \epsilon GT + (\delta_{\text{pad}} + \delta_{\text{move}} + \delta_{\text{query}})T \\
&= \tilde{O}(T^{3/4})
\end{aligned}$$

when taking each of $\{\delta_{\text{pad}}, \delta_{\text{move}}, \delta_{\text{query}}\} = \frac{1}{T^{1/4}}$. We can also bound the empirical distance from H_c.

Lemma 17. *The diversity constraint is $O(\epsilon)$-satisfied by the empirical distribution v_T with probability $1 - O(T^{-1/4})$.*

Proof. Note that after t_0, the empirical distribution v_{t_0} is within total variation distance $\frac{\beta}{2F_Q}$ from x_U (which is necessarily in H_c). Further, each vector x_t played by `RC-FKM` results in a per-round expected item distribution y_t which lies in H_c by the robustness guarantee. We can apply a similar martingale analysis as in Lemma 14 to the sequence of realizations of any item versus its cumulative expectation $\sum_{t > t^*} y_t$ to get a bound of (much less than) $\frac{\beta}{2F_Q}$ in total variation distance as well, which is preserved under mixture. For any locally learnable class, $\beta = O(\epsilon)$. Note that for all the classes we consider, we have $\beta/(2F_Q) \ll \epsilon$. Both events hold with probability $1 - O(T^{-1/4})$, as we can apply the same failure probabilities used for the learning stage for each.

Note that for a constraint H_c where c is sufficiently bounded away from $\log(n)$ and for large enough T, this will in fact yield an empirical distribution which exactly satisfies H_c, as the weight $\tilde{O}(T^{3/4})$ uniform window will "draw" the empirical distribution back towards the center of H_c, as it dominates the total $\tilde{O}(T^{1/2})$ total error bound (for the unnormalized empirical histogram $T \cdot v_T$) obtainable with a martingale analysis over the entire RC-FKM window. □

This completes the proof of the theorem.

□