Repeated Inverse Reinforcement Learning

Kareem Amin[*]
Google Research
New York, NY 10011
`kamin@google.com`

Nan Jiang[*] **Satinder Singh**
Computer Science & Engineering,
University of Michigan, Ann Arbor, MI 48104
`{nanjiang,baveja}@umich.edu`

Abstract

We introduce a novel *repeated* Inverse Reinforcement Learning problem: the agent has to act on behalf of a human in a sequence of tasks and wishes to minimize the number of tasks that it surprises the human by acting suboptimally with respect to how the human would have acted. Each time the human is surprised, the agent is provided a demonstration of the desired behavior by the human. We formalize this problem, including how the sequence of tasks is chosen, in a few different ways and provide some foundational results.

1 Introduction

One challenge in building AI agents that learn from experience is how to set their goals or rewards. In the Reinforcement Learning (RL) setting, one interesting answer to this question is inverse RL (or IRL) in which the agent infers the rewards of a human by observing the human's policy in a task [2]. Unfortunately, the IRL problem is ill-posed for there are typically many reward functions for which the observed behavior is optimal in a single task [3]. While the use of heuristics to select from among the set of feasible reward functions has led to successful applications of IRL to the problem of learning from demonstration [e.g., 4], not identifying the reward function poses fundamental challenges to the question of how well and how safely the agent will perform when using the learned reward function in other tasks.

We formalize multiple variations of a *new repeated IRL problem* in which the agent and (the same) human face multiple tasks over time. We separate the reward function into two components, one which is invariant across tasks and can be viewed as intrinsic to the human, and a second that is task specific. As a motivating example, consider a human doing tasks throughout a work day, e.g., getting coffee, driving to work, interacting with co-workers, and so on. Each of these tasks has a task-specific goal, but the human brings to each task intrinsic goals that correspond to maintaining health, financial well-being, not violating moral and legal principles, etc. In our repeated IRL setting, the agent presents a policy for each new task that it thinks the human would do. If the agent's policy "surprises" the human by being sub-optimal, the human presents the agent with the optimal policy. The objective of the agent is to minimize the number of surprises to the human, i.e., to generalize the human's behavior to new tasks.

In addition to addressing generalization across tasks, the repeated IRL problem we introduce and our results are of interest in resolving the question of unidentifiability of rewards from observations in standard IRL. Our results are also of interest to a particular aspect of the concern about how to make sure that the AI systems we build are safe, or AI safety. Specifically, the issue of reward misspecification is often mentioned in AI safety articles [e.g., 5, 6, 7]. These articles mostly discuss broad ethical concerns and possible research directions, while our paper develops mathematical formulations and algorithmic solutions to a specific way of addressing reward misspecification.

[*]This paper extends an unpublished arXiv paper by the authors [1].
[*]Equal contribution.

In summary form, our contributions include: (1) an efficient reward-identification algorithm when the agent can choose the tasks in which it observes human behavior; (2) an upper bound on the number of total surprises when no assumptions are made on the tasks, along with a corresponding lower bound; (3) an extension to the setting where the human provides sample trajectories instead of complete behavior; and (4) identification guarantees when the agent can only choose the task rewards but is given a fixed task environment.

2 Markov Decision Processes (MDPs)

An MDP is specified by its state space $\mathcal{S}$, action space $\mathcal{A}$, initial state distribution $\mu \in \Delta(\mathcal{S})$, transition function (or dynamics) $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, reward function $Y : \mathcal{S} \to \mathbb{R}$, and discount factor $\gamma \in [0, 1)$. We assume finite $\mathcal{S}$ and $\mathcal{A}$, and $\Delta(\mathcal{S})$ is the space of all distributions over $\mathcal{S}$. A policy $\pi : \mathcal{S} \to \mathcal{A}$ describes an agent's behavior by specifying the action to take in each state. The (normalized) value function or long-term utility of π is defined as $V^\pi(s) = (1-\gamma)\,\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} Y(s_t) | s_0 = s; \pi]$.[2] Similarly, the Q-value function is $Q^\pi(s,a) = (1-\gamma)\,\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} Y(s_t) | s_0 = s, a_0 = a; \pi]$. Where necessary we will use the notation $V^\pi_{P,Y}$ to avoid ambiguity about the dynamics and the reward function. Let $\pi^\star : \mathcal{S} \to \mathcal{A}$ be an optimal policy, which maximizes V^π and Q^π in all states (and actions) simultaneously.

Given an initial distribution over states, μ, a scalar value that measures the goodness of π is defined as $\mathbb{E}_{s\sim\mu}[V^\pi(s)]$. We introduce some further notation to express $\mathbb{E}_{s\sim\mu}[V^\pi(s)]$ in vector-matrix form. Let $\eta^\pi_{\mu,P} \in \mathbb{R}^{|\mathcal{S}|}$ be the *normalized state occupancy* under initial distribution μ, dynamics P, and policy π, whose s-th entry is $(1-\gamma)\,\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} \mathbb{I}(s_t = s) | s_0 \sim \mu; \pi]$ ($\mathbb{I}(\cdot)$ is the indicator function). This vector can be computed in closed-form as $\eta^\pi_{\mu,P} = (1-\gamma)\left(\mu^\top P^\pi \left(\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi\right)^{-1}\right)^\top$, where P^π is an $|\mathcal{S}| \times |\mathcal{S}|$ matrix whose (s, s')-th element is $P(s'|s, \pi(s))$, and $\mathbf{I}_{|\mathcal{S}|}$ is the $|\mathcal{S}| \times |\mathcal{S}|$ identity matrix. For convenience we will also treat the reward function Y as a vector in $\mathbb{R}^{|\mathcal{S}|}$, and we have

$$\mathbb{E}_{s\sim\mu}[V^\pi(s)] = Y^\top \eta^\pi_{\mu,P}. \tag{1}$$

3 Problem setup

Here we define the **repeated IRL problem**. The human's reward function $\theta_\star$ captures his/her safety concerns and intrinsic/general preferences. This $\theta_\star$ is unknown to the agent and is the object of interest herein, i.e., if $\theta_\star$ were known to the agent, the concerns addressed in this paper would be solved. We assume that the human cannot directly communicate $\theta_\star$ to the agent but can evaluate the agent's behavior in a task as well as demonstrate optimal behavior. Each task comes with an external reward function R, and the goal is to maximize the reward with respect to $Y := \theta_\star + R$ in each task.

As a concrete example, consider an agent for an autonomous vehicle. In this case, $\theta_\star$ represents the cross-task principles that define good driving (e.g., courtesy towards pedestrians and other vehicles), which are often difficult to explicitly describe. In contrast, R, the task-specific reward, could reward the agent for successfully completing parallel parking. While R is easier to construct, it may not completely capture what a human deems good driving. (For example, an agent might successfully parallel park while still boxing in neighboring vehicles.)

More formally, a task is defined by a pair (E, R), where $E = (\mathcal{S}, \mathcal{A}, \mu, P, \gamma)$ is the task environment (i.e., a controlled Markov process) and R is the task-specific reward function (*task reward*). We assume that all tasks share the same $\mathcal{S}, \mathcal{A}, \gamma$, with $|\mathcal{A}| \geq 2$, but **may** differ in the initial distribution μ, dynamics P, and task reward R; all of the task-specifying quantities are known to the agent. In any task, the human's optimal behavior is always with respect to the reward function $Y = \theta_\star + R$. We emphasize again that $\theta_\star$ is intrinsic to the human and remains the same across all tasks. Our use of task specific reward functions R allows for **greater generality** than the usual IRL setting, and most of our results apply equally to the case where $R \equiv \mathbf{0}$.

While $\theta_\star$ is private to the human, the agent has some prior knowledge on $\theta_\star$, represented as a set of possible parameters $\Theta_0 \subset \mathbb{R}^{|\mathcal{S}|}$ that contains $\theta_\star$. Throughout, we assume that the human's reward has bounded and normalized magnitude, that is, $\|\theta_\star\|_\infty \leq 1$.

[2]Here we differ (w.l.o.g.) from common IRL literature in assuming that reward occurs after transition.

A demonstration in (E, R) reveals $\pi^\star$, optimal for $Y = \theta_\star + R$ under environment E, to the agent. A common assumption in the IRL literature is that the full mapping is revealed, which can be unrealistic if some states are unreachable from the initial distribution. We address the issue by requiring only the state occupancy vector $\eta^{\pi^*}_{\mu,P}$. In Section 7 we show that this also allows an easy extension to the setting where the human only demonstrates trajectories instead of providing a policy.

Under the above framework for repeated IRL, we consider two settings that differ in how the sequence of tasks are chosen. In both settings, we will want to minimize the number of demonstrations needed.

1. (Section 5) *Agent chooses the tasks*, observes the human's behavior in each of them, and infers the reward function. In this setting where the agent is powerful enough to choose tasks arbitrarily, we will show that the agent will be able to *identify* the human's reward function which of course implies the ability to generalize to new tasks.

2. (Section 6) *Nature chooses the tasks*, and the agent proposes a policy in each task. The human demonstrates a policy only if the agent's policy is significantly suboptimal (i.e., **a mistake**). In this setting we will derive upper and lower bounds on the number of mistakes our agent will make.

4 The challenge of identifying rewards

Note that it is impossible to identify $\theta_\star$ from watching human behavior in a single task. This is because any $\theta_\star$ is fundamentally indistinguishable from an infinite set of reward functions that yield exactly the policy observed in the task. We introduce the idea of *behavioral equivalence* below to tease apart two separate issues wrapped up in the challenge of identifying rewards.

Definition 1. Two reward functions $\theta, \theta' \in \mathbb{R}^{|\mathcal{S}|}$ are *behaviorally equivalent in all MDP tasks*, if for any (E, R), the set of optimal policies for $(R + \theta)$ and $(R + \theta')$ are the same.

We argue that the task of identifying the reward function should amount only to identifying the (behavioral) equivalence class to which $\theta_\star$ belongs. In particular, identifying the equivalence class is sufficient to get perfect generalization to new tasks. Any remaining unidentifiability is merely representational and of no real consequence. Next we present a constraint that captures the reward functions that belong to the same equivalence class.

Proposition 1. *Two reward functions θ and θ' are behaviorally equivalent in all MDP tasks if and only if $\theta - \theta' = c \cdot \mathbf{1}_{|\mathcal{S}|}$ for some $c \in \mathbb{R}$, where $\mathbf{1}_{|\mathcal{S}|}$ is an all-1 vector of length $|\mathcal{S}|$.*

The proof is elementary and deferred to Appendix A. For any class of θ's that are equivalent to each other, we can choose a canonical element to represent this class. For example, we can fix an arbitrary reference state $s_{\text{ref}} \in \mathcal{S}$, and fix the reward of this state to 0 for $\theta_\star$ and all candidate θ's. In the rest of the paper, we will always assume such canonicalization in the MDP setting, hence $\theta_\star \in \Theta_0 \subseteq \{\theta \in [-1, 1]^{|\mathcal{S}|} : \theta(s_{\text{ref}}) = 0\}$.

5 Agent chooses the tasks

In this section, the protocol is that the agent chooses a sequence of tasks $\{(E_t, R_t)\}$. For each task (E_t, R_t), the human reveals $\pi_t^\star$, which is optimal for environment E_t and reward function $\theta_\star + R_t$. Our goal is to design an algorithm which chooses $\{(E_t, R_t)\}$ and identifies $\theta_\star$ to a desired accuracy, ϵ, using as few tasks as possible. Theorem 1 shows that a simple algorithm can identify $\theta_\star$ after only $O(\log(1/\epsilon))$ tasks, if *any* tasks may be chosen. Roughly speaking, the algorithm amounts to a binary search on each component of $\theta_\star$ by manipulating the task reward R_t.[3] See the proof for the algorithm specification. As noted before, once the agent has identified $\theta_\star$ within an appropriate tolerance, it can compute a sufficiently-near-optimal policy for all tasks, thus completing the generalization objective through the far stronger identification objective in this setting.

Theorem 1. *If $\theta_\star \in \Theta_0 \subseteq \{\theta \in [-1, 1]^{|\mathcal{S}|} : \theta(s_{ref}) = 0\}$, there exists an algorithm that outputs $\theta \in \mathbb{R}^{|\mathcal{S}|}$ that satisfies $\|\theta - \theta_\star\|_\infty \le \epsilon$ after $O(\log(1/\epsilon))$ demonstrations.*

Proof. The algorithm chooses the following fixed environment in all tasks: for each $s \in \mathcal{S} \setminus \{s_{\text{ref}}\}$, let one action be a self-loop, and the other action transitions to s_{ref}. In s_{ref}, all actions cause self-loops.

[3]While we present a proof that manipulates R_t, an only slightly more complex proof applies to the setting where all the R_t are exactly zero and the manipulation is limited to the environment [1].

The initial distribution over states is uniformly at random over $\mathcal{S} \setminus \{s_{\text{ref}}\}$. Each task only differs in the task reward R_t (where $R_t(s_{\text{ref}}) \equiv 0$ always). After observing the state occupancy of the optimal policy, for each s we check if the occupancy is equal to 0. If so, it means that the demonstrated optimal policy chooses to go to s_{ref} from s in the first time step, and $\theta_\star(s) + R_t(s) \le \theta_\star(s_{\text{ref}}) + R_t(s_{\text{ref}}) = 0$; if not, we have $\theta_\star(s) + R_t(s) \ge 0$. Consequently, after each task we learn the relationship between $\theta_\star(s)$ and $-R_t(s)$ on each $s \in \mathcal{S} \setminus \{s_{\text{ref}}\}$, so conducting a binary search by manipulating $R_t(s)$ will identify $\theta_\star$ to ϵ-accuracy after $O(\log(1/\epsilon))$ tasks. $\square$

6 Nature chooses the tasks

While Theorem 1 yields a strong identification guarantee, it also relies on a strong assumption, that $\{(E_t, R_t)\}$ may be chosen by the agent in an arbitrary manner. In this section, we let *nature*, who is allowed to be adversarial for the purpose of the analysis, choose $\{(E_t, R_t)\}$.

Generally speaking, we cannot obtain identification guarantees in such an adversarial setup. As an example, if $R_t \equiv 0$ and E_t remains the same over time, we are essentially back to the classical IRL setting and suffer from the degeneracy issue. However, generalization to future tasks, which is our ultimate goal, is easy in this special case: after the initial demonstration, the agent can mimic it to behave optimally in all subsequent tasks without requiring further demonstrations. More generally, if nature repeats similar tasks, then the agent obtains little new information, but presumably it knows how to behave in most cases; if nature chooses a task unfamiliar to the agent, then the agent is likely to err, but it may learn about $\theta_\star$ from the mistake.

To formalize this intuition, we consider the following protocol: the nature chooses a sequence of tasks $\{(E_t, R_t)\}$ in an arbitrary manner. For every task (E_t, R_t), the agent proposes a policy π_t. The human examines the policy's value under μ_t, and if the loss

$$l_t = \mathbb{E}_{s\sim\mu}\left[V^{\pi_t^\star}_{E_t,\,\theta_\star+R_t}(s)\right] - \mathbb{E}_{s\sim\mu}\left[V^{\pi_t}_{E_t,\,\theta_\star+R_t}(s)\right] \tag{2}$$

is less than some ϵ then the human is satisfied and no demonstration is needed; otherwise a mistake is counted and $\eta^{\pi_t^\star}_{\mu_t,P_t}$ is revealed to the agent (note that $\eta^{\pi_t^\star}_{\mu_t,P_t}$ can be computed by the agent if needed from π_t^* and its knowledge of the task). The main goal of this section is to design an algorithm that has a provable guarantee on the total number of mistakes.

On human supervision Here we require the human to evaluate the agent's policies in addition to providing demonstrations. We argue that this is a reasonable assumption because (1) only a binary signal $\mathbb{I}(l_t > \epsilon)$ is needed as opposed to the precise value of l_t, and (2) if a policy is suboptimal but the human fails to realize it, arguably it should not be treated as a mistake. Meanwhile, we will also provide identification guarantees in Section 6.4, as the human will be relieved from the supervision duty once $\theta_\star$ is identified.

Before describing and analyzing our algorithm, we first notice that the Equation 2 can be rewritten as

$$l_t = (\theta_\star + R)^\top (\eta^{\pi_t^\star}_{\mu_t,P_t} - \eta^{\pi_t}_{\mu_t,P_t}), \tag{3}$$

using Equation 1. So effectively, the given environment E_t in each round induces a set of state occupancy vectors $\{\eta^{\pi}_{\mu_t,P_t} : \pi \in (\mathcal{S} \to \mathcal{A})\}$, and we want the agent to choose the vector that has the largest dot product with $\theta_\star + R$. The exponential size of the set will not be a concern because our main result (Theorem 2) has no dependence on the number of vectors, and only depends on the dimension of those vectors. The result is enabled by studying the *linear bandit* version of the problem, which subsumes the MDP setting for our purpose and is also a model of independent interest.

6.1 The linear bandit setting

In the linear bandit setting, $\mathcal{D}$ is a finite action space with size $|\mathcal{D}| = K$. Each task is denoted as a pair (X, R), where R is the task specific reward function as before. $X = [x^{(1)} \cdots x^{(K)}]$ is a $d \times K$ feature matrix, where $x^{(i)}$ is the feature vector for the i-th action, and $\|x^{(i)}\|_1 \le 1$. When we reduce MDPs to linear bandits, each element of $\mathcal{D}$ corresponds to an MDP policy, and the feature vector is the state occupancy of that policy.

As before, $R, \theta_\star \in \mathbb{R}^d$ are the task reward and the human's unknown reward, respectively. The initial uncertainty set for $\theta_\star$ is $\Theta_0 \subseteq [-1, 1]^d$. The value of the i-th action is calculated as $(\theta_\star + R)^\top x^{(i)}$,

Algorithm 1 Ellipsoid Algorithm for Repeated Inverse Reinforcement Learning

```
1: Input: $\Theta_0$.
2: $\Theta_1 \leftarrow$ MVEE($\Theta_0$).
3: for $t = 1, 2, \ldots$ do
4:     Nature reveals $(X_t, R_t)$.
5:     Learner plays $a_t = \arg\max_{a \in \mathcal{D}} c_t^\top x_t^a$, where $c_t$ is the center of $\Theta_t$. $\Theta_{t+1} \leftarrow \Theta_t$.
6:     if $l_t > \epsilon$ then
7:         Human reveals $a_t^\star$.  $\Theta_{t+1} \leftarrow$ MVEE($\{\theta \in \Theta_t : (\theta - c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) \geq 0\}$).
8:     end if
9: end for
```

and $a^\star$ is the action that maximizes this value. Every round the agent proposes an action $a \in \mathcal{D}$, whose loss is defined as

$$l_t = (\theta_\star + R)^\top (x^{a^\star} - x^a).$$

We now show how to embed the previous MDP setting in linear bandits.

Example 1. Given an MDP problem with variables $\mathcal{S}, \mathcal{A}, \gamma, \theta_\star, s_{\text{ref}}, \Theta_0, \{(E_t, R_t)\}$, we can convert it into a linear bandit problem as follows: (all variables with prime belong to the linear bandit problem, and we use $v^{\setminus i}$ to denote the vector v with the i-th coordinate removed)

- $\mathcal{D} = \{\pi : \mathcal{S} \to \mathcal{A}\}$, $d = |\mathcal{S}| - 1$, $\theta'_\star = \theta_\star^{\setminus s_{\text{ref}}}$, $\Theta'_0 = \{\theta^{\setminus s_{\text{ref}}} : \theta \in \Theta_0\}$.
- $x_t^\pi = (\eta_{\mu_t, P_t}^\pi)^{\setminus s_{\text{ref}}}$. $R'_t = R_t^{\setminus s_{\text{ref}}} - R_t(s_{\text{ref}}) \cdot \mathbf{1}_d$.

Note that there is a more straightforward conversion by letting $d = |\mathcal{S}|, \theta'_\star = \theta_\star, \Theta'_0 = \Theta_0, x_t^\pi = \eta_{\mu_t, P_t}^\pi, R'_t = R_t$, which also preserves losses. We perform a more succinct conversion in Example 1 by canonicalizing both $\theta_\star$ (already assumed) and R_t (explicitly done here) and dropping the coordinate for s_{ref} in all relevant vectors.

MDPs with linear rewards In IRL literature, a generalization of the MDP setting is often considered, that reward is linear in state features $\phi(s) \in \mathbb{R}^d$ [2, 3]. In this new setting, $\theta_\star$ and R are reward parameters, and the actual reward is $(\theta_\star + R)^\top \phi(s)$. This new setting can also be reduced to linear bandits similarly to Example 1, except that the state occupancy is replaced by the discounted sum of expected feature values. Our main result, Theorem 2, will still apply automatically, but now the guarantee will only depend on the dimension of the feature space and has no dependence on $|\mathcal{S}|$. We include the conversion below but do not further discuss this setting in the rest of the paper.

Example 2. Consider an MDP problem with state features, defined by $\mathcal{S}, \mathcal{A}, \gamma, d \in \mathbb{Z}^+, \theta_\star \in \mathbb{R}^d, \Theta_0 \subseteq [-1, 1]^d, \{(E_t, \phi_t \in \mathbb{R}^d, R_t \in \mathbb{R}^d)\}$, where task reward and background reward in state s are $\theta_\star^\top \phi_t(s)$ and $R^\top \phi_t(s)$ respectively, and $\theta_\star \in \Theta_0$. Suppose $\|\phi_t(s)\|_\infty \leq 1$ always holds, then we can convert it into a linear bandit problem as follows: $\mathcal{D} = \{\pi : \mathcal{S} \to \mathcal{A}\}$. d, $\theta_\star$, and R_t remain the same. $x_t^\pi = (1 - \gamma) \sum_{h=1}^\infty \gamma^{h-1} \mathbb{E}[\phi(s_h) \mid \mu_t, P_t, \pi]/d$. Note that the division of d in x_t^π is for the purpose of normalization, so that $\|x_t^\pi\|_1 \leq \|\phi\|_1/d \leq \|\phi\|_\infty \leq 1$.

6.2 Ellipsoid Algorithm for Repeated Inverse Reinforcement Learning

We propose Algorithm 1, and provide the mistake bound in the following theorem.

Theorem 2. *For $\Theta_0 = [-1, 1]^d$, the number of mistakes made by Algorithm 1 is guaranteed to be $O(d^2 \log(d/\epsilon))$.*

To prove Theorem 2, we quote a result from linear programming literature in Lemma 1, which is found in standard lecture notes (e.g., [8], Theorem 8.8; see also [9], Lemma 3.1.34).

Lemma 1 (Volume reduction in ellipsoid algorithm)**.** *Given any non-degenerate ellipsoid B in $\mathbb{R}^d$ centered at $c \in \mathbb{R}^d$, and any non-zero vector $v \in \mathbb{R}^d$, let B^+ be the minimum-volume enclosing ellipsoid (MVEE) of $\{u \in B : (u - c)^\top v \geq 0\}$. We have $vol(B^+)/vol(B) \leq e^{-\frac{1}{2(d+1)}}$.*

Proof of Theorem 2. Whenever a mistake is made, we can induce the constraint $(R_t + \theta_\star)^\top (x_t^{a_t^\star} - x_t^{a_t}) > \epsilon$. Meanwhile, since a_t is greedy w.r.t. c_t, we have $(R_t + c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) \leq 0$, where c_t is the center of Θ_t as in Line 5. Taking the difference of the two inequalities, we obtain

$$(\theta_\star - c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) > \epsilon. \tag{4}$$

Therefore, the update rule on Line 7 of Algorithm 1 preserves $\theta_\star$ in Θ_{t+1}. Since the update makes a central cut through the ellipsoid, Lemma 1 applies and the volume shrinks every time a mistake is made. To prove the theorem, it remains to upper bound the initial volume and lower bound the terminal volume of Θ_t. We first show that an update never eliminates $B_\infty(\theta_\star, \epsilon/2)$, the ℓ_∞ ball centered at $\theta_\star$ with radius $\epsilon/2$. This is because, any eliminated θ satisfies $(\theta + c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) < 0$. Combining this with Equation 4, we have

$$\epsilon < (\theta^\star - \theta)^\top (x_t^{a_t^\star} - x_t^{a_t}) \leq \|\theta_\star - \theta\|_\infty \|x_t^{a_t^\star} - x_t^{a_t}\|_1 \leq 2\|\theta_\star - \theta\|_\infty.$$

The last step follows from $\|x\|_1 \leq 1$. We conclude that any eliminated θ should be $\epsilon/2$ far away from $\theta_\star$ in ℓ_∞ distance. Hence, we can lower bound the volume of Θ_t for any t by that of $\Theta_0 \bigcap B_\infty(\theta_\star, \epsilon/2)$, which contains an ℓ_∞ ball with radius $\epsilon/4$ at its smallest (when $\theta_\star$ is one of Θ_0's vertices). To simplify calculation, we relax this lower bound (volume of the ℓ_∞ ball) to the volume of the inscribed ℓ_2 ball.

Finally we put everything together: let M_T be the number of mistakes made from round 1 to T, C_d be the volume of the unit hypersphere in $\mathbb{R}^d$ (i.e., ℓ_2 ball with radius 1), and vol$(\cdot)$ denote the volume of an ellipsoid, we have

$$\frac{M_T}{2(d+1)} \leq \log(\text{vol}(\Theta_1)) - \log(\text{vol}(\Theta_{T+1})) \leq \log(C_d(\sqrt{d})^d) - \log(C_d(\epsilon/4)^d) = d \log \frac{4\sqrt{d}}{\epsilon}.$$

So $M_T \leq 2d(d+1) \log \frac{4\sqrt{d}}{\epsilon} = O(d^2 \log \frac{d}{\epsilon})$. □

6.3 Lower bound

In Section 5, we get an $O(\log(1/\epsilon))$ upper bound on the number of demonstrations, which has no dependence on $|\mathcal{S}|$ (which corresponds to $d+1$ in linear bandits). Comparing Theorem 2 to 1, one may wonder whether the polynomial dependence on d is an artifact of the inefficiency of Algorithm 1. We clarify this issue by proving a lower bound, showing that $\Omega(d \log(1/\epsilon))$ mistakes are inevitable in the worst case when nature chooses the tasks. We provide a proof sketch below, and the complete proof is deferred to Appendix E.

Theorem 3. *For any randomized algorithm*[4] *in the linear bandit setting, there always exists $\theta_\star \in [-1,1]^d$ and an adversarial sequence of $\{(X_t, R_t)\}$ that potentially adapts to the algorithm's previous decisions, such that the expected number of mistakes made by the algorithm is $\Omega(d \log(1/\epsilon))$.*

Proof Sketch. We randomize $\theta_\star$ by sampling each element i.i.d. from Unif$([-1,1])$. We will prove that there exists a strategy of choosing (X_t, R_t) such that any algorithm's expected number of mistakes is $\Omega(d \log(1/\epsilon)$, which proves the theorem as max is no less than average.

In our construction, $X_t = [\mathbf{0}_d,\ \boldsymbol{e}_{j_t}]$, where j_t is some index to be specified. Hence, every round the agent is essentially asked to decided whether $\theta(j_t) \geq -R_t(j_t)$. The adversary's strategy goes in phases, and R_t remains the same during each phase. Every phase has d rounds where j_t is enumerated over $\{1, \ldots, d\}$.

The adversary will use R_t to shift the posterior on $\theta(j_t) + R_t(j_t)$ so that it is centered around the origin; in this way, the agent has about $1/2$ probability to make an error (regardless of the algorithm), and the posterior interval will be halved. Overall, the agent makes $d/2$ mistakes in each phase, and there will be about $\log(1/\epsilon)$ phases in total, which gives the lower bound. □

Applying the lower bound to MDPs The above lower bound is stated for linear bandits. In principle, we need to prove lower bound for MDPs separately, because linear bandits are more general than MDPs for our purpose, and the hard instances in linear bandits may not have corresponding MDP instances. In Lemma 2 below, we show that a certain type of linear bandit instances can always be emulated by MDPs with the same number of actions, and the hard instances constructed in

[4]While our Algorithm 1 is deterministic, randomization is often crucial for online learning in general [10].

Theorem 3 indeed satisfy the conditions for such a type; in particular, we require the feature vectors to be non-negative and have ℓ_1 norm bounded by 1. As a corollary, an $\Omega(|\mathcal{S}|\log(1/\epsilon))$ lower bound for the MDP setting (even with a small action space $|\mathcal{A}| = 2$) follows directly from Theorem 3. The proof of Lemma 2 is deferred to Appendix B.

Lemma 2 (Linear bandit to MDP conversion)**.** *Let (X, R) be a linear bandit task, and K be the number of actions. If every x^a is non-negative and $\|x^a\|_1 \leq 1$, then there exists an MDP task (E, R') with $d+1$ states and K actions, such that under some choice of s_{ref}, converting (E, R') as in Example 1 recovers the original problem.*

6.4 On identification when nature chooses tasks

While Theorem 2 successfully controls the number of total mistakes, it completely avoids the identification problem and does not guarantee to recover $\theta_\star$. In this section we explore further conditions under which we can obtain identification guarantees when Nature chooses the tasks.

The first condition, stated in Proposition 2, implies that if we have made all the possible mistakes, then we have indeed identified the $\theta_\star$, where the identification accuracy is determined by the tolerance parameter ϵ that defines what is counted as a mistake. Due to space limit, the proof is deferred to Appendix C.

Proposition 2. *Consider the linear bandit setting. If there exists T_0 such that for any round $t \geq T_0$, no more mistakes can be ever made by the algorithm for any choice of (E_t, R_t) and any tie-braking mechanism, then we have $\theta_\star \in B_\infty(c_{T_0}, \epsilon)$.*

While the above proposition shows that identification is guaranteed if the agent exhausts the mistakes, the agent has no ability to actively fulfill this condition when nature chooses tasks. For a stronger identification guarantee, we may need to grant the agent some freedom in choosing the tasks.

Identification with fixed environment Here we consider a setting that fits in between Section 5 (completely active) and Section 6.1 (completely passive), where the environment E (hence the induced feature vectors $\{x^{(1)}, x^{(2)}, \ldots, x^{(K)}\}$) is given and fixed, and the agent can arbitrarily choose the task reward R_t. The goal is to obtain identification guarantee in this intermediate setting.

Unfortunately, a degenerate case can be easily constructed that prevents the revelation of any information about $\theta_\star$. In particular, if $x^{(1)} = x^{(2)} = \ldots = x^{(K)}$, i.e., the environment is completely uncontrolled, then all actions are equally optimal and nothing can be learned. More generally, if for some $v \neq \mathbf{0}$ we have $v^\top x^{(1)} = v^\top x^{(2)} = \ldots = v^\top x^{(K)}$, then we may never recover $\theta_\star$ along the direction of v. In fact, Proposition 1 can be viewed as an instance of this result where $v = \mathbf{1}_{|\mathcal{S}|}$ (recall that $\mathbf{1}_{|\mathcal{S}|}^\top \eta_{\mu,P}^\pi \equiv 1$), and that is why we have to remove such redundancy in Example 1 in order to discuss identification in MDPs. Therefore, to guarantee identification in a fixed environment, the feature vectors must have significant variation in all directions, and we capture this intuition by defining a diversity score spread(X) (Definition 2) and showing that the identification accuracy depends inversely on the score (Theorem 4).

Definition 2. Given the feature matrix $X = \begin{bmatrix} x^{(1)} & x^{(2)} & \cdots & x^{(K)} \end{bmatrix}$ whose size is $d \times K$, define spread(X) as the d-th largest singular value of $\widetilde{X} := X(\mathbf{I}_K - \frac{1}{K}\mathbf{1}_K\mathbf{1}_K^\top)$.

Theorem 4. *For a fixed feature matrix X, if spread$(X) > 0$, then there exists a sequence $R_1, R_2, \ldots, R_T$ with $T = O(d^2 \log(d/\epsilon))$ and a sequence of tie-break choices of the algorithm, such that after round T we have $\|c_T - \theta_\star\|_\infty \leq \epsilon\sqrt{(K-1)/2}/\text{spread}(X)$.*

The proof is deferred to Appendix D. The $\sqrt{K}$ dependence in Theorem 4 may be of concern as K can be exponentially large. However, Theorem 4 also holds if we replace X by any matrix that consists of X's columns, so we may choose a small yet most diverse set of columns as to optimize the bound.

7 Working with trajectories

In previous sections, we have assumed that the human evaluates the agent's performance based on the state occupancy of the agent's policy, and demonstrates the optimal policy in terms of state occupancy as well. In practice, we would like to instead assume that for each task, the agent rolls out a trajectory, and the human shows an optimal trajectory if he/she finds the agent's trajectory unsatisfying. We

Algorithm 2 Trajectory version of Algorithm 1 for MDPs

```
1: Input: $\Theta_0, H, n$.
2: $\Theta_1 \leftarrow$ MVEE$(\Theta_0)$, $i \leftarrow 0$, $\bar{Z} \leftarrow 0$, $\bar{Z}^\star \leftarrow 0$.
3: for $t = 1, 2, \ldots$ do
4:     Nature reveals $(E_t, R_t)$. Agent rolls-out a trajectory using $\pi_t$ greedily w.r.t. $c_t + R_t$.
5:     $\Theta_{t+1} \leftarrow \Theta_t$.
6:     if agent takes $a$ in $s$ with $Q^\star(s,a) < V^\star(s) - \epsilon$ then
7:         Human produces an $H$-step trajectory from $s$. Let the empirical state occupancy be $\hat{z}_i^{\star,H}$.
8:         $i \leftarrow i+1$, $\bar{Z}^\star \leftarrow \bar{Z}^\star + \hat{z}_i^{\star,H}$.
9:         Let $z_i$ be the state occupancy of $\pi_t$ from initial state $s$, and $\bar{Z} \leftarrow \bar{Z} + z_i$.
10:        if $i = n$ then
11:            $\Theta_{t+1} \leftarrow$ MVEE$(\{\theta \in \Theta_t : (\theta - c_t)^\top(\bar{Z}^\star - \bar{Z}) \geq 0\})$.  $i \leftarrow 0$, $\bar{Z} \leftarrow 0$, $\bar{Z}^\star \leftarrow 0$.
12:        end if
13:    end if
14: end for
```

are still concerned about upper bounding the number of total mistakes, and aim to provide a parallel version of Theorem 2.

Unlike in traditional IRL, in our setting the agent is also acting, which gives rise to many subtleties. First, the total reward on the agent's single trajectory is a random variable, and may deviate from the expected value of its policy. Therefore, it is generally impossible to decide if the agent's policy is near-optimal, and instead we assume that the human can check if each action that the agent takes in the trajectory is near-optimal: when the agent takes a at state s, an error is counted if and only if $Q^\star(s,a) < V^\star(s) - \epsilon$. This criterion can be viewed as a noisy version of the one used in previous sections, as taking expectation of $V^\star(s) - Q^\star(s, \pi(s))$ over the occupancy induced by π will recover Equation 2.

While this resolves the issue on the agent's side, how should the human provide his/her optimal trajectory? The most straightforward protocol is that the human rolls out a trajectory from the initial distribution of the task, μ_t. We argue that this is not a reasonable protocol for two reasons: (1) in expectation, the reward collected by the human may be less than that by the agent, because conditioning on the event that an error is spotted may introduce a selection bias; (2) the human may not encounter the problematic state in his/her own trajectory, hence the information provided in the trajectory may be irrelevant.

To resolve this issue, we consider a different protocol where the human rolls out a trajectory using an optimal policy from the very state where the agent errs.

Now we discuss how we can prove a parallel of Theorem 2 under this new protocol. First, let's assume that the demonstration were still given in the form a state occupancy vector starting at the problematic state. In this case, we can reduce to the setting of Section 6 by changing μ_t to a point mass on the problematic state.[5] To apply the algorithm and the analysis in Section 6, it remains to show that the notion of error in this section (a suboptimal action) implies the notion of error in Section 6 (a suboptimal policy): let s be the problematic state and π be the agent's policy, we have $V^\pi(s) = Q^\pi(s, \pi(s)) \leq Q^\star(s, \pi(s)) < V^\star(s) - \epsilon$. So whenever a suboptimal *action* is spotted in state s, it indeed implies that the agent's *policy* is suboptimal for s as the initial state. Hence, we can run Algorithm 1 as-is and Theorem 2 immediately applies.

To tackle the remaining issue that the demonstration is in terms of a single trajectory, we will not update Θ_t after each mistake as in Algorithm 1, but only make an update after every mini-batch of mistakes, and aggregate them to form accurate update rules. See Algorithm 2. The formal guarantee of the algorithm is stated in Theorem 5, whose proof is deferred to Appendix G.

[5]At the first glance this might seem suspicious: the problematic state is random and depends on the learner's current policy, but in RL the initial distribution is usually fixed and the learner has no control over it. This concern is removed thanks to our adversarial setup on (E_t, R_t) (of which μ_t is a component).

Theorem 5. $\forall \delta \in (0,1)$, *with probability at least* $1-\delta$, *the number of mistakes made by Algorithm 2 with parameters* $\Theta_0 = [-1,1]^d$, $H = \left\lceil \frac{\log(12/\epsilon)}{1-\gamma} \right\rceil$, *and* $n = \left\lceil \frac{\log(\frac{4d(d+1)\log\frac{6\sqrt{d}}{\epsilon}}{\delta})}{32\epsilon^2} \right\rceil$ *where* $d = |\mathcal{S}|$,[6] *is at most* $\tilde{O}(\frac{d^2}{\epsilon^2}\log(\frac{d}{\delta\epsilon}))$.[7]

8 Related work & Conclusions

Most existing work in IRL focused on inferring the reward function[8] using data acquired from a fixed environment [2, 3, 16, 17, 18, 19, 20]. There is prior work on using data collected from multiple — but exogenously fixed — environments to predict agent behavior [21]. There are also applications where methods for single-environment MDPs have been adapted to multiple environments [17]. Nevertheless, all these works consider the objective of mimicking an optimal behavior in the presented environment(s), and do not aim at generalization to new tasks that is the main contribution of this paper. Recently, Hadfield et. al. [22] proposed cooperative inverse reinforcement learning, where the human and the agent act in the same environment, allowing the human to actively resolve the agent's uncertainty on the reward function. However, they only consider a single environment (or task), and the unidentifiability issue of IRL still exists. Combining their interesting framework with our resolution to unidentifiability (by multiple tasks) can be an interesting future direction.

Acknowledgement

This work was supported in part by NSF grant IIS 1319365 (Singh & Jiang) and in part by a Rackham Predoctoral Fellowship from the University of Michigan (Jiang). Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Kareem Amin and Satinder Singh. Towards resolving unidentifiability in inverse reinforcement learning. *arXiv preprint arXiv:1601.06569*, 2016.

[2] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670, 2000.

[3] Pieter Abbeel and Andrew Y Ng. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine learning*, page 1. ACM, 2004.

[4] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of reinforcement learning to aerobatic helicopter flight. *Advances in neural information processing systems*, 19:1, 2007.

[5] Nick Bostrom. Ethical issues in advanced artificial intelligence. *Science Fiction and Philosophy: From Time Travel to Superintelligence*, pages 277–284, 2003.

[6] Stuart Russell, Daniel Dewey, and Max Tegmark. Research priorities for robust and beneficial artificial intelligence. *AI Magazine*, 36(4):105–114, 2015.

[7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[6] Here we use the simpler conversion explained right after Example 1. We can certainly improve the dimension to $d = |\mathcal{S}| - 1$ by dropping the s_{ref} coordinate in all relevant vectors but that complicates presentation.

[7] A $\log\log(1/\epsilon)$ term is suppressed in $\tilde{O}(\cdot)$.

[8] While we do not discuss it here, in the economics literature, the problem of inferring an agent's utility from behavior-queries has long been studied under the heading of utility or preference elicitation [11, 12, 13, 14, 15]. While our result in Section 5 uses similar techniques to elicit the reward function, we do so purely by observing the human's behavior without external source of information (e.g., query responses).

[8] Ryan O'Donnell. *15-859(E) – linear and semidefinite programming: lecture notes*. Carnegie Mellon University, 2011. `https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15859-f11/www/notes/lecture08.pdf`.

[9] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[10] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

[11] Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *AAAI/IAAI*, pages 363–369, 2000.

[12] John Von Neumann and Oskar Morgenstern. *Theory of games and economic behavior (60th Anniversary Commemorative Edition)*. Princeton university press, 2007.

[13] Kevin Regan and Craig Boutilier. Regret-based reward elicitation for markov decision processes. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 444–451. AUAI Press, 2009.

[14] Kevin Regan and Craig Boutilier. Eliciting additive reward functions for markov decision processes. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 2159, 2011.

[15] Constantin A Rothkopf and Christos Dimitrakakis. Preference elicitation and inverse reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 34–48. Springer, 2011.

[16] Adam Coates, Pieter Abbeel, and Andrew Y Ng. Learning for control from multiple demonstrations. In *Proceedings of the 25th international conference on Machine learning*, pages 144–151. ACM, 2008.

[17] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008.

[18] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. *Urbana*, 51:61801, 2007.

[19] Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in neural information processing systems*, pages 1449–1456, 2007.

[20] Kevin Regan and Craig Boutilier. Robust policy computation in reward-uncertain mdps using nondominated policies. In *AAAI*, 2010.

[21] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 729–736. ACM, 2006.

[22] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3909–3917, 2016.

Appendix

A Proof of Proposition 1

To show that $\theta - \theta' = c \cdot \mathbf{1}_{|\mathcal{S}|}$ implies behavioral equivalence, we note that for any policy π the occupancy vector $\eta^{\pi}_{\mu,P}$ always satisfies $\mathbf{1}_{|\mathcal{S}|}^{\top}\eta^{\pi}_{\mu,P} = 1$, so $\forall \pi$, $|\theta^T \eta^{\pi}_{\mu,P} - \theta'^T \eta^{\pi}_{\mu,P}| = c$, and therefore the set of optimal policies is the same.

To show the other direction, we prove that if $\theta - \theta' \notin \text{span}(\{\mathbf{1}_{|\mathcal{S}|}\})$, then there exists (E, R) such that the sets of optimal policies differ. In particular, we choose $R = -\theta'$, so that all policies are optimal under $R + \theta' = 0$. Since $\theta - \theta' \notin \text{span}(\{\mathbf{1}_{|\mathcal{S}|}\})$, there exists states i and j such that $\theta(i) + R(i) \neq \theta(j) + R(j)$. Suppose i is the one with smaller sum of rewards, then we can make j an absorbing state, and have two deterministic actions in i that transition to i and j respectively. Under $R + \theta$, the self-loop in state i is suboptimal, and this completes the proof. □

B Proof of Lemma 2

The construction is as follows. Choose s_{ref} as the initial state, and make all other states absorbing. Let $R'(s_{\text{ref}}) = 0$ and R' restricted on $\mathcal{S} \setminus \{s_{\text{ref}}\}$ coincide with R. The remaining work is to design the transition distribution of each action in s_{ref} so that the induced state occupancy matches exactly one column of X.

Fixing any action a, and let x be the feature that we want to associate a with. The next-state distribution of (s_{ref}, a) is as follows: with probability $p = \frac{1-\|x\|_1}{1-\gamma\|x\|_1}$ the next-state is s_{ref} itself, and the probability of transitioning to the j-th state in $\mathcal{S} \setminus \{s_{\text{ref}}\}$ is $\frac{1-\gamma}{1-\gamma\|x\|_1}x(j)$. Given $\|x\|_1 \leq 1$ and $x \geq 0$, it is easy to verify that this is a valid distribution.

Now we calculate the occupancy of policy $\pi(s_{\text{ref}}) = a$. The normalized occupancy on s_{ref} is

$$(1-\gamma)(p + \gamma p^2 + \gamma^2 p^3 + \cdots) = \frac{p(1-\gamma)}{1-\gamma p} = 1 - \|x\|_1.$$

The remaining occupancy, with a total ℓ_1 mass of $\|x\|_1$, is split among $\mathcal{S} \setminus \{s_{\text{ref}}\}$ proportional to x. Therefore, when we convert the MDP problem as in Example 1, the corresponding feature vector is exactly x, so we recover the original linear bandit problem. □

C Proof of Proposition 2

Assume towards contradiction that $\|c_{T_0} - \theta_\star\|_\infty > \epsilon$. We will choose $(R_t, x_t^{(1)}, x_t^{(2)})$ to make the algorithm err. In particular, let $R_t = -c_{T_0}$, so that the algorithm acts greedily with respect to $\mathbf{0}_d$. Since $\mathbf{0}_d^\top x_t^a \equiv 0$, any action would be a valid choice for the algorithm.

On the other hand, $\|c_{T_0} - \theta_\star\|_\infty > \epsilon$ implies that there exists a coordinate j such that $|\boldsymbol{e}_j^\top(\theta_\star - c_{T_0})| > \epsilon$, where $\boldsymbol{e}_j$ is a basis vector. Let $x_t^{(1)} = \mathbf{0}_d$ and $x_t^{(2)} = \boldsymbol{e}_j$. So the value of action 1 is always 0 under any reward function (including $\theta_\star + R_t$), and the value of action 2 is $(\theta_\star + R_t)^\top x_t^{(2)} = (\theta_\star - c_{T_0})^\top \boldsymbol{e}_j$, whose absolute value is greater than ϵ. At least one of the 2 actions is more than ϵ suboptimal, and the algorithm may take any of them, so the algorithm can err again. □

D Proof of Theorem 4

It suffices to show that in any round t, if $\|c_t - \theta_\star\|_\infty > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$, then $l_t > \epsilon$. The bound on T follows directly from Theorem 2. Similar to the proof of Proposition 2, our choice of the task reward is $R_t = -c_t$, so that any $a \in A$ would be a valid choice of a_t, and we will choose the worst action. Note that $\forall a, a' \in \mathcal{D}$,

$$l_t = (\theta_\star + R_t)^\top (x^{a_t^\star} - x^{a_t}) \geq (\theta_\star - c_t)^\top (x^a - x^{a'}).$$

So it suffices to show that there exists $a, a' \in \mathcal{D}$, such that $(\theta_\star - c_t)^\top (x^a - x^{a'}) > \epsilon$. Let $y_t = \theta_\star - c_t$, and the precondition implies that $\|y_t\|_2 \geq \|y_t\|_\infty > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$.

Define a matrix D of size $K \times (K(K-1))$, where each column

$$D = \begin{bmatrix} 1 & 1 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & -1 \\ 0 & 0 & \cdots & 1 \end{bmatrix}. \tag{5}$$

contains exactly one 1 and one -1 (the remaining entries are 0), and the columns enumerate all possible positions of them. With the help of this matrix, we can rewrite the desired result ($\exists\, a, a' \in A$, s.t. $(\theta_\star - c_t)^\top (x^a - x^{a'}) > \epsilon$) as $\|y_t^\top XD\|_\infty \geq \epsilon$. We relax the LHS as $\|y_t^\top XD\|_\infty \geq \|y_t^\top XD\|_2 / \sqrt{K(K-1)}$, and will provide a lower bound on $\|y_t^\top XD\|_2$. Note that

$$y_t^\top XD = y_t^\top (\widetilde{X} + (X - \widetilde{X}))D = y_t^\top \widetilde{X} D,$$

because every row of $(X - \widetilde{X})$ is some multiple of $\mathbf{1}_K^\top$ (recall Definition 2), and every column of D is orthogonal to $\mathbf{1}_K$. Let $\widehat{(\cdot)}$ be the vector normalized to unit length,

$$\|y_t^\top \widetilde{X} D\|_2 = \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X} D\|_2 = \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X}\|_2 \|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2.$$

We lower bound each of the 3 terms. For the first term, we have the precondition $\|y_t\|_2 > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$. The second term is $\widetilde{X}$ left multiplied by a unit vector, so its ℓ_2 norm can be lower bounded by the smallest non-zero singular value of $\widetilde{X}$ (recall that $\widetilde{X}$ is full-rank), which is spread(X).

To lower bound the last term, note that $DD^\top = 2K\mathbf{I}_K - 2\mathbf{1}_K\mathbf{1}_K^\top$, and rows of $\widetilde{X}$ are orthogonal to $\mathbf{1}_K^\top$ and so is $y_t^\top \widetilde{X}$, so

$$\|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2^2 \geq \inf_{\|z\|_2 = 1,\, z \perp \mathbf{1}_K} z^\top DD^\top z = \inf_{\|z\|_2 = 1,\, z \perp \mathbf{1}_K} z^\top (2K\mathbf{I}_K - 2\mathbf{1}_K\mathbf{1}_K^\top) z = 2K.$$

Putting all the pieces together, we have

$$\|y_t^\top \widetilde{X} D\|_\infty \geq \frac{\|y_t\|_2 \|\hat{y}_t^\top \widetilde{X}\|_2 \|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2}{\sqrt{K(K-1)}} > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)} \cdot \text{spread}(X) \cdot \frac{\sqrt{2K}}{\sqrt{K(K-1)}} = \epsilon.$$

E Proof of Theorem 3

As a standard trick, we randomize $\theta_\star$ by sampling each element i.i.d. from Unif$([-1, 1])$. We will prove that there exists a strategy of choosing (X_t, R_t) such that any algorithm's expected number of mistakes is $\Omega(d \log(1/\epsilon)$, where the expectation is with respect to the randomness of $\theta_\star$ and the internal randomness of the algorithm. This immediately implies a worst-case result as max is no less than average (regarding the sampling of $\theta_\star$).

In our construction, $X_t = [\mathbf{0}_d,\ \boldsymbol{e}_{j_t}]$, where j_t is some index to be specified. Hence, every round the agent is essentially asked to decided whether $\theta(j_t) \geq -R_t(j_t)$. The adversary's strategy goes in phases, and R_t remains the same during each phase. Every phase has d rounds where j_t is enumerated over $\{1, \ldots, d\}$. To fully specify the nature's strategy, it remains to specify R_t for each phase.

In the 1st phase, $R_t \equiv 0$. For each coordinate j, the information revealed to the agent is one of the following: $\theta_\star(j) > \epsilon$, $\theta_\star(j) \geq -\epsilon$, $\theta_\star(j) < -\epsilon$, $\theta_\star(j) \leq \epsilon$. For clarity we first make an simplification, that the revealed information is either $\theta_\star(j) > 0$ or $\theta_\star(j) \leq 0$; we will deal with the subtleties related to ϵ at the end of the proof.

In the 2nd phase, we fix R_t as

$$R_t(j) = \begin{cases} -1/2 & \text{if } \theta_\star(j) \geq 0, \\ 1/2 & \text{if } \theta_\star(j) < 0. \end{cases}$$

Since $\theta_\star$ is randomized i.i.d. for each coordinate, the posterior of $\theta_\star + R_t$ conditioned on the revealed information is Unif$[-1/2, 1/2]$, for any algorithm and any interaction history. Therefore the 2nd phase is almost identical to the 1st phase except that the intervals have shrunk by a factor of 2. Similarly in the 3rd phase we use R_t to offset the posterior of $\theta_\star + R_t$ to Unif$([-1/4, 1/4])$, and so on.

In phase m, the half-length of the interval is 2^{-m+1}, and the probability that a mistake occurs is at least $1/2 - \epsilon/2^{-m+2}$ for any algorithm. The whole process continues as long as this probability is greater than 0. By linearity of expectation, we can lower bound the total mistakes by the sum of expected mistakes in each phase, which gives

$$\sum_{2^{-m+1} \geq \epsilon} d(1/2 - \epsilon/2^{-m+2}) \geq \sum_{2^{-m+1} \geq 2\epsilon} d \cdot 1/4 \geq \lfloor \log_2(1/\epsilon) \rfloor d/4. \tag{6}$$

The above analysis made a simplification that the posterior of $\theta_\star + R_t$ in phase m is $[-2^{-m+1}, 2^{-m+1}]$. We now remove the simplification. Note, however, that if we choose R_t to center the posterior, R_t reveals no additional information about $\theta_\star$, and in the worst case the interval shrinks to half of its previous size minus ϵ. So the length of interval in phase m is at least $2^{-m+2}(1+\epsilon) - 2\epsilon$, and the error probability is at least $1/2 - \epsilon/(2^{-m+1}(1+\epsilon) - \epsilon)$. The rest of the analysis is similar: we count the number of phases until the error probability drops below $1/4$, and in each of these phases we get at least $d/4$ mistakes in expectation. The number of such phases is given by

$$1/2 - \epsilon/(2^{-m+1}(1+\epsilon) - \epsilon) \geq 1/4,$$

which is satisfied when $2^{-m+1} \geq 5\epsilon$, that is, when $m \leq \lfloor \log_2 \frac{2}{5\epsilon} \rfloor$. This completes the proof. □

F Bounding the ℓ_∞ distance between $\theta_\star$ and the ellipsoid center

To prove Theorem 5, we need an upper bound on $\|\theta_\star - c\|_\infty$ for quantifying the error due to H-step truncation and sampling effects, where c is the ellipsoid center. As far as we know there is no standard result on this issue. However, a simple workaround, described below, allows us to assume $\|\theta_\star - c\|_\infty \leq 2$ without loss of generality.

Whenever $\|c\|_\infty > 1$, there exists coordinate j such that $|c_j| > 1$. We can make a central cut $\boldsymbol{e}_j^\top(\theta - c) < 0$ (or > 0 depending on the sign of c_j), and replace the original ellipsoid with the MVEE of the remaining shape. This operation never excludes any point in Θ_0, hence it allows the proofs of Theorem 2 and 5 to work. We keep making such cuts and update the ellipsoid accordingly, until the new center satisfies $\|c\|_\infty \leq 1$. Since central cuts reduce volume substantially (Lemma 1) and there is a lower bound on the volume, the process must stop after finite number of operations. After the process stops, we have $\|\theta_\star - c\|_\infty \leq \|\theta_\star\|_\infty + \|c\|_\infty \leq 2$.

G Proof of Theorem 5

We first introduce a standard concentration inequality for martingales.

Lemma 3 (Azuma's inequality for martingales)**.** *Suppose $\{S_0, S_1, \ldots, S_n\}$ is a martingale and $|S_i - S_{i-1}| \leq b$ almost surely. Then with probability at least $1-\delta$ we have $|S_n - S_0| \leq b\sqrt{2n\log(2/\delta)}$.*

Proof. Since the update rule is still in the format of a central cut through the ellipsoid, Lemma 1 applies. It remains to show that the update rule preserves $\theta_\star$ and a certain volume around it, and then we can follow the same argument as for Theorem 2.

Fixing a mini-batch, let t_0 be the round on which the last update occurs, and $\Theta = \Theta_{t_0}, c = c_{t_0}$. Note that $\Theta_t = \Theta$ during the collection of the current mini-batch and does not change, and $c_t = c$ similarly.

For each $i = 1, 2, \ldots, n$, define $z_i^{\star,H}$ as the expected value of $\hat{z}_i^{\star,H}$, where expectation is with respect to the randomness of the trajectory produced by the human, and let $z_i^\star$ be the infinite-step expected state occupancy. Note that $\hat{z}_i^{\star,H}, z_i^{\star,H}, z_i^\star \in \mathbb{R}^{|\mathcal{S}|}$.

As before, we have $\theta_\star^\top(z_i^\star - z_i) > \epsilon$ and $c^\top(z_i^\star - z_i) \leq 0$, so $(\theta_\star - c)^\top(z_i^\star - z_i) > \epsilon$. Taking average over i, we get $(\theta_\star - c)^\top(\frac{1}{n}\sum_{i=1}^n z_i^\star - \frac{1}{n}\sum_{i=1}^n z_i) > \epsilon$.

What we will show next is that $(\theta_\star - c)^\top(\frac{\bar{Z}^\star}{n} - \frac{\bar{Z}}{n}) > \epsilon/3$ for $\bar{Z}^\star$ and $\bar{Z}$ on Line 11, which implies that the update rule is valid and has enough slackness for lower bounding the volume of Θ_t as before. Note that

$$(\theta_\star - c)^\top(\tfrac{\bar{Z}^\star}{n} - \tfrac{\bar{Z}}{n}) = (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^\star - \tfrac{1}{n}\sum_{i=1}^n z_i)$$
$$- (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^\star - \tfrac{1}{n}\sum_{i=1}^n z_i^{\star,H})$$
$$- (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^{\star,H} - \tfrac{1}{n}\sum_{i=1}^n \hat{z}_i^{\star,H}).$$

Here we decompose the expression of interest into 3 terms. The 1st term is lower bounded by ϵ as shown above, and we will upper bound each of the remaining 2 terms by $\epsilon/3$. For the 2nd term, since $\|z_i^{\star,H} - z_i^\star\|_1 \le \gamma^H$, the ℓ_1 norm of the average follows the same inequality due to convexity, and we can bound the term using Hölder's inequality given $\|\theta_\star - c\|_\infty \le 2$ (see details of this result in Appendix F). To verify that the choice of H in the theorem statement is appropriate, we can upper bound the 2nd term as

$$2\gamma^H = 2((1-(1-\gamma))^{\frac{1}{1-\gamma}})^{\log(6/\epsilon)} \le 2e^{-\log(6/\epsilon)} = \tfrac{\epsilon}{3}.$$

For the 3rd term, fixing $\theta_\star$ and c, the partial sum $\sum_{j=1}^i(\theta_\star - c)^\top(z_i^{\star,H} - \hat{z}_i^{\star,H})$ is a martingale. Since $\|z_i^{\star,H}\|_1 \le 1$, $\|\hat{z}_i^{\star,H}\|_1 \le 1$, and $\|\theta_\star - c\|_\infty \le 2$, we can initiate Lemma 3 by letting $b = 4$, and setting n to sufficiently large to guarantee that the 3rd term is upper bounded by $\epsilon/3$ with high probability.

Given $(\theta_\star - c)^\top(\frac{\bar{Z}^\star}{n} - \frac{\bar{Z}}{n}) > \epsilon/3$, we can follow exactly the same analysis as for Theorem 2 to show that $B_\infty(\theta_\star, \epsilon/6)$ is never eliminated, and the number of updates can be bounded by $2d(d+1)\log\frac{12\sqrt{d}}{\epsilon}$. The number of total mistakes is the number of updates multiplied by n, the size of the mini-batches. Via Lemma 3, we can verify that the choice of n in the theorem statement satisfies $|\sum_{j=1}^i(\theta_\star - c)^\top(z_i^{\star,H} - \hat{z}_i^{\star,H})| \le n\epsilon/3$ with probability at least $1-\delta/\left(2d(d+1)\log\frac{12\sqrt{d}}{\epsilon}\right)$. Union bounding over all updates and the total failure probability can be bounded by δ. □