Inverse Reinforcement Learning From Like-Minded Teachers

Ritesh Noothigattu,[1*] **Tom Yan,**[1*] **Ariel D. Procaccia**[2]
[1] Carnegie Mellon University [2] Harvard University [*] Equal contribution

Abstract

We study the problem of learning a policy in a Markov decision process (MDP) based on observations of the actions taken by multiple teachers. We assume that the teachers are like-minded in that their reward functions — while different from each other — are random perturbations of an underlying reward function. Under this assumption, we demonstrate that inverse reinforcement learning algorithms that satisfy a certain property — that of *matching feature expectations* — yield policies that are approximately optimal with respect to the underlying reward function, and that no algorithm can do better in the worst case. We also show how to efficiently recover the optimal policy when the MDP has one state — a setting that is akin to multi-armed bandits.

1 Introduction

A *Markov decision process (MDP)* is a formal specification of a sequential decision making environment, which consists of a set of states, a set of actions, a reward function, and a stochastic transition function. *Reinforcement learning (RL)* deals with learning a *policy* in an MDP — which specifies a possibly randomized action that is taken in each state — to maximize cumulative reward.

RL has long history in AI (Sutton and Barto 1998; Kaelbling, Littman, and Moore 1996), as well as in many other disciplines. But in recent years, interest in the area has exploded, in part due to breakthroughs in game playing (Mnih et al. 2015; Silver et al. 2016) and fast-growing applications to robotics (Kober, Bagnell, and Peters 2013). It is safe to say that, nowadays, RL is widely considered to be one of the basic building blocks in the construction of intelligent agents.

While most work in the area focuses on maximizing a given reward function, some settings require the AI system to emulate the behavior of an expert or teacher (Ng and Russell 2000; Abbeel and Ng 2004) — this is known as *inverse reinforcement learning (IRL)*. The idea is to observe an agent executing a policy in an MDP, where everything is known to the learner except the reward function, and extract a reward function that is most likely to be the one being optimized by the agent. Using this reward function — and knowledge of the other components of the MDP — the agent can easily compute an optimal policy to follow.

Our point of departure is that we are interested in IRL from multiple agents rather than a single agent. Specifically, we observe n different agents executing policies that are optimal for their individual reward functions. Our approach is to aggregate these observations into a single policy, by applying an inverse reinforcement learning algorithm to the set of all observations.

However, if individual agents have wildly divergent reward functions then the aggregate policy may not represent coherent behavior. In addition, to formally reason about the quality of the optimal policy, we need to relate it to some notion of ground truth. For these reasons, we assume that the agents are *like-minded*, in that individual reward functions are nothing but noisy versions of an underlying reward function.

In summary, our research challenge is this:

Given observations from policies that are optimal with respect to different reward functions, each of which is a perturbation of an underlying reward function, identify IRL algorithms that can recover a good policy with respect to the underlying reward function.

We believe that this problem is both natural and general. To further motivate it, though, let us briefly instantiate it in the context of beneficial AI. One of the prominent approaches in this area is to align the values of the AI system with the values of a human through IRL (Russell, Dewey, and Tegmark 2015; Hadfield-Menell et al. 2016). Our extension to multiple agents would allow the alignment of the system with the values of *society*.

A compelling aspect of this instantiation is that, if we think of the underlying reward function as embodying a common set of moral propositions, then our technical assumption of like-minded agents can be justified through the *linguistic analogy*, originally introduced by Rawls (1971). It draws on the work of Chomsky (1965), who argued that competent speakers have a set of grammatical principles in mind, but their linguistic behavior is hampered by "grammatically irrelevant conditions such as memory limitations, distractions, shifts of attention and interest, and errors." Analogously, Rawls claimed, humans have moral rules — a common "moral grammar" — in our minds, but, due to various limitations, our moral behavior is only an approximation thereof. Interestingly, this theory lends itself to empirical experimentation, and, indeed, it has been validated

through work in moral psychology (Mikhail 2011).

Our Model and Results. We start from a common IRL setup: each reward function is associated with a weight vector $\mathbf{w}$, such that the reward for taking a given action in a given state is the dot product of the weight vector and the feature vector of that state-action pair. The twist is that there is an underlying reward function represented by a weight vector $\mathbf{w}^\star$, and each of the agents is associated with a weight vector $\mathbf{w}_i$, which induces an optimal policy π_i. We observe a trajectory from each π_i.

In Section 3, we focus on competing with a uniform mixture over the optimal policies of the agents, $\pi_1, \ldots, \pi_n$ (for reasons that we explicate momentarily). We can do this because the observed trajectories are "similar" to the uniform mixture, in the sense that their feature vectors — the discounted frequencies of the features associated with the observed state-action pairs — are close to that of the uniform mixture policy. Therefore, due to the linearity of the reward function, any policy whose feature expectations approximately match those of the observed trajectories must be close to the uniform mixture with respect to $\mathbf{w}^\star$. We formalize this idea in Theorem 3.2, which gives a lower bound on the number of agents and length of observed trajectories such that any policy that $\epsilon/3$-matches feature expectations is ϵ-close to the uniform mixture. Furthermore, we identify two well-known IRL algorithms, Apprenticeship Learning (Abbeel and Ng 2004) and Max Entropy (Ziebart et al. 2008), which indeed output policies that match the feature expectations of the observed trajectories, and therefore enjoy the guarantees provided by this theorem.

Needless to say, competing with the uniform mixture is only useful insofar as this benchmark exhibits "good" performance. We show that this is indeed the case in Section 4, assuming (as stated earlier) that each weight vector $\mathbf{w}_i$ is a noisy perturbation of $\mathbf{w}^\star$. Specifically, we first establish that, under relatively weak assumptions on the noise, it is possible to bound the difference between the reward of the uniform mixture and that of the optimal policy (Theorem 4.1). More surprisingly, Theorem 4.3 asserts that in the worst case it is impossible to outperform the uniform mixture, by constructing an MDP where the optimal policy cannot be identified — even if we had an infinite number of agents and infinitely long trajectories! Putting all of these results together, we conclude that directly running an IRL algorithm that matches feature expectations on the observed trajectories is a sensible approach to our problem.

Nevertheless, it is natural to ask whether it is possible to outperform the uniform mixture in typical instances. In Section 5 we show that this is indeed the case; in fact, we are able to recover the optimal policy whenever it is identifiable, albeit under stringent assumptions — most importantly, that the MDP has only one state. This leads to a challenge that we call the *inverse multi-armed bandit problem*. To the best of our knowledge, this problem is novel; its study contributes to the (relatively limited) understanding of scenarios where it is possible to outperform teacher demonstrations.

Related work. The most closely related work deals with IRL when the observations come from an agent who acts according to multiple *intentions*, each associated with a different reward function (Babeş-Vroman et al. 2011; Choi and Kim 2012). The main challenge stems from the need to cluster the observations — the observations in each cluster are treated as originating from the same policy (or intention). By contrast, clustering is a nonissue in our framework. Moreover, our assumption that each $\mathbf{w}_i$ is a noisy perturbation of $\mathbf{w}^\star$ allows us to provide theoretical guarantees.

Further afield, there is a body of work on robust RL and IRL under reward uncertainty (Givan, Leach, and Dean 2000; Regan and Boutilier 2009, 2010), noisy rewards (Zheng, Liu, and Ni 2014), and corrupted rewards (Everitt et al. 2017). Of these papers the closest to ours is that of Zheng, Liu, and Ni (2014), who design robust IRL algorithms under *sparse* noise, in the sense that only a small fraction of the observations are anomalous; they do not provide theoretical guarantees. Our setting is quite different, as very few observations would typically be associated with a near-perfect policy.

2 MDP Terminology

We assume the environment is modeled as an MDP $\{S, A, T, \gamma, D\}$ with an unknown reward function. S is a finite set of states; A is a finite set of actions; $T(s, a, s')$ is the state transition probability of reaching state s' from state s when action a is taken; $\gamma \in [0, 1)$ is the discount factor; and D the initial-state distribution, from which the start state s_0 is drawn for every trajectory.

As is standard in the literature (Abbeel and Ng 2004), we assume that there is a function $\phi : S \times A \to \mathbb{R}^d$ that maps state-action pairs to their real-valued features. We also overload notation, and say that the feature vector of a trajectory $\tau = \{(s_0, a_0), (s_1, a_1), \ldots, (s_L, a_L)\}$ is defined as $\phi(\tau) = \sum_{t=0}^{L} \gamma^t \phi(s_t, a_t)$.

We make the standard assumption that the immediate reward of executing action a from state s is linear in the features of the state-action pair, i.e. $r^{\mathbf{w}}(s, a) = \mathbf{w}^\intercal \phi(s, a)$. This has a natural interpretation: ϕ represents the different factors, and $\mathbf{w}$ weighs them in varying degrees.

Let μ denote the feature expectation of policy π, that is, $\mu(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t) | \pi]$, where π defines the action a_t taken from state s_t, and the expectation is taken over the transition probabilities $T(s_t, a_t, s_{t+1})$. Hence, the cumulative reward of a policy π under weight $\mathbf{w}$ can be rewritten as:

$$\begin{aligned} R^{\mathbf{w}}(\pi) &= \mathbb{E}_{s_0 \sim D}[V^\pi(s_0)] \\ &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r^{\mathbf{w}}(s_t, a_t) \middle| \pi\right] \\ &= \mathbf{w}^\intercal \cdot \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a) \middle| \pi\right] \\ &= \mathbf{w}^\intercal \mu(\pi). \end{aligned}$$

Let $P_\pi(s, t)$ denote the probability of getting to state s at time t under policy π. Then, the cumulative reward $R^{\mathbf{w}}$ is

$$R^{\mathbf{w}}(\pi) = \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} P_\pi(s, t) r^{\mathbf{w}}(s, \pi(s)).$$

3 Approximating the Uniform Mixture

We consider an environment with n agents $N = \{1, \ldots, n\}$. Furthermore, the reward function of each agent $i \in N$ is associated with a weight vector $\mathbf{w}_i$, and, therefore, with a reward function $r^{\mathbf{w}_i}$. This determines the optimal policy π_i executed by agent i, from which we observe the trajectory τ_i, which consists of L steps. We observe such a trajectory for each $i \in N$, giving us trajectories $\{\tau_1, ..., \tau_n\}$.

As we discussed in Section 1, we assume that the reward function associated with each agent is a noisy version of an underlying reward function. Specifically, we assume that there exists a ground truth weight vector $\mathbf{w}^\star$, and for each agent $i \in N$ we let $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, where $\boldsymbol{\eta}_i$ is the corresponding noise vector; we assume throughout that $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n$ are i.i.d. Following Abbeel and Ng (2004), we also assume in some of our results (when stated explicitly) that $\|\mathbf{w}^\star\|_2 \leq 1$ and $\|\phi(s,a)\|_\infty \leq 1$.

Let us denote by π^u the *uniform mixture* over the policies $\pi_1, \ldots, \pi_n$, that is, the (randomized) policy that, in each trajectory, selects one of these policies uniformly at random and executes it throughout the trajectory.

Our goal in this section is to "approximate" the uniform mixture (and we will justify this choice in subsequent sections). To do so, we focus on IRL algorithms that "match feature expectations." Informally, the property of interest is that the feature expectations of the policy match the (discounted) feature vectors of observed trajectories. This idea is already present in the IRL literature, but it is helpful to define it formally, as it allows us to identify specific IRL algorithms that work well in our setting.

Definition 3.1. Given n trajectories $\tau_1, ..., \tau_n$, a (possibly randomized) policy π *ϵ-matches their feature expectations* if and only if $\|\mu(\pi) - \frac{1}{n}\sum_{i=1}^n \phi(\tau_i)\|_2 \leq \epsilon$.

In a nutshell, due to the linearity of the reward function, two policies that have the same feature expectations have the same reward. Therefore, if the observed trajectories closely mimic the feature expectations of π_u, and a policy $\tilde{\pi}$ matches the feature expectations of the observed trajectories, then the reward of $\tilde{\pi}$ would be almost identical to that of π^u. This is formalized in the following theorem, whose proof is relegated to Appendix B.

Theorem 3.2. *Assume that $\|\phi(s,a)\|_\infty \leq 1$ for all $s \in S, a \in A$. Let $\mathbf{w}^\star$ such that $\|\mathbf{w}^\star\|_2 \leq 1$, fix any $\mathbf{w}_1, \ldots, \mathbf{w}_n$, and, for all $i \in N$, let τ_i be a trajectory of length L sampled by executing π_i. Let $\tilde{\pi}$ be a policy that $\epsilon/3-$matches the feature expectation of these trajectories. If*

$$n \geq \frac{72 \ln\left(\frac{2}{\delta}\right) d}{\epsilon^2 (1-\gamma)^2} \quad \text{and} \quad L \geq \log_{1/\gamma} \frac{3\sqrt{d}}{(1-\gamma)\epsilon}$$

then, with probability at least $1-\delta$, it holds that $\left|R^{\mathbf{w}^\star}(\tilde{\pi}) - R^{\mathbf{w}^\star}(\pi^u)\right| \leq \epsilon$.

Note that the required number of agents n may be significant; fortunately, we can expect access to data from many agents in applications of interest. For example, Noothigattu et al. (2018) built a system that decides ethical dilemmas based on data collected from 1.3 million people.

To apply Theorem 3.2, we need to use IRL algorithms that match feature expectations. We have identified two algorithms that satisfy this property: the *Apprenticeship Learning* algorithm of Abbeel and Ng (2004), and the *Max Entropy* algorithm of Ziebart et al. (2008). For completeness we present these algorithms, and formally state their feature-matching guarantees, in Appendix A.

4 How Good is the Uniform Mixture?

In Section 3 we showed that it is possible to (essentially) match the performance of the uniform mixture with respect to the ground truth reward function. In this section we justify the idea of competing with the uniform mixture in two ways: first, we show that the uniform mixture approximates the optimal policy under certain assumptions on the noise, and, second, we prove that in the worst case it is actually impossible to outperform the uniform mixture.

4.1 The Uniform Mixture Approximates the Optimal Policy

Recall that for all $i \in n$, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$. It is clear that without imposing some structure on the noise vectors $\boldsymbol{\eta}_i$, no algorithm would be able to recover a policy that does well with respect to $\mathbf{w}^\star$.

Let us assume, then, that the noise vectors $\boldsymbol{\eta}_i$ are such that the η_{ik} are independent and each η_{ik}^2 is sub-exponential. Formally, a random variable X with mean $u = \mathbb{E}[X]$ is *sub-exponential* if there are non-negative parameters (ν, b) such that $\mathbb{E}\left[\exp\left(\lambda(X-u)\right)\right] \leq \exp\left(\nu^2\lambda^2/2\right)$ for all $|\lambda| < 1/b$. This flexible definition simply means that the moment generating function of the random variable X is bounded by that of a Gaussian in a neighborhood of 0. Note that if a random variable is sub-Gaussian, then its square is sub-exponential. Hence, our assumption is strictly weaker than assuming that each η_{ik} is sub-Gaussian.

Despite our assumption about the noise, it is *a priori* unclear that the uniform mixture would do well. The challenge is that the noise operates on the coordinates of the individual weight vectors, which in turn determine individual rewards, but, at first glance, it seems plausible that relatively small perturbations of rewards would lead to severely suboptimal policies. Our result shows that this is not the case: π^u is approximately optimal with respect to $R^{\mathbf{w}^\star}$, in expectation.

Theorem 4.1. *Assume that $\|\phi(s,a)\|_\infty \leq 1$ for all $s \in S, a \in A$. Let $\mathbf{w}^\star$ such that $\|\mathbf{w}^\star\|_2 \leq 1$, and suppose that $\mathbf{w}_1, ..., \mathbf{w}_n$ are drawn from i.i.d. noise around $\mathbf{w}^\star$, i.e., $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, where each of its coordinates is such that η_{ik}^2 is an independent sub-exponential random variable with parameters (ν, b). Then*

$$\mathbb{E}[R^{\mathbf{w}^\star}(\pi^u)] \geq R^{\mathbf{w}^\star}(\pi^\star) - O\left(d\sqrt{u} + \nu\sqrt{\frac{d}{u}} + \frac{b}{\sqrt{u}}\right),$$

where $u = \frac{1}{d}\sum_{k=1}^d \mathbb{E}\left[\eta_{ik}^2\right]$, and the expectation is taken over the noise.

The exact expression defining the gap between $\mathbb{E}[R^{\mathbf{w}^\star}(\pi^u)]$ and $R^{\mathbf{w}^\star}(\pi^\star)$ can be found in the proof

of Theorem 4.1, which appears in Appendix C; we give the asymptotic expression in the theorem's statement because it is easier to interpret. As one might expect, this gap increases as ν or b is increased (and, in a linear fashion). This is intuitive because a smaller ν or b imposes a strictly stronger assumption on the sub-exponential random variable (and its tails).

To gain more insight, we analyze the upper bound on the gap when η_{ik} follows a Gaussian distribution, that is, $\eta_{ik} \sim \mathcal{N}(0, \sigma^2)$. Note that this implies that η_{ik}^2 follows a χ_1^2 distribution scaled by σ^2; a χ_1^2 distributed random variable is known to be sub-exponential with parameters $(2, 4)$, and hence this implies that η_{ik}^2 is sub-exponential with parameters $(2\sigma^2, 4\sigma^2)$. Further, in this case, $u = \mathbb{E}\left[\eta_{ik}^2\right] = \sigma^2$. Plugging these quantities into the upper bound of Theorem 4.1 shows that the gap is bounded by $O(d\sigma)$.

Theorem 4.1 shows that the gap depends linearly on the number of features d. An example given in Appendix D shows that this upper bound is tight. Nevertheless, the tightness holds in the worst case, and one would expect the practical performance of the uniform mixture to be very good. To corroborate this intuition, we provide (unsurprising) experimental results in Appendix E.

4.2 It is Impossible to Outperform the Uniform Mixture in the Worst Case

An ostensible weakness of Theorem 4.1 is that even as the number of agents n goes to infinity, the reward of the uniform mixture may not approach that of the optimal policy, that is, there is a persistent gap. The example given in Section 4.1 shows the gap is not just an artifact of our analysis. This is expected, because the data contains some agents with suboptimal policies π_i, and a uniform mixture over these suboptimal policies must itself be suboptimal.

It is natural to ask, therefore, whether it is generally possible to achieve performance arbitrarily close to $\pi^\star$ (at least in the limit that n goes to infinity). The answer is negative. In fact, we show that — in the spirit of *minimax optimality* (Hodges Jr and Lehmann 1950; Perron and Marchand 2002) — one cannot hope to perform better than π^u itself in the worst case. Intuitively, there exist scenarios where it is impossible to tell good and bad policies apart by looking at the data, which means that the algorithm's performance depends on what can be gleaned from the "average data".

This follows from a surprising[1] result that we think of as "non-identifiability" of the optimal policy. To describe this property, we introduce some more notation. The distribution over the weight vector of each agent i, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$, in turn induces a distribution over the optimal policy π_i executed by each agent. Denote this distribution by $\mathcal{P}(\mathbf{w}^\star)$.[2] Hence, each agent's optimal policy π_i is just a sample from this distribution $\mathcal{P}(\mathbf{w}^\star)$. In particular, as the number of agents goes to infinity, the empirical distribution of their optimal policies would exactly converge to $\mathcal{P}(\mathbf{w}^\star)$.

[1] At least it was surprising for us — we spent significant effort trying to prove the opposite result!

[2] Note that this distribution does not depend on i itself since the noise $\boldsymbol{\eta}_i$ is i.i.d. across the different agents.

For the rest of this section, we make minimal assumptions on the noise vector $\boldsymbol{\eta}_i$. In particular, we merely assume that $\boldsymbol{\eta}_i$ follows a continuous distribution and that each of its coordinates is i.i.d. We are now ready to state our non-identifiability lemma.

Lemma 4.2 (non-identifiability)**.** *For every continuous distribution $\mathcal{D}$ over $\mathbb{R}$, if η_{ik} is independently sampled from $\mathcal{D}$ for all $i \in N$ and $k \in [d]$, then there exists an MDP and weight vectors $\mathbf{w}_a^\star$, $\mathbf{w}_b^\star$ with optimal policies $\pi_a^\star$, $\pi_b^\star$, respectively, such that $\pi_a^\star \neq \pi_b^\star$ but $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star)$.*

Even if we had an infinite number of trajectories in our data, and even if we knew the exact optimal policy played by each player i, this information would amount to knowing $\mathcal{P}(\mathbf{w}^\star)$. Hence, if there exist two weight vectors $\mathbf{w}_a^\star$, $\mathbf{w}_b^\star$ with optimal policies $\pi_a^\star$, $\pi_b^\star$ such that $\pi_a^\star \neq \pi_b^\star$ and $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star)$, then we would not be able to identify whether the optimal policy is $\pi_a^\star$ or $\pi_b^\star$ regardless of how much data we had.

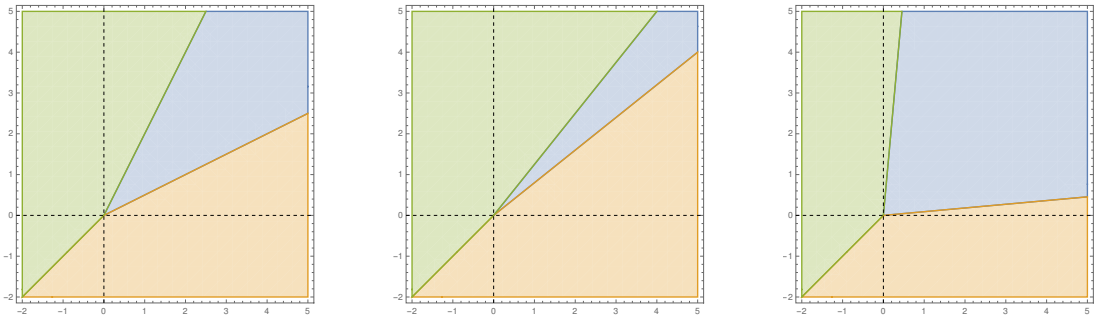
The proof of Lemma 4.2 is relegated to Appendix F. Here we provide a proof sketch.

Proof sketch of Lemma 4.2. The intuition for the lemma comes from the construction of an MDP with three possible policies, all of which have probability $1/3$ under $\mathcal{P}(\mathbf{w}^\star)$, even though one is better than the others. This MDP has a single state s, and three actions $\{a, b, c\}$ that lead back to s. Denote the corresponding policies by π_a, π_b, π_c. Let the feature expectations be $\phi(s, a) = [0.5, 0.5], \phi(s, b) = [1, -\delta/2], \phi(s, c) = [-\delta/2, 1]$, where $\delta > 0$ is a parameter. Let the ground truth weight vector be $\mathbf{w}^\star = (v_o, v_o)$, where v_o is such that the noised weight vector $\mathbf{w} = \mathbf{w}^\star + \boldsymbol{\eta}$ has probability strictly more than $1/3$ of lying in the first quadrant; such a value always exists for any noise distribution that is continuous and i.i.d. across coordinates.

Let us look at weight vectors $\mathbf{w}$ for which each of the three policies π_a, π_b and π_c are optimal. π_a is the optimal policy when $\mathbf{w}^\intercal \mu_a > \mathbf{w}^\intercal \mu_b$ and $\mathbf{w}^\intercal \mu_a > \mathbf{w}^\intercal \mu_c$, which is the intersection of the half-spaces $\mathbf{w}^\intercal(-1, 1 + \delta) > 0$ and $\mathbf{w}^\intercal(1 + \delta, -1) > 0$. Similarly, we can reason about the regions where π_b and π_c are optimal. These regions are illustrated in Figure 1 for different values of δ. Informally, as δ is decreased, the lines separating (π_a, π_c) and (π_a, π_b) move closer to each other (as shown for $\delta = 0.25$), while as δ is increased, these lines move away from each other (as shown for $\delta = 10$). By continuity and symmetry, there exists δ such that the probability of each of the regions (with respect to the random noise) is exactly $1/3$, showing that the MDP has the desired property.

To complete the proof of the lemma, we extend the MDP by adding two more features to the existing two. By setting these new features appropriately (in particular, by cycling the two original features across the arms), we can show that the two weight vectors $\mathbf{w}_a^\star = (v_o, v_o, 0, 0)$ and $\mathbf{w}_b^\star = (0, 0, v_o, v_o)$ lead to $\mathcal{P}(\mathbf{w}_a^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}) = \mathcal{P}(\mathbf{w}_b^\star)$, even though their corresponding optimal policies are π_a and π_b, respectively. □

For the next theorem, therefore, we can afford to be "generous:" we will give the algorithm (which is trying to com-



$\delta = 1$ $\delta = 0.25$ $\delta = 10$

Figure 1: Regions of each optimal policy for different values of δ. Blue depicts the region where π_a is optimal, orange is where π_b is optimal, and green is where π_c is optimal.

pete with π^u) access to $\mathcal{P}(\mathbf{w}^\star)$, instead of restricting it to sampled trajectories. Formally, the theorem holds for any algorithm that takes a distribution over policies as input, and returns a randomized policy.

Theorem 4.3. *For every continuous distribution $\mathcal{D}$ over $\mathbb{R}$, if η_{ik} is independently sampled from $\mathcal{D}$ for all $i \in N$ and $k \in [d]$, then there exists an MDP such that for any algorithm $\mathcal{A}$ from distributions over policies to randomized policies, there exists a ground truth weight vector $\mathbf{w}^\star$ such that $R^{\mathbf{w}^\star}(\mathcal{A}(\mathcal{P}(\mathbf{w}^\star))) \leq R^{\mathbf{w}^\star}(\pi^u) < R^{\mathbf{w}^\star}(\pi^\star)$.*

In words, the constructed instance is such that, even given infinite data, no algorithm can outperform the uniform mixture, and, moreover, the reward of the uniform mixture is bounded away from the optimum. The theorem's proof is given in Appendix G.

5 The Inverse Multi-Armed Bandit Problem

In Section 4, we have seen that it is impossible to outperform the uniform mixture in the worst case, as the optimal policy is not identifiable. However, it is natural to ask when the optimal policy is identifiable and how it may be practically recovered. In this section we give an encouraging answer, albeit in a restricted setting.

Specifically, we focus on the multi-armed bandit problem, which is an MDP with a single state. Note that the non-identifiability result of Lemma 4.2 still holds in this setting, as the example used in its proof is an MDP with a single state. Hence, even in this setting of bandits, it is impossible to outperform the uniform mixture in the worst case. However, we design an algorithm that can guarantee optimal performance when the problem is identifiable, under some additional conditions.

Like the general setting considered earlier, there exists a ground truth weight vector $\mathbf{w}^\star$, and for each agent $i \in N$, $\mathbf{w}_i = \mathbf{w}^\star + \boldsymbol{\eta}_i$. For this section, we assume the noise vector $\boldsymbol{\eta}_i$ to be Gaussian and i.i.d. across agents and coordinates. In particular, $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2 I_d)$, and independent across i.

The bandit setting is equivalent to a single-state MDP, and hence the components S, T, γ and D are moot. Instead, there are m arms to pull, denoted by $A = \{1, 2, \ldots, m\}$. Similar to our original feature function ϕ, we now have features $\mathbf{x}_j \in \mathbb{R}^d$ associated with arm j, for each $j \in A$. Although in standard stochastic bandit problems we have a reward sampled from a distribution when we pull an arm, we care only about its mean reward in this section. For weight vector $\mathbf{w}$, the (mean) reward of pulling arm j is given by $r^{\mathbf{w}}(j) = \mathbf{w}^\intercal \mathbf{x}_j$. For each agent i (with weight vector $\mathbf{w}_i$), we assume that we observe the optimal arm being played by this agent, i.e., $\tilde{a}_i = \text{argmax}_{j \in A} \mathbf{w}_i^\intercal \mathbf{x}_j$.

We observe the dataset $\mathcal{D} = \{\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n\}$ which is the set of optimal arms played by the agents. Define $\mathcal{Q}(\mathbf{w}^\star)$ to be the distribution over optimal arms induced when the ground truth weight vector is $\mathbf{w}^\star$. In particular, ground truth weight vector $\mathbf{w}^\star$ induces a distribution over the noised weight vector of each agent (via $\mathbf{w} = \mathbf{w}^\star + \eta$), which in turn induces a discrete distribution over the optimal arm that would be played, which we call $\mathcal{Q}(\mathbf{w}^\star)$ — analogously to the $\mathcal{P}(\mathbf{w}^\star)$ of Section 4. Observe that the dataset $\mathcal{D}$ could be rewritten as a distribution over arms, $\tilde{\mathcal{Q}} = (\tilde{\mathcal{Q}}_1, \tilde{\mathcal{Q}}_2, \ldots, \tilde{\mathcal{Q}}_m)$, which is the observed distribution of optimal arms. Moreover, as each agent's optimal arm played is an i.i.d. sample from $\mathcal{Q}(\mathbf{w}^\star)$, the empirical distribution $\tilde{\mathcal{Q}}$ is an unbiased estimate of $\mathcal{Q}(\mathbf{w}^\star)$.

The *inverse multi-armed bandit problem* is to recover $\mathbf{w}^\star$ given the distribution $\tilde{\mathcal{Q}}$, which allows us to identify the optimal arm. In order to achieve this, we aim to find $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, or matches it as closely as possible. Ideally, we would want to find $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \mathcal{Q}(\mathbf{w}^\star)$,[3] but since we do not have access to $\mathcal{Q}(\mathbf{w}^\star)$, we use the unbiased estimate $\tilde{\mathcal{Q}}$ in its place.[4] Below, we produce conditions under which the optimal policy is recoverable, and provide a practical algorithm that achieves this for all settings that meet the criteria.

[3] Note that there might be multiple $\mathbf{w}$ such that $\mathcal{Q}(\mathbf{w}) = \mathcal{Q}(\mathbf{w}^\star)$. However, since we care only about the corresponding optimal arm, and identifiability tells us that all weight vectors with the same $\mathcal{Q}$ value have the same optimal arm, we just need to find one such weight vector.

[4] In most cases we will have collected sufficient data such that the optimal arm corresponding to $\tilde{\mathcal{Q}}$ coincides with the optimal arm corresponding to $\mathcal{Q}(\mathbf{w}^\star)$. Although they may not coincide, this probability goes to zero as the size of the dataset $\mathcal{D}$ increases.

5.1 Identifying the Optimal Arm

Since the constraint $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ is "far" from being convex in $\mathbf{w}$, we reformulate the problem such that the new problem is convex, and all its optimal solutions satisfy the required constraint (and vice versa). The new objective we use is the cross entropy loss between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w})$. That is, the optimization problem to solve is

$$\min_{\mathbf{w}} -\sum_{k \in A} \tilde{\mathcal{Q}}_k \log \mathcal{Q}(\mathbf{w})_k. \tag{1}$$

It is obvious that this objective is optimized at points with $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, if the original problem was feasible. Otherwise, it finds $\mathbf{w}$ whose $\mathcal{Q}$ is as close to $\tilde{\mathcal{Q}}$ as possible in terms of cross-entropy. Furthermore, this optimization problem is convex under a simple condition, which requires the definition of X_k as an $(m-1) \times d$ matrix with rows of the form $(\mathbf{x}_k - \mathbf{x}_j)^\intercal$, for each $j \in A \setminus \{k\}$.

Theorem 5.1. *Optimization problem* (1) *is convex if $X_k X_k^\intercal$ is invertible for each $k \in A$.*

The proof of the theorem appears in Appendix H. An exact characterization of when $X_k X_k^\intercal$ is full rank is $\text{rank}(X_k X_k^\intercal) = \text{rank}(X_k) = m - 1$, i.e. when X_k is full row rank. For this to be true, a necessary condition is that $d \geq m - 1$ as $\text{rank}(X_k) \leq \min(d, m-1)$. And under this condition, the requirement for X_k to to be full row rank is that the rows $(\mathbf{x}_k - \mathbf{x}_j)^\intercal$ are linearly independent, which is very likely to be the case, unless the feature vectors were set up adversarially. One potential scenario where the condition $d \geq m - 1$ would arise is when there are many features but feature vectors $\mathbf{x}_j$ are sparse.

As the optimization problem (1) is convex, we can use gradient descent to find a minimizer. And for this, we need to be able to compute the gradient accurately, which we show is possible; the calculation is given in Appendix I.

Importantly, we can also use our procedure to determine whether the optimal arm is identifiable. Given $\tilde{\mathcal{Q}}$, we solve the optimization problem (1) to first find a $\mathbf{w}_o$ such that $\mathcal{Q}(\mathbf{w}_o) = \tilde{\mathcal{Q}}$. Let $\mathbf{w}_o$ have the optimal arm $a_o \in A$. Now, our goal is to check if there exists any other weight $\mathbf{w}$ that has $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ but whose corresponding optimal arm is not a_o. To do this, we can build a set of convex programs, each with the exact same criterion (taking care of the $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$ requirement), but with the constraint that arm $a_i \neq a_o$ is the optimal arm (or at least beats a_o) with respect to $\mathbf{w}$. In particular, the constraint for program i could be $\mathbf{w}^\intercal \mathbf{x}_i > \mathbf{w}^\intercal \mathbf{x}_{a_o}$.[5] As this is a simple affine constraint, solving the convex program is very similar to running gradient descent as before. If any of these convex programs outputs an optimal solution that satisfies $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, then the problem is not identifiable, as it implies that there exist weight vectors with different optimal arms leading to the same $\tilde{\mathcal{Q}}$. On the other hand, if none of them satisfies $\mathcal{Q}(\mathbf{w}) = \tilde{\mathcal{Q}}$, we can conclude that a_o is the desired unique optimal arm.

[5]The strong inequality can be implemented in the standard way via $\mathbf{w}^\intercal \mathbf{x}_i \geq \mathbf{w}^\intercal \mathbf{x}_{a_o} + \epsilon$ for a sufficiently small $\epsilon > 0$ that depends on the program's bit precision.

5.2 Experiments

We next study the empirical performance of our algorithm for the inverse multi-armed bandit problem. We focus on instances inspired by the counter-example from Lemma 4.2. The reason for this is that in randomly generated bandit problems, the optimal arm $a^\star$ is very likely to be the mode of $\mathcal{Q}(\mathbf{w}^\star)$, making the mode of $\tilde{\mathcal{Q}}$ a very good estimator of $a^\star$.[6] By contrast, the counterexample allows us to generate "hard" instances.

Specifically, the bandit instances we consider have two features ($d = 2$) and three arms $A = \{1, 2, 3\}$, and their features are defined as $\mathbf{x}_1 = [1, 1]$, $\mathbf{x}_2 = [2, -\delta]$ and $\mathbf{x}_3 = [-\delta, 2]$, where $\delta > 0$ is a positive constant. The ground truth weight vector is given as $\mathbf{w}^\star = [1, 1]$. Hence, for any $\delta > 0$, the optimal arm is arm 1. The noise is $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$. Such an instance is very similar to the one of Lemma 4.2, except that the features are not replicated to extend from two to four features, and hence the problem remains identifiable.

Observe that when the value of δ is small enough, the blue region of Figure 1 becomes a sliver, capturing a very small density of the noise $\boldsymbol{\eta}$, and causing arm 1 to not be the mode of $\mathcal{Q}(\mathbf{w}^\star)$. Alternatively, for a given value of δ, if σ is large enough, most of the noise's density escapes the blue region, again causing arm 1 to not be the mode of $\mathcal{Q}(\mathbf{w}^\star)$. In the following experiments, we vary both δ and σ, and show that even when the optimal arm almost never appears in $\mathcal{Q}(\mathbf{w}^\star)$, our algorithm is able to recover it.

Varying parameter δ. In the first set of experiments, we fix the noise standard deviation σ to 1, generate $n = 500$ agents according to the noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$, and vary parameter δ from 0.01 to 3. Figure 2 shows the percentage of times our algorithm and the mode recover the optimal arm 1. This graph is averaged over 1000 runs, and error bars depict 95% confidence intervals.

When δ is extremely close to 0, the optimal arm's region almost vanishes. Hence, small differences between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w}^\star)$ could have a substantial effect, and unless $\mathbf{w}^\star$ is numerically recovered within this sliver, the optimal arm would not be recovered. As we move to even slightly larger values of δ, however, the performance of the algorithm improves substantially and it ends up recovering the optimal arm 100% of the time.

By contrast, as δ is varied from 0 to ∞, the density of the noise $\boldsymbol{\eta}$ captured by the blue region increases continuously from 0 to that of the first quadrant. In particular, there is a point where $\mathcal{Q}(\mathbf{w}^\star)$ has probability tied across the three arms, after which arm 1 is always the mode (i.e. mode has 100% performance), and before which arms 2 and 3 are the modes (i.e the mode has 0% performance). This tipping point is evident from the graph and occurs around $\delta = 1$.[7] Observe that the performance of the algorithm rises to 100%

[6]This is because, for each arm a, the region $\mathcal{R}_a = \{\mathbf{w} : \mathbf{w}^\intercal \mathbf{x}_a \geq \mathbf{w}^\intercal \mathbf{x}_j \text{ for each } j\}$, corresponding to where arm a is optimal, forms a polytope, and the optimal arm's region $\mathcal{R}_{a^\star}$ contains $\mathbf{w}^\star$. Hence, as long as $\mathcal{R}_{a^\star}$ has enough volume around $\mathbf{w}^\star$, it would capture a majority of the density of the noise $\boldsymbol{\eta}$, and $a^\star$ would be the mode of the distribution $\mathcal{Q}(\mathbf{w}^\star)$.

[7]The transition in this graph is smoother than a step function

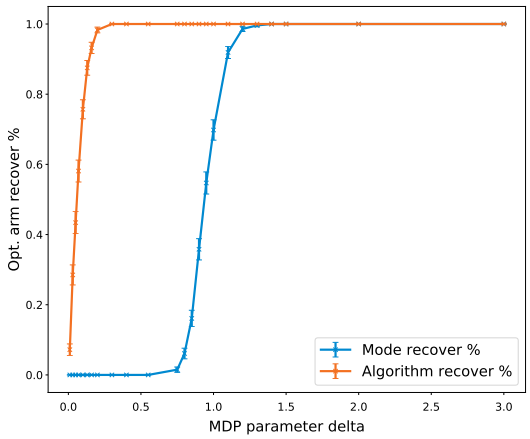


Figure 2: Performance as δ is varied.

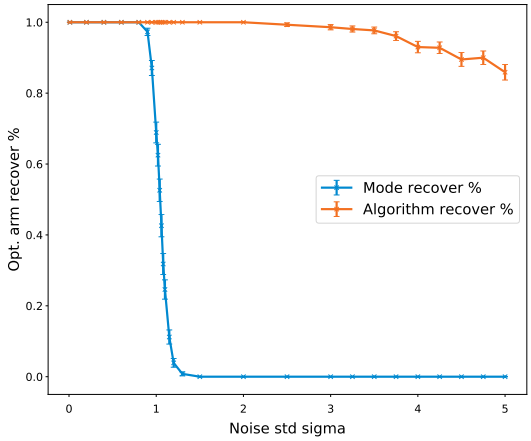


Figure 3: Performance as σ is varied.

much before this tipping point, serving as evidence that it can perform well even if the optimal arm barely appears in the dataset. Appendix J.1 includes similar results when the parameters are set to $\sigma \in \{0.5, 2.0\}$ or $n \in \{250, 1000\}$.

Varying noise parameter σ. Next, we fix the parameter δ to 1 and generate $n = 500$ agents according to noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)$, while varying the noise parameter σ from 0.01 to 5. Figure 3 shows the percentage of times our algorithm and the mode recover the optimal arm 1. This graph is also averaged over 1000 runs, and error bars depict 95% confidence intervals.

The results are similar in spirit to Figure 2. When σ is extremely large (relative to the ground truth vector $\mathbf{w}^\star = [1, 1]$), the weight space becomes less and less distinguishable with respect to the corresponding $\mathcal{Q}$ values. In particular, small differences between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}(\mathbf{w}^\star)$ again have a substantial effect on the corresponding optimal arms, causing a suboptimal arm to be recovered. At more reasonable levels of noise, however, we can see that the algorithm recovers the optimal arm 100% of the time.

The mode's performance also has a similar flavor to Figure 2. For a given value of δ, the regions of Figure 1 are completely decided. When σ is close to zero, the noise is almost negligible, and hence the blue region captures most of the density of the noise $\boldsymbol{\eta}$, and the optimal arm is the mode. But as σ is varied from 0 to ∞, the density captured by this region decreases continuously from 1 to a ratio of the volumes of the regions. In particular, we again come across a point where $\mathcal{Q}(\mathbf{w}^\star)$ has probability tied across the three arms, before which arm 1 is always the mode (i.e. mode has 100% performance), and after which arms 2 and 3 are the modes (i.e. the mode has 0% performance). Note that, for $\sigma = 1$, this point is achieved around $\delta = 1$ (Figure 2). Hence, when we vary σ while fixing $\delta = 1$, the tipping point is expected to be achieved around $\sigma = 1$, which is indeed the case, as evident from Figure 3. Again, observe that the performance of the algorithm is still around 100% significantly after this tipping point. Appendix J.2 includes similar results when the parameters are set to $\delta \in \{0.5, 2.0\}$ or $n \in \{250, 1000\}$.

6 Discussion

We have shown that it is possible to match the performance of the uniform mixture π^u, or that of the average agent. In Section 5 we then established that it is possible to learn policies from demonstrations with *superior* performance compared to the teacher, albeit under simplifying assumptions. An obvious challenge is to relax the assumptions, but this is very difficult, and we do not know of existing work that can be applied directly to our general setting. Indeed, the most relevant theoretical work is that of Syed and Schapire (2008). Their approach can only be applied if the sign of the reward weight is known for every feature. This is problematic in our setting as some agents may consider a feature to be positive, while others consider it to be negative. A priori, it is unclear how the sign can be determined, which crucially invalidates the algorithm's theoretical guarantees. Moreover, it is unclear under which cases the algorithm would produce a policy with superior performance, or if such cases exist.

We also remark that, although in the general setting we seek to compete with π^u, we are actually doing something quite different. Indeed, *ex post* (after the randomness has been instantiated) the uniform mixture π^u simply coincides with one of the individual policies. By contrast, IRL algorithms pool the feature expectations of the trajectories $\tau_1, \ldots, \tau_n$ together, and try to recover a policy that approximately matches them. Therefore, we believe that IRL algorithms do a much better job of aggregating the individual policies than π^u does, while giving almost the same optimality guarantees.

Ethics Statement

As mentioned in Section 1, our work can conceivably be used to align the values of an AI system with those of a group of people or even those of society. Although such an application would be far in the future, we acknowledge that it gives rise to ethical issues. Most notably, in cases where

because we use the empirical mode from $\tilde{\mathcal{Q}}$ whose performance varies smoothly as the distance between probabilities of arms 1 and $\{2, 3\}$ changes.

the values of the set of agents are not centered around a reasonable moral system, learning from those agents may lead to undesirable behavior. Thought must be given both to the choice of appropriate applications as well as to the selection of agents who serve as teachers.

References

Abbeel, P.; and Ng, A. Y. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 1–8.

Babeş-Vroman, M.; Marivate, V.; Subramanian, K.; and Littman, M. L. 2011. Apprenticeship Learning About Multiple Intentions. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 897–904.

Choi, J.; and Kim, K.-E. 2012. Nonparametric Bayesian Inverse Reinforcement Learning for Multiple Reward Functions. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 314–322.

Chomsky, N. 1965. *Aspects of the Theory of Syntax*. MIT Press.

Everitt, T.; Krakovna, V.; Orseau, L.; and Legg, S. 2017. Reinforcement Learning with a Corrupted Reward Channel. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 4705–4713.

Givan, R.; Leach, S.; and Dean, T. 2000. Bounded-Parameter Markov Decision Processes. *Artificial Intelligence* 122(1–2): 71–109.

Hadfield-Menell, D.; Russell, S. J.; Abbeel, P.; and Dragan, A. D. 2016. Cooperative Inverse Reinforcement Learning. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 3909–3917.

Hodges Jr, J. L.; and Lehmann, E. L. 1950. Some problems in minimax point estimation. *The Annals of Mathematical Statistics* 182–197.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research* 4: 237–285.

Kober, J.; Bagnell, J. A.; and Peters, J. 2013. Reinforcement Learning in Robotics: A Survey. *International Journal of Robotics Research* 32(11): 1238–1274.

Kocsis, L.; and Szepesvári, C. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, 282–293. Springer.

Mikhail, J. 2011. *Elements of Moral Cognition: Rawls' Linguistic Analogy and the Cognitive Science of Moral and Legal Judgment*. Cambridge University Press.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-Level Control Through Deep Reinforcement Learning. *Nature* 518: 529–533.

Ng, A. Y.; and Russell, S. 2000. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 663–670.

Noothigattu, R.; Gaikwad, S. S.; Awad, E.; Dsouza, S.; Rahwan, I.; Ravikumar, P.; and Procaccia, A. D. 2018. A Voting-Based System for Ethical Decision Making. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, 1587–1594.

Perron, F.; and Marchand, E. 2002. On the minimax estimator of a bounded normal mean. *Statistics and Probability Letters* 58: 327–333.

Rawls, J. 1971. *A Theory of Justice*. Harvard University Press.

Regan, K.; and Boutilier, C. 2009. Regret-based Reward Elicitation for Markov Decision Processes. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 444–451.

Regan, K.; and Boutilier, C. 2010. Robust Policy Computation in Reward-uncertain MDPs using Nondominated Policies. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, 1127–1133.

Russell, S.; Dewey, D.; and Tegmark, M. 2015. Research Priorities for Robust and Beneficial Artificial Intelligence. *AI Magazine* 36(4): 105–114.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* 529: 484–489.

Sutton, R. S.; and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Syed, U.; and Schapire, R. E. 2008. A Game-Theoretic Approach to Apprenticeship Learning. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NeurIPS)*, 1449–1456.

Wainwright, M. J. 2019. *High-dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge University Press.

Wu, Y.-H.; and Lin, S.-D. 2018. A Low-Cost Ethics Shaping Approach for Designing Reinforcement Learning Agents. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, 1687–1694.

Zheng, J.; Liu, S.; and Ni, L. M. 2014. Robust Bayesian Inverse Reinforcement Learning with Sparse Behavior Noise. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, 2198–2205.

Ziebart, B. D. 2010. Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy. Ph.D. thesis, Carnegie Mellon University.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; and Dey, A. K. 2008. Maximum Entropy Inverse Reinforcement Learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, 1433–1438.

Appendix

A IRL Algorithms

In this appendix we identify two well-known algorithms that match feature expectations.

A.1 Apprenticeship Learning

Under the classic Apprenticeship Learning algorithm, designed by Abbeel and Ng (2004), a policy $\pi^{(0)}$ is selected to begin with. Its feature expectation $\mu(\pi^{(0)})$ is computed and added to the bag of feature expectations. At each step,

$$t^{(i)} = \max_{\mathbf{w}:\|\mathbf{w}\|_2 \leq 1} \min_{j \in \{0,..,i-1\}} \mathbf{w}^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) - \mu\left(\pi^{(j)}\right) \right)$$

is computed along with the weight $\mathbf{w}^{(i)}$ that achieved this. When $t^{(i)} \leq \epsilon$ the algorithm terminates, otherwise the associated optimal policy $\pi^{(i)}$ is computed, and its corresponding feature expectation vector $\mu(\pi^{(i)})$ is added to the bag of feature expectations. The algorithm provides the following guarantee.

Theorem A.1 (adapted from Abbeel and Ng (2004))**.** *For any $\epsilon > 0$, the Apprenticeship Learning algorithm terminates with $t^{(i)} \leq \epsilon$ after a number of iterations bounded by*

$$T = O\left(\frac{d}{(1-\gamma)^2 \epsilon^2} \ln \frac{d}{(1-\gamma)\epsilon} \right),$$

and outputs a mixture over $\pi^{(1)}, ..., \pi^{(T)}$ that ϵ-matches the feature expectations of the observed trajectories.

Note that it is necessary for us to use a randomized policy, in contrast to the case where a single deterministic policy generated all the trajectory samples, as, in our case, typically there is no single deterministic policy that matches the feature expectations of the observed trajectories.

A.2 Max Entropy

We next discuss the Max Entropy algorithm of Ziebart et al. (2008), which optimizes the max entropy of the probability distribution over trajectories subject to the distribution satisfying approximate feature matching. This is done to resolve the potential ambiguity of there being multiple stochastic policies that satisfy feature matching. Optimizing entropy is equivalent to maximizing the regularized likelihood $L(\mathbf{w})$ of the observed trajectories. Specifically, the objective is

$$L(\mathbf{w}) = \max_{\mathbf{w}} \sum_{i=1}^{n} \log \Pr[\tau_i | \mathbf{w}, T] - \sum_{i=1}^{d} \rho_i \|\mathbf{w}_i\|_1,$$

with

$$\Pr[\tau_i | \mathbf{w}, T] = \frac{e^{\mathbf{w}^\intercal \phi(\tau_i)}}{Z(\mathbf{w}, T)} \prod_{s_t, a_t, s_{t+1} \in \tau_i} T(s_t, a_t, s_{t+1}).$$

The regularization term is introduced to allow for approximate feature matching since the observed empirical feature expectation may differ from the true expectation. Let $\boldsymbol{\rho}$ be an upper bound on this difference, i.e., for all $k = 1, \ldots, d$,

$$\rho_k \geq \left| \frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i)_k - \mathbb{E}\left[\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i)_k \right] \right|.$$

One may then derive that the gradient of $L(\mathbf{w})$ is the difference between the feature expectation induced $\mathbf{w}$ and the observed feature expectation.

Theorem A.2 (adapted from Ziebart et al. (2008))**.** *Let $\epsilon > 0$, and assume that the Max Entropy algorithm finds $\mathbf{w}$ such that $|\nabla L(\mathbf{w})| < \epsilon$, then this $\mathbf{w}$ corresponds to a randomized policy that $(\epsilon + \|\boldsymbol{\rho}\|_1)$-matches the feature expectations of the observed trajectories.*

The assumption on the gradient is needed because the above optimization objective is derived only with the approximate feature matching constraint. MDP dynamics is not explicitly encoded into the optimization. Instead, heuristically, the likelihood of each trajectory $\Pr[\tau_i | \mathbf{w}, T]$ is weighted by the product of the transition probabilities of its steps. The follow-up work of Ziebart (2010) addresses this by explicitly introducing MDP constraints into the optimization, and optimizing for the causal entropy, thereby achieving unconditional feature matching.

B Proof of Theorem 3.2

We need to bound the difference between $R^{\mathbf{w}^\star}(\tilde{\pi})$ and $R^{\mathbf{w}^\star}(\pi^u)$. First, recall that $\tilde{\pi}$ $\epsilon/3-$matches the feature expectations of $\tau_1, \ldots, \tau_n$. It holds that

$$\begin{aligned}\left| R^{\mathbf{w}^\star}(\tilde{\pi}) - (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) \right| &= \left| (\mathbf{w}^\star)^\intercal \left(\mu(\tilde{\pi}) - \frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) \right| \\ &\leq \|\mathbf{w}^\star\|_2 \left\| \mu(\tilde{\pi}) - \frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right\|_2 \leq \frac{\epsilon}{3},\end{aligned} \tag{2}$$

where the second transition follows from the Cauchy-Schwarz inequality, and the last from the assumption that $\|\mathbf{w}^\star\|_2 \leq 1$. Hence, it is sufficient to demonstrate that, with probability at least $1 - \delta$,

$$\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) - R^{\mathbf{w}^\star}(\pi^u) \right| \leq \frac{2\epsilon}{3}, \tag{3}$$

as the theorem would then follow from Equations (2), and (3) by the triangle inequality.

We note that the difference on the left hand side of Equation (3) is due to two sources of noise.

1. The finite number of samples of trajectories which, in our setting, originates from multiple policies.
2. The truncated trajectories τ_i which are limited to L steps.

Formally, let τ_i' denote the infinite trajectory for each i, then the difference can be written as

$$\begin{aligned}\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) - R^{\mathbf{w}^\star}(\pi^u) \right| \leq &\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) - (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i') \right) \right| \\ &+ \left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i') \right) - R^{\mathbf{w}^\star}(\pi^u) \right|\end{aligned}$$

Bounding finite sample noise. We wish to bound:

$$\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i') \right) - R^{\mathbf{w}^\star}(\pi^u) \right| = \left| \frac{1}{n} \left(\sum_{i=1}^{n} (\mathbf{w}^\star)^\intercal (\phi(\tau_i') - \mu(\pi_i)) \right) \right|. \tag{4}$$

Define random variable $Z_i = (\mathbf{w}^\star)^\intercal(\phi(\tau_i') - \mu(\pi_i))$. Then the right-hand side of Equation (4) may be expressed as $|\frac{1}{n}\sum_{i=1}^{n} Z_i|$. Furthermore, Z_i is such that $\mathbb{E}[\phi(\tau_i')_k] = \mu(\pi_i)_k$ for all $k = 1, \ldots, d$. This is because a policy π_i defines a distribution over trajectories, and τ_i' is a draw from this distribution. Using the linearity of expectation, it follows that

$$\mathbb{E}[Z_i] = (\mathbf{w}^\star)^\intercal \mathbb{E}[\phi(\tau_i') - \mu(\pi_i)] = 0.$$

Moreover,

$$|Z_i| \leq \|\mathbf{w}^\star\|_2 \|\phi(\tau_i')\|_2 + \|\mathbf{w}^\star\|_2 \|\mu(\pi_i)\|_2 \leq \frac{2\sqrt{d}}{1-\gamma},$$

since $\|\phi(s, \cdot)\|_\infty = 1$. Thus, using Hoeffding's inequality, we conclude that

$$\Pr\left[\left| \frac{1}{n} \sum_{i=1}^{n} Z_i \right| > \frac{\epsilon}{3} \right] \leq 2\text{exp}\left(- \frac{2n\left(\frac{\epsilon}{3}\right)^2}{(\frac{4\sqrt{d}}{1-\gamma})^2} \right) \leq \delta,$$

where the last transition holds by our choice of n.

Bounding bias due to truncated trajectories. We wish to bound:

$$\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) - (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i') \right) \right|.$$

For each trajectory τ_i, truncating after L steps incurs a reward difference of:

$$|(\mathbf{w}^\star)^\intercal \phi(\tau_i') - (\mathbf{w}^\star)^\intercal \phi(\tau_i)| = \left| (\mathbf{w}^\star)^\intercal \sum_{t=L}^{\infty} \gamma^t \phi(\tau_i'(s_t), \tau_i'(a_t)) \right|$$

$$\leq \sum_{t=L}^{\infty} \gamma^t \|\mathbf{w}^\star\|_2 \|\phi(\tau_i'(s_t), \tau_i'(a_t))\|_2 \leq \gamma^L \frac{\sqrt{d}}{1-\gamma} \leq \frac{\epsilon}{3},$$

where the third transition holds because $\|\phi(\tau_i(s_t), \tau_i(a_t))\|_2 \leq \sqrt{d}$, and the last transition follows from our choice of L. Hence, we obtain

$$\left| (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i) \right) - (\mathbf{w}^\star)^\intercal \left(\frac{1}{n} \sum_{i=1}^{n} \phi(\tau_i') \right) \right| \leq \frac{1}{n} \sum_{i=1}^{n} |(\mathbf{w}^\star)^\intercal \phi(\tau_i) - (\mathbf{w}^\star)^\intercal \phi(\tau_i')| \leq \frac{\epsilon}{3}.$$

□

C Proof of Theorem 4.1

We require a key property of sub-exponential random variables, which is captured by the following well known tail inequality; its proof can be found, for example, in Chapter 2 of Wainwright (2019).

Lemma C.1. *Let $X_1, \ldots, X_m$ be independent sub-exponential random variables with parameters (ν, b). Then*

$$\Pr\left[\frac{1}{m} \sum_{j=1}^{m} (X_j - u_j) \geq t \right] \leq \begin{cases} \exp\left(-\frac{mt^2}{2\nu^2}\right) \text{ for } 0 \leq t \leq \frac{\nu^2}{b} \\ \exp\left(-\frac{mt}{2b}\right) \text{ for } t > \frac{\nu^2}{b} \end{cases},$$

where $u_j = \mathbb{E}[X_j]$.

Turning to the theorem's proof, as π^u is a uniform distribution over the policies $\pi_1, \ldots, \pi_n$, its expected reward is given by

$$R^{\mathbf{w}^\star}(\pi^u) = \frac{1}{n} \sum_{i=1}^{n} R^{\mathbf{w}^\star}(\pi_i). \tag{5}$$

Observe that $R^{\mathbf{w}^\star}(\pi_i)$ is a random variable which is i.i.d. across i, as the corresponding noise $\boldsymbol{\eta}_i$ is i.i.d. as well. We analyze the expectation of the difference with respect to $R^{\mathbf{w}^\star}(\pi^\star)$.

First, note that for a weight vector $\mathbf{w}$ and policy π,

$$R^{\mathbf{w}}(\pi) = \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} P_\pi(s,t) \mathbf{w}^\intercal \phi(s, \pi(s)), \tag{6}$$

where $P_\pi(s,t)$ denotes the probability of being in state s on executing policy π from the start. Hence, for each $i \in N$, we have

$$\begin{aligned}
&R^{\mathbf{w}^\star}(\pi^\star) - R^{\mathbf{w}^\star}(\pi_i) \\
&= \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} \Big[P_{\pi^\star}(s,t) (\mathbf{w}^\star)^\intercal \phi(s, \pi^\star(s)) - P_{\pi_i}(s,t) (\mathbf{w}^\star)^\intercal \phi(s, \pi_i(s)) \Big] \\
&= \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} \Big[P_{\pi^\star}(s,t) (\mathbf{w}_i - \boldsymbol{\eta}_i)^\intercal \phi(s, \pi^\star(s)) - P_{\pi_i}(s,t) (\mathbf{w}_i - \boldsymbol{\eta}_i)^\intercal \phi(s, \pi_i(s)) \Big] \\
&= R^{\mathbf{w}_i}(\pi^\star) - R^{\mathbf{w}_i}(\pi_i) + \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} \Big[- P_{\pi^\star}(s,t) \boldsymbol{\eta}_i^\intercal \phi(s, \pi^\star(s)) + P_{\pi_i}(s,t) \boldsymbol{\eta}_i^\intercal \phi(s, \pi_i(s)) \Big] \\
&\leq \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} \Big[- P_{\pi^\star}(s,t) \boldsymbol{\eta}_i^\intercal \phi(s, \pi^\star(s)) + P_{\pi_i}(s,t) \boldsymbol{\eta}_i^\intercal \phi(s, \pi_i(s)) \Big] \\
&= \sum_{k=1}^{d} \eta_{ik} \left[\sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} \Big[- P_{\pi^\star}(s,t) \phi(s, \pi^\star(s))_k + P_{\pi_i}(s,t) \phi(s, \pi_i(s))_k \Big] \right] \\
&:= \sum_{k=1}^{d} \eta_{ik} \alpha_{ik},
\end{aligned} \tag{7}$$

where the inequality holds since $R^{\mathbf{w}_i}(\pi_i) \geq R^{\mathbf{w}_i}(\pi^\star)$, which, in turn, holds because π_i is optimal under $\mathbf{w}_i$.

Using the assumption that $\|\phi(s,a)\|_\infty \leq 1$, it holds that $\left|\sum_{s\in S} P_\pi(s,t)\phi(s,a)_k\right| \leq 1$ for any policy π. We can therefore bound $|\alpha_{ik}|$ as follows.

$$\begin{aligned}
|\alpha_{ik}| &\leq \sum_{t=0}^{\infty} \gamma^t \left| \sum_{s\in S} [-P_{\pi^\star}(s,t)\phi(s,\pi^\star(s))_k + P_{\pi_i}(s,t)\phi(s,\pi_i(s))_k] \right| \\
&\leq \sum_{t=0}^{\infty} \gamma^t \left[\left| \sum_{s\in S} P_{\pi^\star}(s,t)\phi(s,\pi^\star(s))_k \right| + \left| \sum_{s\in S} P_{\pi_i}(s,t)\phi(s,\pi_i(s))_k \right| \right] \\
&\leq \frac{2}{1-\gamma}.
\end{aligned}$$

Therefore, it holds that

$$\|\boldsymbol{\alpha}_i\|_2 = \sqrt{\sum_{k=1}^{d} \alpha_{ik}^2} \leq \sqrt{\sum_{k=1}^{d} \left(\frac{2}{1-\gamma}\right)^2} = \frac{2\sqrt{d}}{(1-\gamma)}.$$

Using this bound along with Equation (7), we obtain

$$\begin{aligned}
R^{\mathbf{w}^\star}(\pi^\star) - R^{\mathbf{w}^\star}(\pi_i) &\leq \sum_{k=1}^{d} \eta_{ik}\alpha_{ik} \leq \|\boldsymbol{\eta}_i\|_2 \|\boldsymbol{\alpha}_i\|_2 \leq \frac{2\sqrt{d}}{(1-\gamma)} \sqrt{\sum_{k=1}^{d} \eta_{ik}^2} \\
&= \frac{2d}{(1-\gamma)} \sqrt{\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2}. \qquad (8)
\end{aligned}$$

Denote $u = \mathbb{E}[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2]$. To compute the expected value of the previous expression (with respect to the randomness of the noise $\boldsymbol{\eta}_i$), we analyze

$$\begin{aligned}
\mathbb{E}\left[\sqrt{\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2}\right] &= \int_0^\infty \Pr\left[\sqrt{\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2} \geq x\right] dx = \int_0^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq x^2\right] dx \\
&= \int_0^{\sqrt{u}} \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq x^2\right] dx + \int_{\sqrt{u}}^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq x^2\right] dx \\
&\leq \int_0^{\sqrt{u}} 1\, dx + \int_{\sqrt{u}}^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq x^2\right] dx \\
&= \sqrt{u} + \int_0^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq u+t\right] \frac{1}{2\sqrt{u+t}} dt \\
&\leq \sqrt{u} + \frac{1}{2\sqrt{u}} \int_0^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq u+t\right] dt,
\end{aligned}$$

where the fourth transition is obtained by changing the variable using $x = \sqrt{u+t}$. But since each η_{ik}^2 is sub-exponential with parameters (ν, b), from Lemma C.1 we have

$$\Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq u+t\right] \leq \begin{cases} \exp\left(-\frac{dt^2}{2\nu^2}\right) & \text{for } 0 \leq t \leq \frac{\nu^2}{b} \\ \exp\left(-\frac{dt}{2b}\right) & \text{for } t > \frac{\nu^2}{b} \end{cases}.$$

Plugging this into the upper bound for the expected value gives us

$$\begin{aligned}
\mathbb{E}\left[\sqrt{\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2}\right] &\leq \sqrt{u} + \frac{1}{2\sqrt{u}} \int_0^\infty \Pr\left[\frac{1}{d}\sum_{k=1}^{d} \eta_{ik}^2 \geq u+t\right] dt \\
&\leq \sqrt{u} + \frac{1}{2\sqrt{u}} \left[\int_0^{\frac{\nu^2}{b}} \exp\left(-\frac{dt^2}{2\nu^2}\right) dt + \int_{\frac{\nu^2}{b}}^\infty \exp\left(-\frac{dt}{2b}\right) dt\right]
\end{aligned}$$

$$
\begin{aligned}
&= \sqrt{u} + \frac{1}{2\sqrt{u}} \left[\int_0^{\frac{\nu\sqrt{d}}{b}} \exp\left(-\frac{z^2}{2}\right) \frac{\nu}{\sqrt{d}} dz + \left(-\frac{2b}{d}\right) \exp\left(-\frac{dt}{2b}\right) \Big|_{\frac{\nu^2}{b}}^{\infty} \right] \\
&= \sqrt{u} + \frac{1}{2\sqrt{u}} \left[\sqrt{\frac{2\pi}{d}} \nu \int_0^{\frac{\nu\sqrt{d}}{b}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right) dz + \frac{2b}{d} \exp\left(-\frac{d\nu^2}{2b^2}\right) \right] \\
&= \sqrt{u} + \frac{1}{2\sqrt{u}} \left[\sqrt{\frac{2\pi}{d}} \nu \left(\Phi\left(\frac{\nu\sqrt{d}}{b}\right) - \frac{1}{2} \right) + \frac{2b}{d} \exp\left(-\frac{d\nu^2}{2b^2}\right) \right] \\
&= \sqrt{u} + \sqrt{\frac{\pi}{2ud}} \nu \left(\Phi\left(\frac{\nu\sqrt{d}}{b}\right) - \frac{1}{2} \right) + \frac{b}{d\sqrt{u}} \exp\left(-\frac{d\nu^2}{2b^2}\right), \qquad (9)
\end{aligned}
$$

where the transition in the third line is obtained by changing the variable using $t = \frac{v}{\sqrt{d}} z$, and Φ denotes the CDF of a standard normal distribution. Hence, taking an expected value for Equation (8) and plugging in Equation (9), we obtain

$$
\mathbb{E}\left[R^{\mathbf{w}^\star}(\pi^\star) - R^{\mathbf{w}^\star}(\pi_i) \right] \leq \frac{2d}{(1-\gamma)} \left[\sqrt{u} + \sqrt{\frac{\pi}{2ud}} \nu \left(\Phi\left(\frac{\nu\sqrt{d}}{b}\right) - \frac{1}{2} \right) + \frac{b}{d\sqrt{u}} \exp\left(-\frac{d\nu^2}{2b^2}\right) \right].
$$

Rearranging this equation, we have

$$
\mathbb{E}\left[R^{\mathbf{w}^\star}(\pi_i) \right] \geq R^{\mathbf{w}^\star}(\pi^\star) - \frac{2d}{(1-\gamma)} \left[\sqrt{u} + \sqrt{\frac{\pi}{2ud}} \nu \left(\Phi\left(\frac{\nu\sqrt{d}}{b}\right) - \frac{1}{2} \right) + \frac{b}{d\sqrt{u}} \exp\left(-\frac{d\nu^2}{2b^2}\right) \right].
$$

Taking an expectation over Equation (5) gives us $\mathbb{E}\left[R^{\mathbf{w}^\star}(\pi^u)\right] = \mathbb{E}\left[R^{\mathbf{w}^\star}(\pi_i)\right]$, and the theorem directly follows. □

We remark that Theorem 4.1 can easily be strengthened to obtain a high probability result (at the cost of complicating its statement). Indeed, the reward of the uniform mixture $R^{\mathbf{w}^\star}(\pi^u)$ is the average of the individual policy rewards $R^{\mathbf{w}^\star}(\pi_i)$, which are i.i.d. Further, each of these rewards is bounded, because of the constraints on $\mathbf{w}^\star$ and ϕ. Hence, Hoeffding's inequality would show that $R^{\mathbf{w}^\star}(\pi^u)$ strongly concentrates around its mean.

D Example for the Tightness of Theorem 4.1

Assume $\eta_{ik} \sim \mathcal{N}(0, \sigma^2)$ with $\sigma \leq 2/d$ (to avoid violating the constraint $\|\phi(s,a)\|_\infty \leq 1$). Suppose the MDP has just one state and $2^{d-1}+1$ actions. One action has feature vector $(d\sigma/2, 0, \ldots, 0)$, and for each subset $S \subseteq \{2, \ldots, d\}$, there is an action a_S with a binary feature vector such that it is 1 for coordinates in S and 0 everywhere else. Let $w^\star = (1, 0, ..., 0)$. The optimal policy is to pick the first action which has cumulative reward of $\frac{d\sigma}{2(1-\gamma)}$. As $\eta_{ik} \sim \mathcal{N}(0, \sigma^2)$ for each k, with constant probability, roughly $d/2$ of the coordinates of the noised vector reward $\mathbf{w}_i$ will deviate by roughly $+\sigma$ and the first coordinate will not increase too much. In this case, the action corresponding to the coordinates with positive deviations will have reward on the order of $d\sigma/2$, beating action 1 to become optimal. Hence, this would lead to π_i picking this action and having 0 reward under $\mathbf{w}^\star$. As this occurs with constant probability for a policy in the data, and π^u is simply a mean of their rewards, its expected value would deviate from the optimum by at least a constant fraction of $d\sigma/2$.

E Empirical Results for the MDP setting

As we have seen in Section 4.1, the gap between $R^{\mathbf{w}^\star}(\pi^\star)$ and $R^{\mathbf{w}^\star}(\pi^u)$ is upper bounded by $O(d\sqrt{u} + \nu\sqrt{d/u} + b/\sqrt{u})$ when η_{ik}^2 is sub-exponential, or $O(d\sigma)$ when η_{ik} is Gaussian. Further, Section 3 shows that a policy $\tilde{\pi}$ that matches feature expectations of the observed trajectories is very close to π^u in terms of cumulative reward $R^{\mathbf{w}^\star}$. In this appendix, we empirically examine the gaps between $\tilde{\pi}$ (obtained by a "feature matching" IRL algorithm), π^u and $\pi^\star$.

E.1 Methodology

As our IRL algorithm we use Apprenticeship Learning, which guarantees the feature-matching property (see Section 3 and Appendix A). By Theorem 3.2 we may safely assume that any IRL algorithm that matches feature expectations would have essentially identical rewards, and therefore would show very similar behavior in our experiments.

We perform our experiments in the following two domains.

Grab a Milk. We adapt the "Grab a Milk" MDP, a route planning RL domain (Wu and Lin 2018), to our setting. The MDP is defined by a 10 by 10 grid room, where the agent starts at $(0, 0)$ and has to reach a bottle of milk positioned at $(9, 9)$. There are also 16 babies in the room, 5 of which are crying for attention. When the agent crosses a crying baby, they can help soothe the baby, but on crossing a non-crying baby, the agent disturbs the baby. Hence, the goal of this task is to minimize the number of steps to the milk, while at the same time soothing as many crying babies as possible along the way and avoiding crossing

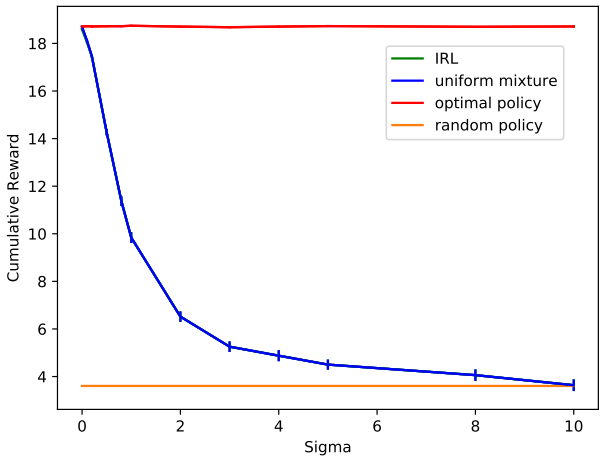


Figure 4: Performance on the Sailing MDP. Error bars show 95% confidence intervals.

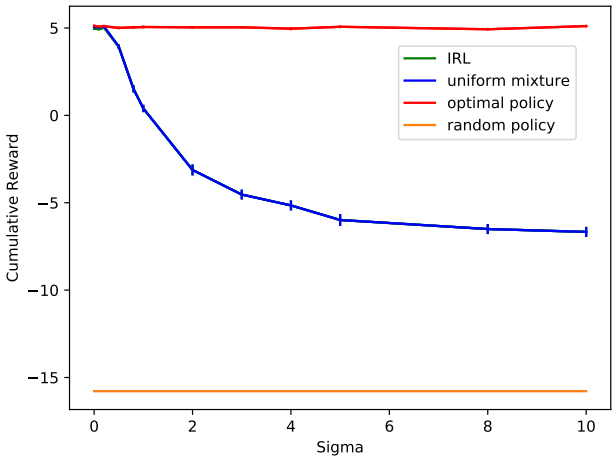


Figure 5: Performance on the Grab a Milk MDP. Error bars show 95% confidence intervals.

non-crying babies. This MDP is adapted to our setting, by defining each state (or grid square) to have three features $\phi(s)$.[8] The first feature captures the reward of taking a step, and is set to -1 if the state is non-terminal, whereas it is set to 5 for the terminal state $(9, 9)$. The second is a boolean feature depicting whether there is a crying baby in the particular grid square, and similarly the third is a boolean feature depicting whether there is a non-crying baby in the particular grid square. The rewards in the MDP are then defined as $r^{\mathbf{w}^\star}(s) = (\mathbf{w}^\star)^\intercal \phi(s)$ where the ground truth weight vector is given by $\mathbf{w}^\star = [1, 0.5, -0.5]$. Intuitively, this weight vector $\mathbf{w}^\star$ can be interpreted as the weights for different ethical factors, and each member of society has a noised version of this weight.

Sailing. The other domain we use is a modified version of the "Sailing" MDP (Kocsis and Szepesvári 2006). The Sailing MDP is also a gridworld domain (we use the same size of 10 by 10), where there is a sailboat starting at $(0, 0)$ and navigating the grid under fluctuating wind conditions. The goal of the MDP is to reach a specified grid square as quickly as possible. We adapt this domain to our setting by removing the terminal state, and instead adding features for each grid square.[9] Now, the goal of the agent is not to reach a certain point as quickly as possible, but to navigate this grid while maximizing (or minimizing) the weighted sum of these features. We use 10 features for each grid square, and these are independently sampled from a uniform distribution over $(-1, 1)$. The ground truth weight vector $\mathbf{w}^\star$, which defines the weights of these features for the net reward, is also randomly sampled from independent Unif$(-1, 1)$ for each coordinate. As before, this weight vector $\mathbf{w}^\star$ can be interpreted as the weights for different bounties, and each member has a noised version of this weight.

Being gridworld domains, in both the MDPs, the agent has four actions to choose from at each state (one for each direction). The transition dynamics are as follows: On taking a particular action from a given state, the agent moves in that direction with probability 0.95, but with a probability of 0.05 it moves in a different direction uniformly at random. We use a discount factor of 0.95 in both domains.

We generate the trajectories $\{\tau_1, \ldots, \tau_n\}$ as described in Section 3, and use a Gaussian distribution for the noise. That is, $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2 I_d)$. We generate a total of $n = 50$ trajectories, each of length $L = 30$. IRL is then performed on this data and we analyze its reward as σ is varied. A learning rate of 0.001 is used for the Apprenticeship Learning algorithm.

E.2 Results

Figures 4 and 5 show the performance of π^u and the IRL algorithm as σ is varied. We also include the performance of $\pi^\star$ and a purely random policy π^r (which picks a uniformly random action at each step), as references. Each point in these graphs is averaged over 50 runs (of data generation).

For both domains, the first thing to note is that the uniform mixture π^u and the IRL algorithm have nearly identical rewards, which is why the green IRL curve is almost invisible. This confirms that matching feature expectations leads to performance approximating the uniform mixture.

Next, as expected, one can observe that as σ increases, the gap between $R^\star(\pi^\star)$ and $R^\star(\pi^u)$ also increases. Further, for both domains, this gap saturates around $\sigma = 10$ and the $R^\star(\pi^u)$ curve flattens from there (hence, we do not include larger values of

[8] For these MDPs, the rewards depend only on the states and not state-action pairs, and hence the reward function can be defined as $r^{\mathbf{w}}(s, a) = r^{\mathbf{w}}(s) = \mathbf{w}^\intercal \phi(s)$.

[9] Intuitively, these features could represent aspects like "abundance of fish" in that grid square for fishing, "amount of trash" in that square that could be cleaned up, "possible treasure" for treasure hunting, etc.

σ in either graph). Note that, in both domains, the ground truth weight vector $\mathbf{w}^\star$ is generated such that $\|\mathbf{w}^\star\|_\infty \leq 1$. Hence, a standard deviation of 10 in the noise overshadows the true weight vector $\mathbf{w}^\star$, leading to the large gap shown in both graphs. Looking at more reasonable levels of noise (with respect to the norm of the weights), like $\sigma \in [0, 1]$, we can see that $R^\star(\pi^u)$ drops approximately linearly, as suggested by Theorem 4.1. In particular, it is 14.27 at $\sigma = 0.5$ and 9.84 at $\sigma = 1.0$ for Sailing, and it is 3.93 at $\sigma = 0.5$ and 0.39 at $\sigma = 1.0$ for Grab a Milk.

Finally, we compare the performance of π^u with that of the purely random policy π^r. As σ becomes very large, each $\mathbf{w}_i$ is distributed almost identically across the coordinates. Nevertheless, because of the structure of the Grab a Milk MDP, $R^\star(\pi^u)$ still does significantly better than $R^\star(\pi^r)$. By contrast, Sailing has features that are sampled i.i.d. from Unif$(-1, 1)$ for each state, which leads the two policies, π^u and π^r, to perform similarly for large values of σ.

F Proof of Lemma 4.2

Before proving the lemma, we look at a relatively simple example that we will use later to complete the proof.

F.1 Simpler Example

Consider an MDP with a single state s, and three actions $\{a, b, c\}$. Since s is the only state, $T(s, a, s) = T(s, b, s) = T(s, c, s) = 1$, and D is degenerate at s. This implies that there are only three possible policies, denoted by π_a, π_b, π_c (which take actions a, b, c respectively from s). Let the feature expectations be

$$\begin{aligned}\phi(s, a) &= [0.5, 0.5],\\ \phi(s, b) &= [1, -\delta/2],\\ \phi(s, c) &= [-\delta/2, 1],\end{aligned}$$

where $\delta > 0$ is a parameter. Hence, the feature expectations of the policies $\{\pi_a, \pi_b, \pi_c\}$ are respectively

$$\begin{aligned}\mu_a &= \frac{1}{2(1-\gamma)}[1, 1],\\ \mu_b &= \frac{1}{2(1-\gamma)}[2, -\delta],\\ \mu_c &= \frac{1}{2(1-\gamma)}[-\delta, 2].\end{aligned}$$

Let the ground truth weight vector be $\mathbf{w}^\star = (v_o, v_o)$, where v_o is a "large enough" positive constant. In particular, v_o is such that the noised weight vector $\mathbf{w} = \mathbf{w}^\star + \boldsymbol{\eta}$ has probability strictly more than $1/3$ of lying in the first quadrant. For concreteness, set v_o to be such that $\Pr(\mathbf{w} > 0) = 1/2$. Such a point always exists for any noise distribution (that is continuous and i.i.d. across coordinates). Specifically, it is attained at $v_o = -F^{-1}(1 - \frac{1}{\sqrt{2}})$, where F^{-1} is the inverse CDF of each coordinate of the noise distribution. This is because at this value of v_o,

$$\begin{aligned}\Pr(\mathbf{w} > 0) &= \Pr((v_o, v_o) + (\eta_1, \eta_2) > 0) = \Pr(v_o + \eta_1 > 0)^2\\ &= \Pr(\eta_1 > -v_o)^2 = (1 - F(-v_o))^2 = \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}.\end{aligned}$$

Let us look at weight vectors $\mathbf{w}$ for which each of the three policies π_a, π_b and π_c are optimal. π_a is the optimal policy when $\mathbf{w}^\intercal\mu_a > \mathbf{w}^\intercal\mu_b$ and $\mathbf{w}^\intercal\mu_a > \mathbf{w}^\intercal\mu_c$, which is the intersection of the half-spaces $\mathbf{w}^\intercal(-1, 1+\delta) > 0$ and $\mathbf{w}^\intercal(1+\delta, -1) > 0$. On the other hand, π_b is optimal when $\mathbf{w}^\intercal\mu_b > \mathbf{w}^\intercal\mu_a$ and $\mathbf{w}^\intercal\mu_b > \mathbf{w}^\intercal\mu_c$, which is the intersection of the half-spaces $\mathbf{w}^\intercal(-1, 1+\delta) < 0$ and $\mathbf{w}^\intercal(1, -1) > 0$. Finally, π_c is optimal when $\mathbf{w}^\intercal\mu_c > \mathbf{w}^\intercal\mu_a$ and $\mathbf{w}^\intercal\mu_c > \mathbf{w}^\intercal\mu_b$, which is the intersection of the half-spaces $\mathbf{w}^\intercal(1+\delta, -1) < 0$ and $\mathbf{w}^\intercal(1, -1) < 0$. These regions are illustrated in Figure 1 for different values of δ. Informally, as δ is decreased, the lines separating (π_a, π_c) and (π_a, π_b) move closer to each other (as shown for $\delta = 0.25$), while as δ is increased, these lines move away from each other (as shown for $\delta = 10$).

Formally, let R_δ denote the region of $\mathbf{w}$ for which π_a is optimal (i.e. the blue region in the figures), that is,

$$R_\delta = \left\{\mathbf{w} : \frac{w_1}{1+\delta} < w_2 < w_1(1+\delta)\right\}.$$

This is bounded below by the line $w_1 = (1+\delta)w_2$, which makes an angle of $\theta_\delta = \text{Tan}^{-1}(\frac{1}{1+\delta})$ with the x-axis, and bounded above by the line $w_2 = (1+\delta)w_1$, which makes an angle of θ_δ with the y-axis. We first show that for any value of δ, the regions of π_b and π_c have the exact same probability. The probability that π_b is optimal is the probability of the orange region which is

$$\Pr(\pi_b \text{ is optimal}) = \int_{-\infty}^{0}\int_{-\infty}^{w_1} \Pr(\mathbf{w})dw_2dw_1 + \int_{0}^{\infty}\int_{-\infty}^{\frac{w_1}{(1+\delta)}} \Pr(\mathbf{w})dw_2dw_1$$

$$
\begin{aligned}
&= \int_{-\infty}^{0}\int_{-\infty}^{t_2} \Pr(t_2, t_1)dt_1dt_2 + \int_{0}^{\infty}\int_{-\infty}^{\frac{t_2}{(1+\delta)}} \Pr(t_2, t_1)dt_1dt_2 \\
&= \int_{-\infty}^{0}\int_{-\infty}^{t_2} \Pr(t_1, t_2)dt_1dt_2 + \int_{0}^{\infty}\int_{-\infty}^{\frac{t_2}{(1+\delta)}} \Pr(t_1, t_2)dt_1dt_2 \\
&= \Pr(\pi_c \text{ is optimal}),
\end{aligned}
$$

where the second equality holds by changing the variables as $t_1 = w_2$ and $t_2 = w_1$, and the third one holds because the noise distribution is i.i.d. across the coordinates. Hence, we have

$$
\Pr(\pi_b \text{ is optimal}) = \Pr(\pi_c \text{ is optimal}) = \frac{1 - \Pr(R_\delta)}{2},
$$

as R_δ denotes the region where π_a is optimal.

Finally, we show that there exists a value of δ such that $\Pr(R_\delta) = 1/3$. Observe that as $\delta \to 0$, the lines bounding the region R_δ make angles that approach $Tan^{-1}(1) = \pi/4$ and the two lines touch, causing the region to have zero probability. On the other hand, as $\delta \to \infty$, the angles these lines make approach $Tan^{-1}(0) = 0$, so the region coincides with the first quadrant in the limit. Based on our selection of v_o, the probability of this region is exactly $1/2$. Hence, as δ varies from 0 to ∞, the probability of the region R_δ changes from 0 to $1/2$. Next, note that as $\theta_\delta = \text{Tan}^{-1}(\frac{1}{1+\delta})$, this angle changes continuously as δ changes, and hence does the region R_δ. Finally, as the noise distribution is continuous, the probability of this region R_δ also changes continuously as δ is varied. That is, $\lim_{\epsilon\to 0} \Pr(R_{\delta+\epsilon}) = \Pr(R_\delta)$. Coupling this with the fact that $\Pr(R_\delta)$ changes from 0 to $1/2$ as δ changes from 0 to ∞, it follows that there exists a value of δ in between such that $\Pr(R_\delta)$ is exactly $1/3$. Denote this value of δ by δ_o.

We conclude that for $\mathbf{w}^\star = (v_o, v_o)$ and our MDP construction with $\delta = \delta_o$, $\mathcal{P}(\mathbf{w}^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

F.2 Completing the Proof

Consider the same MDP as in Section F.1. However, for this example, let the feature expectations be

$$
\begin{aligned}
\phi(s, a) &= [0.5, 0.5 \ , \ -\delta_o/2, 1], \\
\phi(s, b) &= [1, -\delta_o/2, \ \ 0.5, 0.5], \\
\phi(s, c) &= [-\delta_o/2, 1, \ \ 1, -\delta_o/2],
\end{aligned}
$$

where δ_o is as defined in Section F.1. Hence, the feature expectations of the policies $\{\pi_a, \pi_b, \pi_c\}$ are respectively

$$
\begin{aligned}
\mu_a &= \frac{1}{2(1-\gamma)}[1, 1 \quad , \ -\delta_o, 2], \\
\mu_b &= \frac{1}{2(1-\gamma)}[2, -\delta_o, \ \ 1, 1], \\
\mu_c &= \frac{1}{2(1-\gamma)}[-\delta_o, 2, \ \ 2, -\delta_o].
\end{aligned}
$$

Consider two weight vectors $\mathbf{w}_a^\star = (v_o, v_o, 0, 0)$ and $\mathbf{w}_b^\star = (0, 0, v_o, v_o)$, where v_o is as defined in Section F.1. Since $\mathbf{w}_a^\star$ completely discards the last two coordinates, it immediately follows from the example of Section F.1 that $\mathcal{P}(\mathbf{w}_a^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Similarly, the same analysis on the last two coordinates shows that $\mathcal{P}(\mathbf{w}_b^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ as well. On the other hand, the optimal policy according to $\mathbf{w}_a^\star$ is π_a while the optimal policy according to $\mathbf{w}_b^\star$ is π_b. Hence, $\pi_a^\star \neq \pi_b^\star$, but we still have $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star)$, leading to non-identifiability. □

G Proof of Theorem 4.3

The proof of this theorem strongly relies on Lemma 4.2 and the example used to prove it. Consider the MDP as in Section F.2, but now with 6 features instead of just 4. In particular, let the feature expectations of the three policies be

$$
\begin{aligned}
\phi(s, a) &= [0.5, 0.5 \ \ , \ -\delta_o/2, 1, \ \ 1, -\delta_o/2], \\
\phi(s, b) &= [1, -\delta_o/2, \ \ 0.5, 0.5 \ \ , \ -\delta_o/2, 1], \\
\phi(s, c) &= [-\delta_o/2, 1, \ \ 1, -\delta_o/2, \ \ 0.5, 0.5 \ ].
\end{aligned}
$$

Hence, the feature expectations of the policies $\{\pi_a, \pi_b, \pi_c\}$ are respectively

$$
\mu_a = \frac{1}{2(1-\gamma)}[1, 1 \quad , \ -\delta_o, 2, \ \ 2, -\delta_o],
$$

$$\mu_b = \frac{1}{2(1-\gamma)}[2, -\delta_o, \ 1, 1 \quad , -\delta_o, 2],$$
$$\mu_c = \frac{1}{2(1-\gamma)}[-\delta_o, 2, \ 2, -\delta_o, \ 1, 1 \ ].$$

Consider three weight vectors

$$\mathbf{w}_a^\star = (v_o, v_o, 0, 0, 0, 0),$$
$$\mathbf{w}_b^\star = (0, 0, v_o, v_o, 0, 0),$$
$$\mathbf{w}_c^\star = (0, 0, 0, 0, v_o, v_o).$$

Since $\mathbf{w}_a^\star$ completely discards the last four coordinates, the example of Section F.1 shows that $\mathcal{P}(\mathbf{w}_a^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Similarly, the same analysis on the middle two and last two coordinates shows that $\mathcal{P}(\mathbf{w}_b^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ and $\mathcal{P}(\mathbf{w}_c^\star) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, respectively. However, the optimal policy according to $\mathbf{w}_a^\star$ is π_a, according to $\mathbf{w}_b^\star$ it is π_b, and according to $\mathbf{w}_c^\star$ it is π_c.

Now, consider an arbitrary algorithm $\mathcal{A}$, which takes as input a distribution over policies and outputs a (possibly randomized) policy. Look at the randomized policy $\mathcal{A}(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ returned by $\mathcal{A}$ when the input is $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, and let p_a, p_b, p_c be the probabilities it assigns to playing π_a, π_b and π_c. Let p_i (where $i \in \{a, b, c\}$) denote the smallest probability among the three. Then, $p_i \leq 1/3$. Pick the ground truth weight vector to be $\mathbf{w}_i^\star$. As $\mathcal{P}(\mathbf{w}_a^\star) = \mathcal{P}(\mathbf{w}_b^\star) = \mathcal{P}(\mathbf{w}_c^\star)$, the data generated by $\mathbf{w}_i^\star$ follows the distribution $\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$, and the policy distribution chosen by $\mathcal{A}$ is simply (p_a, p_b, p_c).

Now, with probability $p_i \leq 1/3$, the policy played is π_i leading to a reward of $\mathbf{w}_i^{\star\intercal}\mu_i = \frac{v_o}{(1-\gamma)}$, and with probability $(1-p_i)$, the policy played is some π_j (where $j \neq i$) leading to a reward of $\mathbf{w}_i^{\star\intercal}\mu_j = \frac{(2-\delta_o)}{2}\frac{v_o}{(1-\gamma)}$ (which is independent of the value of j).[10] Hence, the expected reward of algorithm $\mathcal{A}$ in this case is

$$p_i \cdot \frac{v_o}{(1-\gamma)} + (1-p_i) \cdot \frac{(2-\delta_o)}{2}\frac{v_o}{(1-\gamma)} = \frac{(2-\delta_o)}{2}\frac{v_o}{(1-\gamma)} + p_i \cdot \frac{\delta_o}{2}\frac{v_o}{(1-\gamma)}$$
$$\leq \frac{(2-\delta_o)v_o}{2(1-\gamma)} + \frac{\delta_o v_o}{6(1-\gamma)}.$$

Observe that the uniform mixture π^u in this case is just the input distribution $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Whatever be the chosen $\mathbf{w}_i^\star$, the expected reward of this distribution is exactly

$$\frac{1}{3} \cdot \frac{v_o}{(1-\gamma)} + \frac{2}{3} \cdot \frac{(2-\delta_o)}{2}\frac{v_o}{(1-\gamma)} = \frac{(2-\delta_o)v_o}{2(1-\gamma)} + \frac{\delta_o v_o}{6(1-\gamma)},$$

which is nothing but the upper bound on the expected reward of $\mathcal{A}$. Hence, for any algorithm $\mathcal{A}$ there exists a ground truth weight vector $\mathbf{w}_i^\star$ such that $\mathcal{A}$ has an expected reward at most that of π^u (which in turn is strictly suboptimal). □

H Proof of Theorem 5.1

To see that this problem is convex, let us analyze the distribution $\mathcal{Q}(\mathbf{w})$.

$$\begin{aligned}
\mathcal{Q}(\mathbf{w})_k &= \Pr(\text{Arm } k \text{ is optimal under weight } (\mathbf{w} + \boldsymbol{\eta})) \\
&= \Pr((\mathbf{w} + \boldsymbol{\eta})^\intercal \mathbf{x}_k \geq (\mathbf{w} + \boldsymbol{\eta})^\intercal \mathbf{x}_j \text{ for all j}) \\
&= \Pr((\mathbf{w} + \boldsymbol{\eta})^\intercal (\mathbf{x}_k - \mathbf{x}_j) \geq 0 \text{ for all j}) \\
&= \Pr(X_k(\mathbf{w} + \boldsymbol{\eta}) \geq 0) \\
&= \Pr(-X_k\boldsymbol{\eta} \leq X_k\mathbf{w}). \qquad (10)
\end{aligned}$$

Since $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 I_d)$, we have

$$-X_k\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 X_k X_k^\intercal).$$

And since $X_k X_k^\intercal$ is invertible, this distribution is non-degenerate and has a PDF. Let us use F_k to denote its CDF. Equation (10) then reduces to $\mathcal{Q}(\mathbf{w})_k = F_k(X_k\mathbf{w})$. Plugging this back into our optimization problem (1), we have

$$\min_{\mathbf{w}} -\sum_{k \in A} \tilde{\mathcal{Q}}_k \log F_k(X_k\mathbf{w}). \qquad (11)$$

As F_k corresponds to a (multivariate) Gaussian which has a log-concave PDF, this CDF is also log-concave. Hence, $\log F_k(X_k\mathbf{w})$ is concave in $\mathbf{w}$ for each k, and therefore (11) is a convex optimization problem. □

[10] An interesting point to note is that by carefully selecting v_o, one could get the corresponding δ_o to be arbitrarily large, thereby causing the optimal and suboptimal policies to have a much larger gap (equally affecting the uniform mixture π^u as well).

I Gradient Calculation

From Equation (11), we know that the objective function of problem (1) can be rewritten as $f(\mathbf{w}) = -\sum_{k\in A} \tilde{\mathcal{Q}}_k \log F_k(X_k\mathbf{w})$. Taking the gradient with respect to $\mathbf{w}$, we have

$$\begin{aligned}
\nabla_{\mathbf{w}} f(\mathbf{w}) &= -\sum_{k\in A} \tilde{\mathcal{Q}}_k \nabla_{\mathbf{w}} \log F_k(X_k\mathbf{w}) \\
&= -\sum_{k\in A} \frac{\tilde{\mathcal{Q}}_k}{F_k(X_k\mathbf{w})} \nabla_{\mathbf{w}} F_k(X_k\mathbf{w}) \\
&= -\sum_{k\in A} \frac{\tilde{\mathcal{Q}}_k}{F_k(X_k\mathbf{w})} \left[\sum_{i=1}^{m-1} \left. \frac{\partial F_k(\mathbf{z})}{\partial z_i} \right|_{z=X_k\mathbf{w}} \cdot \nabla_{\mathbf{w}} (X_k\mathbf{w})_i \right] \\
&= -\sum_{k\in A} \frac{\tilde{\mathcal{Q}}_k}{F_k(X_k\mathbf{w})} \left[\sum_{i=1}^{m-1} \left. \frac{\partial F_k(\mathbf{z})}{\partial z_i} \right|_{z=X_k\mathbf{w}} \cdot X_k^{(i)} \right], \qquad (12)
\end{aligned}$$

where the third equality holds as $F_k(\mathbf{z})$ has multidimensional input and we're taking the total derivative. Hence, we need to compute $\frac{\partial F_k(\mathbf{z})}{\partial z_i}$. Writing CDF F_k in terms of its PDF p_k (which exists as $X_k X_k^\intercal$ is invertible), we have

$$F_k(\mathbf{z}) = \int_{-\infty}^{z_1} \cdots \int_{-\infty}^{z_{m-1}} p_k(x_1, \ldots, x_{m-1}) dx_1 \ldots dx_{m-1}.$$

We compute partial derivative w.r.t. z_1 first, for simplicity, and generalize it after. In particular,

$$\begin{aligned}
\frac{\partial F_k(\mathbf{z})}{\partial z_1} &= \int_{-\infty}^{z_2} \cdots \int_{-\infty}^{z_{m-1}} \frac{\partial}{\partial z_1} \left[\int_{-\infty}^{z_1} p_k(x_1, \ldots, x_{m-1}) dx_1 \right] dx_2 \ldots dx_{m-1} \\
&= \int_{-\infty}^{z_2} \cdots \int_{-\infty}^{z_{m-1}} p_k(z_1, \ldots, x_{m-1}) dx_2 \ldots dx_{m-1} \\
&= \int_{-\infty}^{z_2} \cdots \int_{-\infty}^{z_{m-1}} p_{k,-1}(x_2, \ldots, x_{m-1}|z_1) p_{k,1}(z_1) dx_2 \ldots dx_{m-1} \\
&= p_{k,1}(z_1) \int_{-\infty}^{z_2} \cdots \int_{-\infty}^{z_{m-1}} p_{k,-1}(x_2, \ldots, x_{m-1}|z_1) dx_2 \ldots dx_{m-1} \\
&= p_{k,1}(z_1) \cdot \Pr_k(Z_2 \le z_2, \ldots, Z_{m-1} \le z_{m-1} | Z_1 = z_1) \\
&= p_{k,1}(z_1) \cdot F_{k,Z_{-1}|Z_1=z_1}(\mathbf{z}_{-1}),
\end{aligned}$$

where $F_{k,Z_{-1}|Z_1=z_1}$ is the conditional CDF of the distribution F_k given the first coordinate is z_1, $p_{k,1}$ is the marginal distribution PDF of this first coordinate, and $p_{k,-1}$ is the PDF of the rest. This derivation holds for the partial derivative w.r.t. any z_i, even though it was derived for z_1. Plugging this into Equation (12), the gradient therefore becomes

$$\nabla_{\mathbf{w}} f(\mathbf{w}) = -\sum_{k\in A} \frac{\tilde{\mathcal{Q}}_k}{F_k(X_k\mathbf{w})} \left[\sum_{i=1}^{m-1} p_{k,i}((X_k\mathbf{w})_i) \cdot F_{k,Z_{-i}|Z_i=(X_k\mathbf{w})_i}((X_k\mathbf{w})_{-i}) \cdot X_k^{(i)} \right].$$

Note that the conditional distribution $F_{k,Z_{-i}|Z_i=z_i}$ is also a Gaussian distribution with known parameters, and hence it can be estimated efficiently. We conclude that we can use gradient descent updates defined by

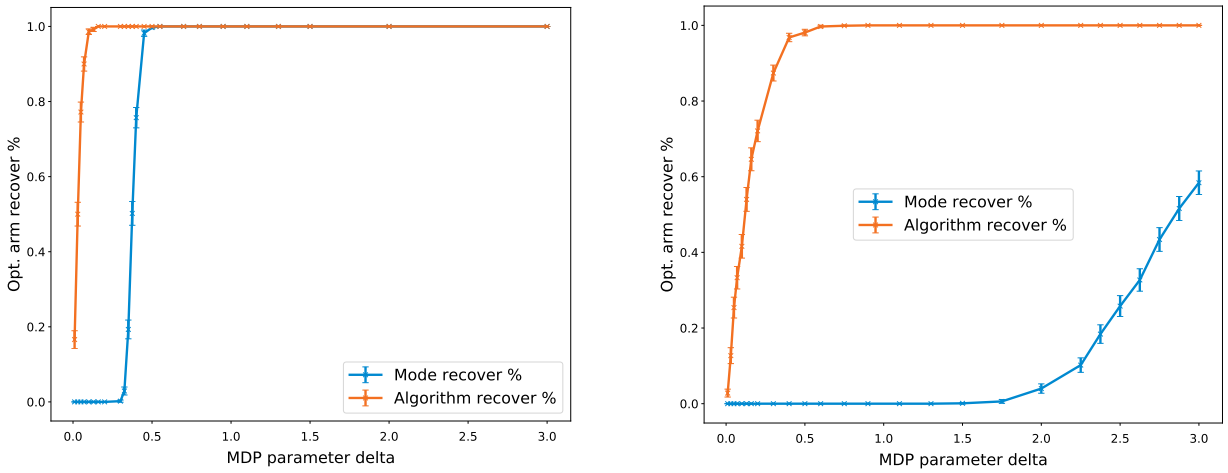
$$\mathbf{w}^+ = \mathbf{w} + \alpha \sum_{k\in A} \frac{\tilde{\mathcal{Q}}_k}{F_k(X_k\mathbf{w})} \left[\sum_{i=1}^{m-1} p_{k,i}((X_k\mathbf{w})_i) \cdot F_{k,Z_{-i}|Z_i=(X_k\mathbf{w})_i}((X_k\mathbf{w})_{-i}) \cdot X_k^{(i)} \right],$$

where α is a suitable step size, to find an optimal solution of (1).

J Additional Empirical Results for Inverse Bandits

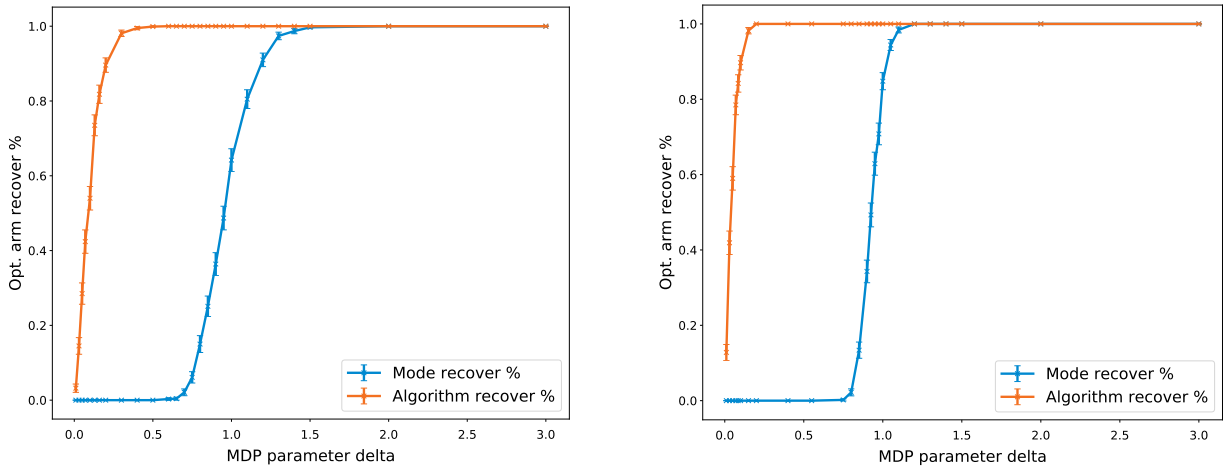
J.1 Varying parameter δ

Here, we present the experimental results as δ is varied for additional values of σ and n. All graphs in this section have also been averaged over 1000 runs, and error bars depict 95% confidence intervals. Figure 6 shows how the performance varies as δ is varied from 0.01 to 3, when σ is set to 0.5 and 2.0 (while n is still 500). As expected, one can observe that the tipping point (where the mode switches to the blue region corresponding to arm 1) occurs much earlier when $\sigma = 0.5$, and much later when $\sigma = 2$.



$\sigma = 0.5$ $\sigma = 2.0.$

Figure 6: Performance as δ is varied, when σ is fixed to 0.5 and 2.



$n = 250$ $n = 1000$

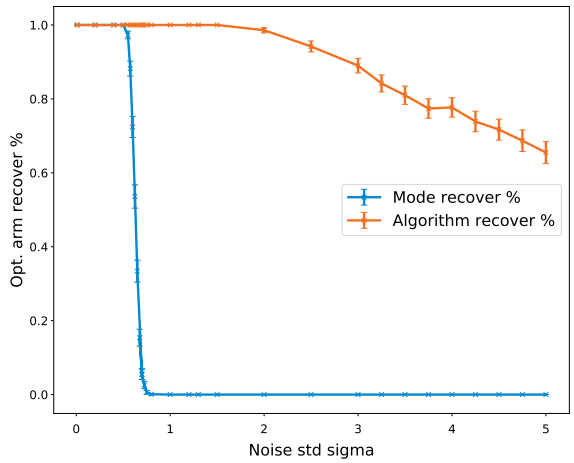
Figure 7: Performance as δ is varied, when the number of agents is 250 and 1000.

Figure 7 shows how the performance varies as δ is varied from 0.01 to 3, when the number of agents n is 250 and 1000 (while σ is still set to 1). First, note that the tipping point (for the mode switch) only depends on the value of δ and σ, and indeed, we can see from the graphs that the tipping point continues to be around $\delta = 1$ irrespective of the number of the agents. But, the number of agents defines how close $\tilde{\mathcal{Q}}$ is to $\mathcal{Q}(\mathbf{w}^\star)$, and hence determines the sharpness of the transition. In particular, for a larger number of agents, the empirical mode (obtained from $\tilde{\mathcal{Q}}$) is more likely to match the true mode (of $\mathcal{Q}(\mathbf{w}^\star)$). Hence, we can see that when $n = 1000$, the transition of the mode's performance is sharper across the tipping point (because of less noise), while when $n = 250$, the transition is smoother across this tipping point (because of more noise).

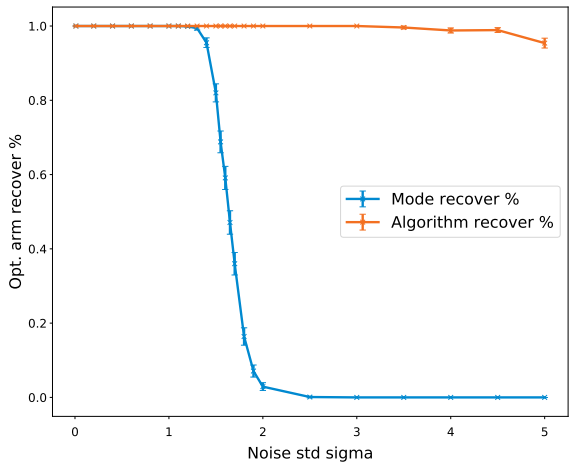
J.2 Varying noise parameter σ

Next, we present the experimental results as σ is varied, for additional values of δ and n. All graphs in this section have also been averaged over 1000 runs, and error bars depict 95% confidence intervals. Figure 8 shows how the performance varies as σ is varied from 0.01 to 5, when δ is set to 0.5 and 2.0 (while n is still 500). As expected, we can see that the tipping point (where the mode switches out of the blue region corresponding to arm 1) occurs earlier when $\delta = 0.5$, and much later when $\delta = 2$. Further, at high values of σ, the algorithm's performance is more robust when $\delta = 2$, as the blue region is larger.

Finally, Figure 9 shows how the performance varies as σ is varied from 0.01 to 5, when number of agents n is 250 and 1000 (while δ is still set to 1). Again, note that the tipping point of the mode switch occurs at the same point (around $\sigma = 1$) irrespective of the number of agents. And, as Section J.1, when $n = 1000$, the transition of the mode's performance is sharper

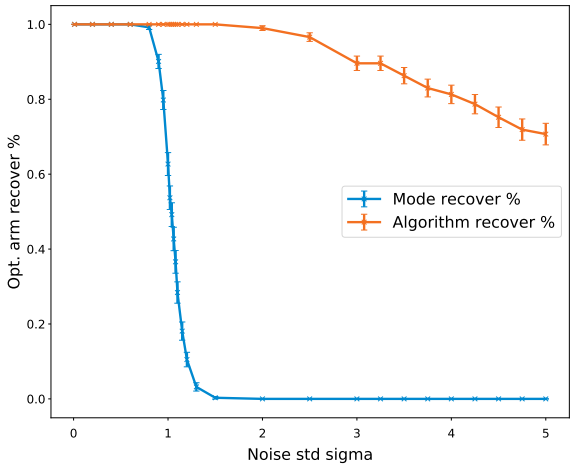


$\delta = 0.5$

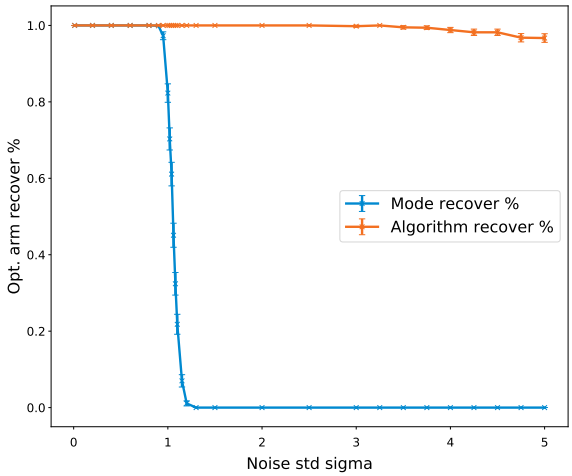


$\delta = 2.0.$

Figure 8: Performance as σ is varied, when δ is fixed to 0.5 and 2.



$n = 250$



$n = 1000$

Figure 9: Performance as σ is varied, when the number of agents is 250 and 1000.

across the tipping point, while when $n = 250$, the transition is smoother across it. Further, at high values of σ, $n = 1000$ has a much better algorithm performance compared to $n = 500$ (which in turn outperforms that at $n = 250$), showing that even at such high levels of noise, if $\tilde{\mathcal{Q}}$ coincides with $\mathcal{Q}(\mathbf{w}^\star)$, the algorithm is still able to recover the optimal arm 1.