A Few Expert Queries Suffices for Sample-Efficient RL with Resets and Linear Value Approximation

Philip Amortila[α*] **Nan Jiang**[α] **Dhruv Madeka**[β] **Dean P. Foster**[β]
[α]University of Illinois, Urbana-Champaign [β]Amazon, NYC

Abstract

The current paper studies sample-efficient Reinforcement Learning (RL) in settings where only the optimal value function is assumed to be linearly-realizable. It has recently been understood that, even under this seemingly strong assumption and access to a generative model, worst-case sample complexities can be prohibitively (i.e., exponentially) large. We investigate the setting where the learner additionally has access to interactive demonstrations from an expert policy, and we present a statistically and computationally efficient algorithm (DELPHI) for blending exploration with expert queries. In particular, DELPHI requires $\tilde{\mathcal{O}}(d)$ expert queries and a $\texttt{poly}(d, H, |\mathcal{A}|, 1/\varepsilon)$ amount of exploratory samples to provably recover an ε-suboptimal policy. Compared to pure RL approaches, this corresponds to an exponential improvement in sample complexity with surprisingly-little expert input. Compared to prior imitation learning (IL) approaches, our required number of expert demonstrations is independent of H and logarithmic in $1/\varepsilon$, whereas all prior work required at least linear factors of both in addition to the same dependence on d. Towards establishing the minimal amount of expert queries needed, we show that, in the same setting, any learner whose exploration budget is *polynomially-bounded* (in terms of d, H, and $|\mathcal{A}|$) will require *at least* $\tilde{\Omega}(\sqrt{d})$ oracle calls to recover a policy competing with the expert's value function. Under the weaker assumption that the expert's policy is linear (rather than their value function), we show that the lower bound increases to $\Omega(d)$.

1 Introduction

Many potential applications of reinforcement learning (RL) have intractably-large state spaces. Thus, we seek provably-correct methods which have statistical and computational requirements that are independent of the size of the state space. This requires some modelling assumptions. One dominating approach has been to introduce *function approximation*, and to posit that the MDP or its value functions are well-represented by the function approximation scheme which is employed. A basic starting point which still lacks comprehensive understanding is the case of linear value function approximation, which models value functions as lying in the span of a known d-dimensional feature mapping and asks for methods which have sample complexities that are polynomial only in d, H, and possibly $|\mathcal{A}|$ (H and $\mathcal{A}$ are the horizon and action sets of the MDP, respectively). This desideratum was recently understood to be impossible for the "minimal" case where only the optimal value function (or optimal action-value function) is assumed to be linear – i.e. there exist MDPs satisfying this assumption where the statistical complexity of any algorithm will be exponentially large, either in d or in H [WAS21; WSG21; WWK21; FKQR21]. Furthermore, this also holds in the case where the learner is equipped with a generative model (or simulator), allowing them to sample transitions from any state-action pair of the MDP. In recent years much has been said about linear value approximation under stronger assumptions, for example under determinism

*philipa4@illinois.edu

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

[WR13], linear/low-rank MDPs [JYWJ20; AJSWY20], Bellman-closedness [LSW20; ZLKB20], or the existence of a "core set" [SS20; ZLKB19]. These stronger assumptions can recover polynomial statistical complexities (thought not always computational ones), but are restrictive and oftentimes unrealistic.

In this work, we consider an alternative possibility for recovering polynomial complexities which does not further restrict the class of MDPs under consideration. We do this by assisting the learner with some additional side information about the problem. Specifically, we assume that there is a deterministic expert policy (which need not be the optimal policy) that the learner can query at any state, whereupon they will be informed of the expert's action at that state. Indeed, such information can often be made readily available if we have some form of prior knowledge (or human input) about the problem. Leveraging such expert demonstrations has been studied in interactive imitation learning (IL), with common applications in simulated domains [RGB11; Ros13; RB14; SVGBB17]. As we will see, however, the amount of expert queries required by a pure IL approach is significantly higher than what we need. Since interaction with a (human) expert might be costly, we wish to minimize the burden of the expert by having the learner explore mostly on their own, and only query the expert in a judicious manner. The question asked by this work, then, is:

> *Under linearity of the optimal value function, what is the minimal amount of expert data required for sample/computationally-efficient learning?*

Our main result is the DELPHI algorithm for exploring with an interactive expert. DELPHI assumes that the expert's value function is linear, and that the agent has access to a RESET function which lets them return to the state *most recently seen*. Under these conditions, our method uses surprisingly-few expert queries combined with some modest (polynomial) amount of exploration to recover the expert policy. Formally, DELPHI recovers a policy that is $\varepsilon-$optimal (with respect to the expert policy) with $\mathcal{O}(d\log(B/\varepsilon))$ oracle calls and $\tilde{\mathcal{O}}(\frac{d^2H^5|\mathcal{A}|B^4}{\varepsilon^4})$ exploratory samples, where B is a bound on the ℓ_2-norm of the unknown linear parameter.[2] Thus, our results show that merely $\tilde{\mathcal{O}}(d)$ expert queries enable an **exponential improvement** in sample complexity when compared to RL without expert advice. Furthermore, the number of oracle calls is completely independent of the horizon of the problem and is logarithmic in $1/\varepsilon$, whereas prior work in IL leveraging expert advice requires (at best) linear factors of both in addition to scaling with d. We also show that DELPHI is computationally efficient, that it is robust to optimization and misspecification errors, and that it can be extended to the case where the *action*-value function of the expert is linear when the MDP dynamics are deterministic.

Towards establishing the optimality of our algorithm, we study the capabilities of expert-augmented learners which have fixed exploration budgets. More specifically, we ask: what is the minimal number of expert queries *required* by any algorithm which is constrained to a *polynomially-bounded* exploration budget? We show that any polynomially-bounded learner (in terms of d, H, and $|\mathcal{A}|$) will require at least $\tilde{\Omega}(\sqrt{d})$ oracle calls to recover a policy competing with the expert's value function. In the weaker setting where only the expert's *policy* is linear, we show that this lower bound increases to $\Omega(d)$, matching our upper bound up to logarithmic factors.

The rest of the paper is structured as follows: in Section 2 we review background and present the problem setting. Section 3 describes our algorithm, its guarantee, a sketch of the proof, and discusses some extensions. Section 4 gives the lower bound on the number of expert queries needed. We conclude with an overview of related work and some discussion in Sections 5 and 6.

2 Background & Problem Setting

Notation We write $\mathrm{Dists}(\mathcal{X})$ for the set of probability measures on some set $\mathcal{X}$. We write $[N] := \{1, \ldots, N\}$. The *direct product* $\oplus$ corresponds to "concatenating" two vectors, i.e. for any two vectors $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^m$, we have $u \oplus v = (u_1, \ldots, u_n, v_1, \ldots, v_m)^\top \in \mathbb{R}^{n+m}$. We write $\otimes$ for the tensor (or outer) product of two vectors, defined by $u \otimes v = uv^\top \in \mathbb{R}^{n\times m}$, and $\flat(u \otimes v) \in \mathbb{R}^{n\cdot m}$ for the flattening (or vectorization) of said tensor product.

[2] The $\tilde{\mathcal{O}}$ notation ignores logarithmic factors.

MDPs, Policies, and Value Functions The typical environment in RL is modelled as an MDP [Put14; Sze10; SB18]. We consider here finite-horizon MDPs, which are specified by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, H, \mu_0)$, where $\mathcal{S}$ is a state space, $\mathcal{A} = [A]$ is a finite action set, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathrm{Dists}([0,1])$ is a (bounded) reward distribution function with expectation $r(s,a)$, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathrm{Dists}(\mathcal{S})$ is the transition distribution function with probability vectors $P(s,a) = [\mathcal{P}(s'|s,a)]_{s' \in \mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|}$, $H \in \mathbb{N}$ is the horizon, and $\mu_0 \in \mathrm{Dists}(\mathcal{S})$ is the starting distribution. Note that we have assumed that the action space is finite, although the state space may be infinite. Without loss of generality we assume that $\mathcal{S}$ is a disjoint union of per-horizon state spaces, i.e. $\mathcal{S} = \cup_{h \in [H]} \mathcal{S}_h$.

A (non-stationary) policy $\pi = (\pi_1, \cdots, \pi_H)$ prescribes a sequence of actions $\pi_h : \mathcal{S}_h \to \mathrm{Dists}(\mathcal{A})$, and its *value function* is $v^\pi(s) = \mathbb{E}[\sum_{h'=h}^{H} r(s_{h'}, a_{h'}) \mid s_h = s, a_{h'} \sim \pi_{h'}(s_{h'})]$, where $s \in \mathcal{S}_h$. The *action-value* function $q^\pi(s,a)$ is defined similarly, save that the first action taken is a and the proceeding actions follow π. Value functions satisfy the recursive relationship:

$$v^\pi(s) = r(s,\pi) + \langle P(s,\pi), v^\pi(\cdot) \rangle \coloneqq \mathcal{T}^\pi v^\pi(s) \tag{1}$$

where we have the shorthands $r(s,\pi) = \mathbb{E}_{a \sim \pi(s)}[r(s,a)]$, $P(s,\pi) = \mathbb{E}_{a \sim \pi(s)}[P(s,a)]$, and the Bellman operator $\mathcal{T}^\pi(\cdot) \coloneqq r(s,\pi) + \langle P(s,\pi), (\cdot) \rangle$. The Bellman operator has v^π as its unique fixed point. The optimal policy is written as $\pi^\star$, and its value function is denoted as $v^\star \coloneqq v^{\pi^\star}$. The objective is to find a π maximizing $v^\pi(\mu_0) \coloneqq \mathbb{E}_{s_0 \sim \mu_0}[v^\pi(s_0)]$.

Function Approximation With an Interactive Expert In the RL setting, the MDP is unknown and must be explored. As stated in the introduction, we seek sample complexities which are independent of the number of states. This is evidently not possible without further assumptions. To assist the learner, we assume that the agent has further access to an *oracle*, which, upon being queried, returns the action of an *expert* policy π° for the state. The expert policy need not be the optimal policy. We will assume that π° is deterministic, which is satisfied for example when $\pi^\circ = \pi^\star$ or when π° is the greedy policy of some value function.

Assumption 2.1 (Interactive expert)**.** *There is an oracle which can be queried at the current state s, which returns the action $\pi^\circ(s)$. Syntactically, the oracle is queried via the* ORACLE(s) *function.*

We note that the oracle can only be queried at the state the learner is currently visiting. The objective of the learner is to recover a policy which competes with the expert policy, namely a $\hat{\pi}$ such that:

$$v^{\hat{\pi}}(\mu_0) \geq v^\circ(\mu_0) - \varepsilon \quad \text{with probability } \geq 1 - \delta, \tag{2}$$

where $v^\circ \coloneqq v^{\pi^\circ}$ is the value function of the expert. In the sequel we refer to a policy satisfying Equation 2 as *ε-optimal*.

Our next assumption concerns the interplay of the expert policy with linear value approximation. Namely, we assume that the expert policy has a value function which is linear in a set of known features.

Assumption 2.2 (v°-linearity, with bounded features)**.** *The value function v° of the expert is linear with known features $\varphi \in \mathbb{R}^d$, i.e.*

$$v^\circ(s) = \langle \varphi(s), \theta^\circ \rangle, \ \forall s \in \mathcal{S}, \tag{3}$$

for some unknown $\theta^\circ \in \mathbb{R}^d$. We further assume that $\|\varphi(s)\|_2 \leq 1 \, \forall s$ and that $\|\theta^\circ\|_2 \leq B$ for some known $B \in \mathbb{R}^d$.

This assumption can be extended to the case of (slight) misspecification, as will be shown in Subsection 3.2. Our last assumption is that the agent has the ability to "reset" to the state most recently seen.

Assumption 2.3 (Resets)**.** *After experiencing a transition (s,a,r,s') in the MDP, the agent can return to the state s. Syntactically, this is done via the* RESET() *function.*

Our RESET assumption is weaker than the full generative model access [KMN02] or the "local" simulator setting [WAJAYJS21; LCCGW21; HLYAYS22] which has appeared in prior works. As noted in the introduction, existing exponential lower bounds entail that Assumptions 2.2 and 2.3 together are not enough to enable sample-efficient learning. Thus, any efficient learner which is provided with the above assumptions (2.1, 2.2, and 2.3) must necessarily make use of the expert.

3 The DELPHI algorithm

We are ready to describe our approach and give the main result. We begin by supposing that the starting distribution is deterministic (we will see later that this comes at no loss of generality).

Theorem 3.1. *Suppose Assumptions 2.1, 2.3, and 2.2 hold. Then the* DELPHI *algorithm will recover a policy $\hat{\pi}$ such that $v^{\hat{\pi}}(s_0) \geq v^\circ(s_0) - \varepsilon$ with probability $\geq 1-\delta$, using $\mathcal{O}(d\ln(B/\varepsilon))$ oracle calls and $\tilde{\mathcal{O}}(\frac{d^2H^5AB^4}{\varepsilon^4})$ interactions with the MDP. Furthermore this algorithm is computationally efficient.*

The pseudo-code for DELPHI is given in Algorithm 1, which uses Algorithm 2 (`measureTD`) as a sub-routine to measure expectations. DELPHI will solve for a parameter $\hat{\theta}$ such that an induced policy $\pi_{\hat{\theta}}$ (defined via Eq. (4)) will compete with the expert's value function. We note that deploying the policy $\pi_{\hat{\theta}}$ also requires the RESET functionality, since expectations must be estimated from a small number of samples at each state encountered. This is a consequence of our assumption that v° is linear (rather than, e.g., q° or the MDP itself), since selecting actions based on state-value functions will always require one-step look-aheads.

Intuition for DELPHI Recall that the expert policy π° satisfies $v^\circ = \mathcal{T}^{\pi^\circ} v^\circ$, and that this fixed point is unique. We say that a candidate value function is *consistent* at a state s if $v(s) = \mathcal{T}^{\pi^\circ} v(s)$. Note that we need consistency to hold globally (i.e. at all states) in order to ensure that $v = v^{\pi^\circ}$. Our methodology is based on ensuring that consistency on a small number of well-chosen states will guarantee global consistency.

DELPHI is inspired by a recent algorithm of [WAJAYJS21] called TensorPlan. As in TensorPlan, DELPHI proceeds via a "guess and check" procedure: at every iteration, we pick the *optimistic* linear parameter which is consistent on the past expert data that we have seen. We then check whether this choice of parameter is globally consistent, by playing $n_{\texttt{rollout}}$ rollouts of length H with a policy derived from the parameter. More specifically, for any θ the policy π_θ takes the form

$$\pi_\theta(s) = \operatorname{argmin}_a \left| \left(\hat{r}(s,a) + \hat{\mathbb{E}}_{s,a}[v_\theta(s')] \right) - v_\theta(s) \right|, \tag{4}$$

where $v_\theta(\cdot) \coloneqq \langle \varphi(\cdot), \theta \rangle$, and $\hat{r}(s,a)$ and $\hat{\mathbb{E}}_{s' \sim P(s,a)}$ are estimated via the RESET function.

After a certain number of rollouts, one of two things happen: either this policy encounters a state where there is no consistent action (i.e. the above minimum has a strictly positive value), or we only encounter states that are consistent.[3] In the first case, we query the oracle for its expert action and use the transition for that action to update the parameter set. In the second case, we derive (cf. Lemma 3.7) that if no inconsistencies are observed for several rollouts, then our "virtual value" v_θ is close to the true value under π_θ (i.e., v^{π_θ}). Using that θ was optimistic (i.e. $v_\theta(s_0) \geq v_{\theta^\circ}(s_0)$), this implies that we are optimal.

The only thing left to argue is that the number of iterations (i.e. the number of times that we can continue finding new parameters which are not globally consistent) is small. Using linearity of v°, it turns out roughly d inconsistencies are sufficient for this. To see this, note that we can re-write the Bellman equation for any $v_\theta(\cdot) = \langle \varphi(\cdot), \theta \rangle$ as:

$$\begin{aligned} v_\theta(s) = \mathcal{T}^{\pi^\circ} v_\theta(s) &\iff 0 = r(s, \pi^\circ(s)) + \langle \mathbb{E}_{s' \sim P(s,\pi^\circ)}[\varphi(s')] - \varphi(s), \theta \rangle \\ &\iff 0 = \langle \Delta_{s,\pi^\circ(s)}, 1 \oplus \theta \rangle, \end{aligned} \tag{5}$$

where we have introduced the notation $\Delta_{s,a} \coloneqq r(s,a) \oplus (\mathbb{E}[\varphi(s')] - \varphi(s))$ and used linearity of expectation, linearity of inner products, the definition of v_θ, and the definition of the direct product. We call the vector $\Delta_{s,a}$ the *temporal difference* (TD) vector for (s,a). Equation (5) is precisely an orthogonality constraint in $d+1$ dimensions. Since the parameter θ_t which is chosen at time t was consistent on past data, it is orthogonal to the previous $t-1$ TD vectors which have been generated from interactions with the oracle. If we happen to find a state which has no consistent action, then the TD vector corresponding to the expert action at that state must not be in the span of the previous expert TD vectors (otherwise it would be consistent). It follows that the iteration complexity is at most $d+1$, since there are at most $d+1$ linearly independent vectors in $\mathbb{R}^{d+1}$. We use the Eluder dimension [RVR13] to generalize this argument to the case where the expectations are estimated.

The next section solidifies the above intuition and sketches the proof more formally.

[3] In reality, due to sampling errors, we tolerate some error during the consistency checks.

Algorithm 1 DELPHI

1: **Inputs:** $s_0, \varphi, \texttt{sub-optimality}\ \varepsilon_{\texttt{target}}, \texttt{confidence}\ \delta, \texttt{parameter bound}\ B$
2: $\Theta_0 \leftarrow \texttt{Ball}_{\ell_2}(B)$ ▷ Θ_t : current consistent parameters
3: Initialize E_d, $n_{\texttt{eval}}$,$n_{\texttt{rollout}}$, and $\varepsilon_{\texttt{tol}}$ via Equations (9), (12), (10), (15)
4: **for** $t = 1$ to $E_d + 1$ **do**
5: Pick $\theta_t \in \operatorname{argmax}_{\theta \in \Theta_{t-1}} \big(v_\theta(s_0) := \theta^\top \varphi(s_0)\big)$ ▷ Optimistic choice over Θ_{t-1}
6: $\texttt{consistent} \leftarrow \texttt{true}$
7: **for** $m = 1$ to $n_{\texttt{rollout}}$ **do** ▷ $n_{\texttt{rollout}}$ number of rollouts with θ_t-induced policy
8: $S_{t,m,h} = s_0$ ▷ Initialize rollout
9: **for** $h = 1$ to H **do**
10: **for** $a \in [A]$ **do** ▷ For each action
11: $\hat{\Delta}_{S_{t,m,h},a} \leftarrow \texttt{measureTD}(S_{t,m,h}, a, n_{\texttt{eval}})$ ▷ Measure TD vector at (s, a)
12: **end for**
13: **if** $\min_a \left|\langle \hat{\Delta}_{S_{t,m,h},a}, 1 \oplus \theta_t \rangle\right| > \varepsilon_{\texttt{tol}}$ **then** ▷ No consistent action
14: $\texttt{consistent} \leftarrow \texttt{false}$
15: $a_t^\circ \leftarrow \textsc{ORACLE}(S_{t,m,h})$ ▷ Query oracle for $\pi^\circ(S_{t,m,h})$
16: $\tilde{\Delta}_{S_{t,m,h},a_t^\circ} \leftarrow \texttt{measureTD}(S_{t,m,h}, a_t^\circ, 4E_d n_{\texttt{eval}})$ ▷ Refined data
17: $\Theta_t \leftarrow \Theta_{t-1} \cap \{\theta \mid |\langle \tilde{\Delta}_{S_{t,m,h},a_t^\circ}, 1 \oplus \theta \rangle| \le \varepsilon_{\texttt{tol}}\}$ ▷ New admissible parameters
18: Exit current iteration, $t \leftarrow t + 1$, Goto Line 5.
19: **end if**
20: $A_{t,m,h} \leftarrow \operatorname{argmin}_{a \in [A]} \left|\langle \hat{\Delta}_{S_{t,m,h},a}, 1 \oplus \theta_t \rangle\right|$ ▷ Else consistent, keep playing
21: Play $A_{t,m,h}$, get $R_{t,m,h}, S_{t,m,h+1} \sim \texttt{MDP}$ ▷ Roll forward
22: **end for**
23: **end for**
24: **if** $\texttt{consistent} == \texttt{true}$ **then**
25: **return** θ_t ▷ No inconsistency for m rollouts $\implies$ success
26: **end if**
27: **end for**
28: **return** θ_{E_d+1}

Algorithm 2 `measureTD`

1: **Inputs:** $s, a, \varphi(\cdot), n, \textsc{RESET}()$
2: **for** $i = 1$ to n **do**
3: Play action a at s, receive sample R_l and S_l' from MDP
4: $\Delta_i \leftarrow R_l \oplus (\varphi(S_l') - \varphi(s))$
5: $\textsc{RESET}()$
6: **end for**
7: **return** $\hat{\Delta}_{s,a} := \frac{1}{n} \sum_{i \in [n]} \Delta_i$

3.1 Proof sketch

The full proof comes in 4 parts. The proofs for all Lemmas and the precise values of all hyperparameters are given in Appendix A.

1. Lemmas 3.2 and 3.3 gives concentration bounds which establish that, with high probability, the measurements $\hat{\Delta}$ (Line 11) and $\tilde{\Delta}$ (Line 16) concentrate to the average $\Delta_{s,a} = \mathbb{E}_{R(s,a)} r \oplus \big(\mathbb{E}_{P(s,a)} \varphi(s') - \varphi(s)\big)$.
2. Lemma 3.4 establishes that, with high probability, the true optimal parameter θ° is not eliminated from Θ_t for any parameter set that is encountered. It follows (Lemma 3.5) by optimism that $v_{\theta_t}(s_0) \ge v^\circ(s_0)$ with high probability, where θ_t is the parameter chosen at time t.
3. Lemma 3.6 establishes an iteration bound: the algorithm will terminate after at most $t = E_d$ iterations of the outermost loop (and thus after at most E_d oracle queries). The quantity E_d happens to be the Eluder dimension of our linear function class.

4. Lastly, Lemma 3.7 establishes that if $n_{\texttt{rollout}}$ number of rollouts occur without observing a consistency break, then the virtual value ($v_\theta(s_0)$) must be roughly equal to the true value under the executed policy ($v^{\pi_\theta}(s_0)$). Theorem 3.1 combines all the ingredients to conclude the proof.

Part 1: Concentration bounds Recall that $\Delta_{s,a} \coloneqq r(s,a) \oplus (\mathbb{E}[\varphi(s')] - \varphi(s))$ is the true TD vector, $\hat{\Delta}$ is the estimated TD vectors obtained with $n_{\texttt{eval}}$ samples (in Line 11), and $\tilde{\Delta}$ is the "refined data" obtained with $4E_d n_{\texttt{eval}}$ samples (in Line 16). The following lemmas establish concentration of $\hat{\Delta}$ and $\tilde{\Delta}$ to the true TD vector.

Lemma 3.2 (Concentration of $\hat{\Delta}_{s,a}$ (Line 11))**.** *For any $s, a \in \mathcal{S} \times \mathcal{A}$ that is observed throughout the execution* DELPHI *, with $n_{\texttt{eval}}$ samples in Line 11, we have that with probability $\geq 1-\delta$, $\left\|\hat{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}$ and thus that $\langle 1 \oplus \theta, \hat{\Delta}_{s,a} - \Delta_{s,a}\rangle \leq \bar{\varepsilon}_{\texttt{eval}}$.*

Lemma 3.3 ($\tilde{\Delta}_{s,a}$ concentrates even more (Line 16))**.** *Similarly, for all s, a where we call the oracle, with probability $1-\delta$, we have $\left\|\tilde{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}/(2\sqrt{E_d})$, and thus, $\forall \theta \in \texttt{Ball}_{\ell_2}(B)$, $|\langle 1 \oplus \theta, \tilde{\Delta}_{s,a} - \Delta_{s,a}\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/(2\sqrt{E_d})$.*

Part 2: Optimism This part shows that (with high probability) the true optimal parameter is not eliminated from the version space, and thus by optimism that the predicted value v_t upper bounds v°.

Lemma 3.4 (θ° not eliminated)**.** *With probability $\geq 1-\delta$, $\theta^\circ \in \Theta_t$ for all iterations $t \in [E_d+1]$.*

Lemma 3.5 (Optimism)**.** *Under the event of Lemma 3.4, we have $v_t(s_0) \geq v^\circ(s_0)$, $\forall t \in [E_d]$.*

Part 3: Iteration bound To bound the iteration complexity of our algorithm, we use the notion of *Eluder* dimension [RVR13]. Loosely, the Eluder dimension with respect to some target function is the longest sequence of points (x_i) such that there exists functions differing from the target function on x_i but which correctly fit it on $x_1, \ldots, x_{i-1}$. A formal definition is provided in Appendix A. We will use the result that the Eluder dimension of linear functions is $\mathcal{O}(d\ln(B/\varepsilon))$ [RVR13; LKFS21].

Lemma 3.6 (Iteration Complexity)**.** *With probability $\geq 1-2\delta$, the iteration complexity of the algorithm is at most the Eluder dimension at scale $\bar{\varepsilon}_{\texttt{eval}}$, i.e. $E_d = \mathcal{O}(d\ln(B/\bar{\varepsilon}_{\texttt{eval}}))$.*

Part 4: Consistency, and putting everything together The last thing to show is that if we are consistent at all states for several rollouts then the virtual value v_θ will be close to the value of the v^{π_θ}.

Lemma 3.7 (Consistency $\implies$ accurate prediction)**.** *If m rollouts have occured without any inconsistencies (i.e., the if statement of Line 13 never gets triggered), then $v^{\pi_\theta}(s_0) > v_\theta(s_0) - 5H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}}$ with probability $\geq 1-3\delta$.*

We are now ready to put everything together.

Proof (of Theorem 3.1). Assume all events introduced so far (i.e. the events in Lemma 3.2, Lemma 3.3, and Lemma 3.7). Together these happen with probability $\geq 1-3\delta$, so we can re-define $\delta \mapsto \delta/3$ such that the events happen together with probability $\geq 1-\delta$ (this only increases factors inside logarithms by 3). By Lemma 3.7, we have:

$$\begin{aligned} v^{\pi_\theta}(s_0) &\geq v_\theta(s_0) - 5H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}} \\ &\geq v_\theta(s_0) - \varepsilon_{\texttt{target}} \\ &\geq v^{\pi^\circ}(s_0) - \varepsilon_{\texttt{target}}, \end{aligned} \tag{6}$$

where the second step follows from plugging in the definitions of $n_{\texttt{eval}}$ (Eq. (12)), $n_{\texttt{rollout}}$ (Eq. (10)), $\bar{\varepsilon}_{\texttt{eval}}$, (Eq. (13)) and $\varepsilon_{\texttt{roll}}$ (Eq. (16)), and the final step follows by optimism (Lemma 3.5). The total sample complexity of our algorithm is: $E_d = \tilde{\mathcal{O}}(d)$ oracle calls, and $Nn_{\texttt{eval}} = (E_d+1)Hn_{\texttt{rollout}}An_{\texttt{eval}} = \tilde{\mathcal{O}}(\frac{d^2H^5AB^4}{\varepsilon^4})$ exploration cost. The last claim to verify is that of computational efficiency. We note that the only computationally intensive step from iteration is Line 5, i.e. the optimization problem corresponding to the optimistic choice over the parameter set $\max_{\theta \in \Theta_{t-1}} \theta^\top \varphi(s_0)$. This is readily seen to be a convex program, since the objective is a linear

function and the constraint set is a convex set in $\mathbb{R}^d$ (it is initialized at $\Theta_0 = \texttt{Ball}_{\ell_2}(B)$ and every update intersects it with half-spaces (Line 17)). Besides the ℓ_2-norm constraint, all other constraints can be represented with a linear program, since each absolute value constraint can be split into two inequalities. By Lemma 3.6 the number of constraints will be at most $2E_d = \tilde{\mathcal{O}}(d)$ (and these can easily be converted to standard form). Thus, there are a plethora of polynomial-time methods for solving this convex program [BBV04; Bub+15].[4] □

3.2 Extensions

In this section, we show that DELPHI can be extended to work with optimization errors, under stochastic starting distributions, with misspecification, and with linear q° whenever dynamics are deterministic.

Optimization errors and stochastic start state If the optimistic program can only be solved upto accuracy $\varepsilon_{\texttt{opt}} \leq c\varepsilon_{\texttt{target}}$ (for $c \in [0, 1)$) at each iteration, then Lemma 3.5 becomes $v_t(s_0) \geq v^\circ(s_0) - \varepsilon_{\texttt{opt}}$ and this only appears in Eq. (6). We then need $5H\bar{\varepsilon}_{\texttt{eval}} + \varepsilon_{\texttt{roll}} \leq (1-c)\varepsilon_{\texttt{target}}$, which is achieved by increasing $n_{\texttt{rollout}}$ and $n_{\texttt{eval}}$ by a factor of $1/(1-c)^2$, which increases the final exploration cost by $1/(1-c)^4$. For stochastic starting states, we simply work with $v_\theta(\mu_0) = \mathbb{E}_{s_0\sim\mu_0}[v_\theta(s_0)]$ (resp. $v^\circ(\mu_0)$) wherever $v_\theta(s_0)$ (resp. $v^\circ(s_0)$) previously appeared. The "starting feature" $\mathbb{E}_{s_0\sim\mu_0}[\varphi(s_0)]$ must be estimated from samples, which is then used for the optimistic program in Line 5 with $\varphi(s_0)$ replaced by this expectation. The error is easily bounded as before by Hoeffding's inequality, and will simply propagate additively through the proof.

Misspecified value functions and innacurate simulators DELPHI inherits some robustness properties from TensorPlan. Namely, with a constant increase in exploration cost, DELPHI continues to work under errors in the modelling assumptions. The first case is where the expert value function is not linear but rather is approximately linear up to some uniform error. Formally, we say that the MDP is η-*misspecified* for the expert policy π° and the feature map φ if there exists θ° such that $\sup_s |v^\circ(s) - \langle\varphi(s), \theta^\circ\rangle| \leq \eta$. The second case is where the simulator itself is flawed. Formally, we say that the simulator is λ-*innacurate* if a transition (r, s') from any state-action pair (s, a) of the MDP is instead observed as $(\Pi(r + \lambda_{s,a}), s')$, where Π is the projection onto $[0, 1]$ and $\lambda_{s,a}$ is a constant uniformly bounded by λ. The following result (proved in Appendix A.5) states that DELPHI can tolerate misspecification or simulator inaccuracies of order roughly $\frac{\bar{\varepsilon}_{\texttt{eval}}}{\sqrt{E_d}} = \mathcal{O}(\frac{1}{H\sqrt{d}})$.

Theorem 3.8 (DELPHI with misspecification)**.** *Redefine $n'_{\texttt{eval}} = 4n_{\texttt{eval}}$ (Eq. 12) and all subsequent hyperparameters which depend on $n'_{\texttt{eval}}$. Then we have that, for all MDPs that are at most $\frac{\bar{\varepsilon}_{\texttt{eval}}}{8\sqrt{E_d}}$-misspecified or for all simulators that are at most $\frac{\bar{\varepsilon}_{\texttt{eval}}}{4\sqrt{E_d}}$-innaccurate, the conclusions of Theorem 3.1 continue to hold when running* DELPHI *with the new hyperparameters.*

q°-linearity, in deterministic dynamics Rather than working with the Bellman equation $v_\theta(s) = \mathcal{T}^{\pi^\circ} v_\theta(s)$ we work with $q_\theta(s, a) = \mathcal{T}^{\pi^\circ} q_\theta(s, a)$, which can be linearized similarly to Eq. (5). Namely:

$$q_\theta(s, a) = \mathcal{T}^{\pi^\circ} q_\theta(s, a) \iff 0 = \langle r(s, a) \oplus (\mathbb{E}[\varphi(s', \pi^\circ(s))] - \varphi(s, a)), 1 \oplus \theta\rangle \tag{7}$$

This derivation holds generally, although to be able to speak of consistency at (s, a) with respect to a next action a', we now assume deterministic dynamics, so that the above becomes

$$0 = \langle r(s, a) \oplus (\varphi(s', a') - \varphi(s, a)), 1 \oplus \theta\rangle,$$

where s' is the unique successor of (s, a) and a' is the action that we are checking consistency for. The algorithm proceeds as before, except rather than checking all actions at a given state (Line 13), we check all proceeding actions a' against the current (s, a), and play the one with the smallest TD error (analogously to Line 20 of Algorithm 1).

[4] In fact, replacing Θ_0 with an ℓ_∞ constraint will only incur a d^2 factor to the exploration cost and log factors to the oracle cost (since we can think of $B \mapsto \sqrt{d}B$), but Line 5 will then be a fully linear program.

4 How many oracle calls are necessary?

Is DELPHI optimal in terms of its number of oracle calls? To answer this question, we must argue that no algorithm can find an $\varepsilon-$optimal solution with less than $\Omega(d)$ expert queries. As stated, we are competing with agents which can (for example) exhaustively search the state-space and do not refer to the expert at all. Thus, an *exploration budget* must be imposed (formally, a maximal amount of allowed interaction with the MDP, excluding oracle calls). To make matters more interesting, we set this cap to be *any* polynomial amount, resulting in the following question:

> *Under Assumptions 2.1, 2.2, and 2.3, what is the minimal amount of expert queries needed for any algorithm with a poly$(d, H, A, \frac{1}{\varepsilon})$ exploration budget to find an $\varepsilon-$optimal solution?*

We firsly note that, measured in the worst-case, this minimal amount of expert queries is strictly positive since there exists MDPs satisfying $v^\star$-linearity for which no algorithm can return a sound solution with poly$(d, H, A, \frac{1}{\varepsilon})$ queries [WAS21; WSG21]. Secondly, while it was possible to restrict ourselves simply to learners which have the same exploration requirements as DELPHI , we opted to study algorithms with arbitrary polynomial exploration budgets, since it is a more fundamental question about the limits of exploration and the benefits of expert advice.

Note that any solution to this question must, a priori, have an exponential sample complexity for pure RL (otherwise the agent does not need to resort to the expert). Interestingly, most constructions which exhibit exponential lower bounds for linearly-realizable RL can be solved with a single query from the oracle (e.g., [WAS21; WWK21]). These constructions rely on having an exponentially large action set with a single correct action that effectively solves the MDP. Our main lower bound comes from extending the recent lower bound of [WSG21], which is also the only known construction for an exponential lower bound which has a polynomial action set (rather than exponential). Our result is that at least $\tilde{\Omega}(\sqrt{d})$ oracle calls are necessary:

Theorem 4.1. *There exists a family of MDPs satisfying 2.1, 2.2, and 2.3, such that any algorithm with poly(d, H, A) exploration budget will need at least $\tilde{\Omega}(\sqrt{d})$ oracle calls to recover a policy such that $v^{\hat{\pi}}(s_0) \geq v^\circ(s_0) - 0.01$.*

Our second lower bound considers an alternative assumption, which instead posits that the expert *policy* is linear. Formally, this assumption says:

Assumption 4.2 (π°-linearity, with bounded features)**.** *The policy π° of the expert is linear with known features $\varphi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, i.e.*

$$\pi^\circ(s) \in \operatorname{argmax}_a \langle \varphi(s, a), \theta^\circ \rangle, \ \forall s \in \mathcal{S}, \tag{8}$$

for some unknown $\theta^\circ \in \mathbb{R}^d \setminus \{0\}$.[5] We further assume that $\|\varphi(s)\|_2 \leq 1 \, \forall s$ and that $\|\theta^\circ\|_2 \leq B$ for some known $B \in \mathbb{R}^d$.

When $\pi^\circ = \pi^\star$, it is easy to see that this assumption is implied by the assumption that $q^\star$ is linearly-realizable. In general however, q°-linearity does not imply that π° is linear (see Appendix C for an example), although it does imply that the greedy policy derived from q° is linear. We give a lower bound for this case which matches the upper bound of DELPHI up to logarithmic factors.

Theorem 4.3. *There exists a family of MDPs satisfying assumptions 2.1, 4.2, and 2.3 such that any algorithm with poly(d, H, A) exploration budget will need at least $\Omega(d)$ oracle calls to recover a policy such that $v^{\hat{\pi}}(s_0) \geq v^\circ(s_0) - 0.01$.*

Intuition for the lower bound Theorems 4.1 and 4.3 use the same MDP construction but with different features. We give some intuition for the MDP construction which is used, but due to its intricacy a full description (and the information-theoretic proof) are deferred to Appendix B. Loosely, the learner has to find a hidden hypercube vector $s^\star \in \{\pm 1\}^p$. The action space is $\mathcal{A} = [p]$, and each action corresponds to flipping one of the bits of the current vector. The MDP has K "phases" which each correspond to p bit flips (thus $H \approx Kp$). A linear reward is given only if a sufficiently small neighborhood of $s^\star$ is reached, and the reward (thus the value) will decay geometrically in each subsequent phase that the neighborhood is not reached. Intuitively, the oracle needs to be used

[5] We exclude 0 since otherwise this would simply be the class of all policies.

$\approx p$ times, since each oracle calls only reveals one action, and thus one bit of the optimal vector $s^\star$. The reason that this results in a $\tilde{\Omega}(\sqrt{d})$ lower bound (rather than $\tilde{\Omega}(d)$) is that the value function will experience a *scale transition* when going from states where $s^\star$ is reachable given the remaining steps in the current phase to states where $s^\star$ is no longer reachable. As just described, the value (and thus the features) will be one order of magnitude smaller in this latter portion of the state space. As this would betray the location of the secret parameter, the value function is instead augmented to be quadratic in p (roughly, the product of the distance achieved at the end of this phase and that of the next phase), which requires that $p \approx \sqrt{d}$ in order to observe linearity. On the other hand, the lower bound for π°-linearity (Theorem 4.3) can remain linear in d, since the definition (Eq. 8) is *scale-insensitive*.

Closing the gap between our upper bound of $\tilde{\mathcal{O}}(d)$ and our lower bound of $\tilde{\Omega}(\sqrt{d})$ remains a challenging but interesting question. In finite horizons, with access to a generative model or a RESET method, we suspect that the only mechanism for creating a hard MDP is by geometrically decaying the maximum possible value for each stage, such that at the final stage of the MDP the (random) reward becomes exponentially small in H (that is the approach taken here and in [WAS21; WSG21]).[6] If the geometric decaying happens in "phases" then this implies that $\Omega(1)$ of the value is located in the first "phase". The tension, then, is to have such a construction while hiding this large value, and forcing the learner to rely on several oracle calls to find it. Prior constructions [WAS21; WWK21] have hid the large initial value by choosing an exponentially large action set, although as discussed above these examples are solved with a single query of the expert oracle. Extending the "needle in a haystack" to occur over multiple decisions (cf. the "phases" used above) leads to increased oracle requirements, although due to the scale transition phenomenon observed above it is far from clear how to do this with phases of length $\approx d$. The question thus boils down to: is geometric value reduction *necessary* for exponential lower bounds in this setting, and if so can we avoid the scale transition problem? The situation is likely to be different in the *online RL* setting. For example, the construction of [WWK21] extends [WAS21] to the online setting, and does not need to decay rewards but instead adds an $\Omega(1)$ probability of death at every transition. This mechanism evidently does not work when the agent has resets, and since it is not known whether DELPHI can be extended to the online setting we opted to keep the settings consistent between our upper bound and our lower bound.

5 Related works

The closest body of work to our setting is the field of *interactive* IL. As in our setting, interactive IL considers the case where the learner has access to an expert oracle that can be queried adaptively. It differs from our setting, however, since traditionally in IL the learner does not observe reward information. We further differ from the IL setting since we consider value function approximation rather than general policy classes, and since we assume access to a RESET function. Despite that many demonstrations of interactive IL occur in simulated domains [RGB11; Ros13; RB14], the benefits of this feature have not previously been studied. Our assumption of v° linearity entails that many IL methods are not directly applicable. Indeed, the policy π° itself does not need to be linear (despite that v° is), so it is unclear which policy class to use for those algorithms. Assuming for the sake of comparison that linear policies can be used, IL methods would still obtain worse oracle rates. Indeed, using results from [AJKS19], Behaviour Cloning (for the passive case) or AggreVaTe [RB14; SVGBB17] (for the interactive case) have worst-case errors $\mathcal{O}(\frac{1}{(1-\gamma)^2}\sqrt{\frac{d\ln(1/\delta)}{N}})$ for discounted MDPs, which roughly translates to an oracle complexity of $N = \mathcal{O}(dH^4/\varepsilon^2)$ when using the standard reduction $H \mapsto (1-\gamma)^{-1}$. This is in sharp contrast to our $\tilde{\mathcal{O}}(d)$ oracle calls, which in independent of H and logarithmic in $1/\varepsilon$, and demonstrates the improvement due to exploration with the help of value-function approximation. Beyond these approaches, another intuitive method would be to perform regression by doing a Monte Carlo estimation for the value of $v^\circ(s)$ for each s along a certain "good" set of features. This would require collecting rollouts from those states, which will again introduce a factor of H in the number of oracle queries. Our algorithm instead finds a set of state-action pairs where the Temporal difference (TD) errors (which we represent as vectors) span orthogonal directions. These can be estimated with a single transition, and this "local fitting" approach is novel to the IL literature and avoids the factors of H and $1/\varepsilon$ from previous works.

[6] In particular, an exponentially small gap is necessary, since backwards induction-type methods are possible when $q^\star$ is linearly-realizable and have sample complexities scaling with the inverse gap [DKWY19; DLMW20].

In terms of other linear structure in IL, the works of [AN04; SS07] assume a known transition function and unknown linear rewards, and derive expert complexities of $\mathcal{O}\left(\frac{dH^2 \log(dH/\varepsilon)}{\varepsilon^2}\right)$ and $\mathcal{O}(\frac{H^2 \log(d)}{\varepsilon^2})$ respectively, but the algorithms involve (tabular) planning in MDPs and thus are not computationally efficient. Most relevant is the recent work of [RHYLJR21], which, in the reward-free case, assumes that the expert policy is linear (i.e., our Assumption 4.2). A sample complexity of $\tilde{\mathcal{O}}(dH/\varepsilon)$ is shown for Behaviour Cloning in this case, though no lower bound is given.

On the technical side, the DELPHI algorithm is inspired by a recent algorithm called TensorPlan [WAJAYJS21]. TensorPlan works for pure RL under Assumptions 2.2 and 2.3 but has a sample complexity scaling as $\text{poly}((\frac{dH}{\varepsilon})^A, B)$ and is computationally intractable. Our extension of TensorPlan naturally incorporates the expert demonstrations, while simultaneously (1) having low oracle requirements, (2) addressing the exponential sample complexity of TensorPlan, and (3) rendering the algorithm computationally efficient. Our approach is based only on finding value functions which satisfy the Bellman equation. Bellman error minimization approaches have appeared in other works (e.g. [JLM21; ZLKB20; CJ19]), but have always required a restrictive "Bellman closedness" assumption. As discussed, our lower bound construction is an extension of the recent remarkable lower bound of [WSG21], although several aspects of the construction have been modified to obtain better rates. In particular, we modified the reward/value functions, the feature mappings, and introduced an expert policy which differs from the optimal policy. For the proof, our setting is more complex as the learner has adaptive access to a second information source (the oracle), and a more sophisticated information-theoretic argument was needed to show that the oracle does not leak too much information.

6 Conclusion

We presented the DELPHI algorithm for RL with an interactive expert. We saw that, with $\tilde{\mathcal{O}}(d)$ oracle calls, exponential improvements in sample complexity are possible for RL with linearly-realizable optimal value functions. Compared to prior works on learning with an interactive expert, we also saw that DELPHI 's oracle requirements were smaller, and in fact are independent of the horizon of the MDP. It would be interesting and fruitful to resolve the gap between the oracle complexity required by DELPHI and the one obtained from our lower bound (either answer would be surprising to the authors). It would also be fruitful to study the case of linearly-realizable *action-value* functions in stochastic MDPs, which would potentially enable our method to be extended to the online setting.

Acknowledgments and Disclosure of Funding

PA gratefully acknowledges funding from the Natural Sciences and Engineering Research Council of Canada (NSERC). Work done in part while PA was an intern at Amazon.

References

[AJKS19] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. "Reinforcement learning: Theory and algorithms". In: *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep* (2019).

[AJSWY20] Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. "Model-based reinforcement learning with value-targeted regression". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 463–474.

[AN04] Pieter Abbeel and Andrew Y Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 1.

[BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Bub+15] Sébastien Bubeck et al. "Convex optimization: Algorithms and complexity". In: *Foundations and Trends® in Machine Learning* 8.3-4 (2015), pp. 231–357.

[CJ19] Jinglin Chen and Nan Jiang. "Information-theoretic considerations in batch reinforcement learning". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 1042–1051.

[DKWY19] Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. "Is a good representation sufficient for sample efficient reinforcement learning?" In: *arXiv preprint arXiv:1910.03016* (2019).

[DLMW20] Simon S Du, Jason D Lee, Gaurav Mahajan, and Ruosong Wang. "Agnostic Q-learning with function approximation in deterministic systems: Tight bounds on approximation error and sample complexity". In: *arXiv preprint arXiv:2002.07125* (2020).

[FKQR21] Dylan J Foster, Sham M Kakade, Jian Qian, and Alexander Rakhlin. "The Statistical Complexity of Interactive Decision Making". In: *arXiv preprint arXiv:2112.13487* (2021).

[HLYAYS22] Botao Hao, Nevena Lazic, Dong Yin, Yasin Abbasi-Yadkori, and Csaba Szepesvari. "Confident Least Square Value Iteration with Local Access to a Simulator". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2022, pp. 2420–2435.

[JLM21] Chi Jin, Qinghua Liu, and Sobhan Miryoosefi. "Bellman eluder dimension: New rich classes of rl problems, and sample-efficient algorithms". In: *Advances in Neural Information Processing Systems* 34 (2021).

[JYWJ20] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. "Provably efficient reinforcement learning with linear function approximation". In: *Conference on Learning Theory*. 2020, pp. 2137–2143.

[KMN02] Michael Kearns, Yishay Mansour, and Andrew Y Ng. "A sparse sampling algorithm for near-optimal planning in large Markov decision processes". In: *Machine learning* 49.2 (2002), pp. 193–208.

[LCCGW21] Gen Li, Yuxin Chen, Yuejie Chi, Yuantao Gu, and Yuting Wei. "Sample-efficient reinforcement learning is feasible for linearly realizable MDPs with limited revisiting". In: *Advances in Neural Information Processing Systems* 34 (2021).

[LKFS21] Gene Li, Pritish Kamath, Dylan J Foster, and Nathan Srebro. "Eluder dimension and generalized rank". In: *arXiv preprint arXiv:2104.06970* (2021).

[LSW20] Tor Lattimore, Csaba Szepesvári, and Gellért Weisz. "Learning with Good Feature Representations in Bandits and in RL with a Generative Model". In: *ICML*. 2020, pp. 9464–9472.

[Put14] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[RB14] Stephane Ross and J Andrew Bagnell. "Reinforcement and imitation learning via interactive no-regret learning". In: *arXiv preprint arXiv:1406.5979* (2014).

[RGB11] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 627–635.

[RHYLJR21] Nived Rajaraman, Yanjun Han, Lin Yang, Jingbo Liu, Jiantao Jiao, and Kannan Ramchandran. "On the Value of Interaction and Function Approximation in Imitation Learning". In: *Advances in Neural Information Processing Systems* 34 (2021).

[Ros13] Stephane Ross. "Interactive learning for sequential decisions and predictions". PhD thesis. Carnegie Mellon University, 2013.

[RVR13] Daniel Russo and Benjamin Van Roy. "Eluder Dimension and the Sample Complexity of Optimistic Exploration." In: Citeseer. 2013.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. 2018.

[SS07] Umar Syed and Robert E Schapire. "A game-theoretic approach to apprenticeship learning". In: *Advances in neural information processing systems* 20 (2007).

[SS20] Roshan Shariff and Csaba Szepesvári. "Efficient planning in large MDPs with weak linear function approximation". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19163–19174.

[SVGBB17] Wen Sun, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell. "Deeply aggrevated: Differentiable imitation learning for sequential prediction". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3309–3318.

[Sze10] Csaba Szepesvári. "Algorithms for reinforcement learning". In: *Synthesis lectures on artificial intelligence and machine learning* 4.1 (2010), pp. 1–103.

[WAJAYJS21] Gellert Weisz, Philip Amortila, Barnabás Janzer, Yasin Abbasi-Yadkori, Nan Jiang, and Csaba Szepesvári. "On Query-efficient Planning in MDPs under Linear Realizability of the Optimal State-value Function". In: *arXiv preprint arXiv:2102.02049* (2021).

[WAS21] Gellért Weisz, Philip Amortila, and Csaba Szepesvári. "Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions". In: *Algorithmic Learning Theory*. PMLR. 2021, pp. 1237–1264.

[WR13] Zheng Wen and Benjamin Van Roy. "Efficient Exploration and Value Function Generalization in Deterministic Systems". In: *Advances in Neural Information Processing Systems*. 2013, pp. 3021–3029.

[WSG21] Gellért Weisz, Csaba Szepesvári, and András György. "TensorPlan and the Few Actions Lower Bound for Planning in MDPs under Linear Realizability of Optimal Value Functions". In: *arXiv preprint arXiv:2110.02195* (2021).

[WWK21] Yuanhao Wang, Ruosong Wang, and Sham Kakade. "An Exponential Lower Bound for Linearly Realizable MDP with Constant Suboptimality Gap". In: *Advances in Neural Information Processing Systems* 34 (2021).

[ZLKB19] Andrea Zanette, Alessandro Lazaric, Mykel J Kochenderfer, and Emma Brunskill. "Limiting extrapolation in linear approximate value iteration". In: *Advances in Neural Information Processing Systems* 32 (2019).

[ZLKB20] Andrea Zanette, Alessandro Lazaric, Mykel Kochenderfer, and Emma Brunskill. "Learning near optimal policies with low inherent bellman error". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10978–10989.

A Proof of Theorem 3.1

Hyperparameters for the algorithm We define

$$E_d = 3d\frac{e}{e-1}\ln\{3+3\left(2B/\varepsilon\right)^2\}+1 \tag{9}$$

$$n_{\texttt{rollout}} = \frac{2H^2(1+2B)^2\log\left(\frac{2(E_d+1)}{\delta}\right)}{\varepsilon^2_{\texttt{target}}} \tag{10}$$

$$N = (E_d+1)n_{\texttt{rollout}}HA \tag{11}$$

$$n_{\texttt{eval}} = \frac{50H^2(1+B^2)(d+1)\log\left(\frac{2(d+1)N}{\delta}\right)}{\varepsilon^2_{\texttt{target}}} \tag{12}$$

$$\varepsilon_{\texttt{eval}} = \sqrt{\frac{\log\left(\frac{2(d+1)N}{\delta}\right)}{2n_{\texttt{eval}}}} \tag{13}$$

$$\bar{\varepsilon}_{\texttt{eval}} = \sqrt{1+B^2}\sqrt{d+1}\varepsilon_{\texttt{eval}} \tag{14}$$

$$\varepsilon_{\texttt{tol}} = 4\bar{\varepsilon}_{\texttt{eval}} \tag{15}$$

$$\varepsilon_{\texttt{roll}} = H(1+2B)\sqrt{\frac{\log(2(E_d+1)/\delta)}{2n_{\texttt{rollout}}}} \tag{16}$$

As we will see, E_d is the upper bound on the number of parameters observed before global consistency holds, $n_{\texttt{eval}}$ is the number of samples taken to estimate each TD vector, $n_{\texttt{rollout}}$ is the number of rollouts executed for each parameter, N is the maximum number of state-action pairs which will be seen by DELPHI , $\varepsilon_{\texttt{eval}}$ is the error between the estimated TD vector and the true TD vector, and $\varepsilon_{\texttt{roll}}$ is the error of replacing the expected total rewards with the average over several rollouts.

A.1 Part 1: Concentration Inequalities

Recall that we write

$$\Delta_{s,a} = r(s,a)\oplus\left(\mathbb{E}_{P(s,a)}[\varphi(s')]-\varphi(s)\right)$$

for the true TD vector for state-action (s,a). We write $\hat{\Delta}_{s,a}$ for any estimated TD vector resulting from Line 11, and $\tilde{\Delta}_{s,a}$ for any doubly-estimated TD vector resulting from Line 16.

Lemma 3.2 (Concentration of $\hat{\Delta}_{s,a}$ (Line 11))**.** *For any $s,a\in\mathcal{S}\times\mathcal{A}$ that is observed throughout the execution* DELPHI *, with $n_{\texttt{eval}}$ samples in Line 11, we have that with probability $\geq 1-\delta$, $\left\|\hat{\Delta}_{s,a}-\Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}$ and thus that $\langle 1\oplus\theta, \hat{\Delta}_{s,a}-\Delta_{s,a}\rangle \leq \bar{\varepsilon}_{\texttt{eval}}$.*

Proof. Pick a single s,a, and omit the dependence on s,a for cleanliness. Starting with the reward concentration: at every step we collect $n_{\texttt{eval}}$ iid samples from $R(s_i,a_i)$. By Hoeffding, since rewards are bounded in $[0,1]$, the empirical average satisfies

$$\left|\frac{1}{n}\sum r_i - \mathbb{E}[r]\right| \overset{\text{w.p. } 1-\delta/((d+1)N)}{\leq} \sqrt{\frac{\log(2(d+1)N/\delta)}{2n_{\texttt{eval}}}}.$$

Onto the transition probabilities. Since the features are such that $\|\varphi\|_2 \leq 1$, we also have that $\|\varphi\|_\infty \leq 1$. Thus, each coordinate of the feature map is bounded by 1. By Hoeffding, for each coordinate i of the vector, we have that

$$|\frac{1}{n}\sum_{j=1}^{n}\varphi_i^{(j)} - \mathbb{E}\varphi_i^{(j)}| \overset{\text{w.p. } 1-\delta/((d+1)N)}{\leq} \sqrt{\frac{\log(2(d+1)/\delta)}{2n_{\texttt{eval}}}}$$

Thus, with a union bound over these events for each coordinate i and the event for the reward concentration, we have that w.p. $\geq 1-\delta/N$

$$\left\|\hat{\Delta}-\Delta\right\|_\infty \leq \sqrt{\frac{\log\left(\frac{2(d+1)N}{\delta}\right)}{2n_{\texttt{eval}}}} = \varepsilon_{\texttt{eval}}.$$

Finally, by a union bound over all N state-action pairs observed by DELPHI , we have that for every s, a encountered:

$$\left\|\hat{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \sqrt{\frac{\log\left(\frac{2(d+1)N}{\delta}\right)}{2n_{\texttt{eval}}}} = \varepsilon_{\texttt{eval}} \text{ with probability } \geq 1-\delta.$$

We further find that: $\left\|\hat{\Delta} - \Delta\right\|_2 \leq \sqrt{d+1}\varepsilon_{\texttt{eval}}$ (since Δ and $\hat{\Delta}$ are $(d+1)$-dimensional). By Cauchy-Schwartz this gives, $\forall \theta \in \texttt{Ball}_{\ell_2}(B)$:

$$|\langle 1 \oplus \theta, \hat{\Delta} - \Delta\rangle| \leq \|1 \oplus \theta\|_2 \left\|\hat{\Delta} - \Delta\right\|_2 = \sqrt{1 + \|\theta\|_2^2}\left\|\hat{\Delta} - \Delta\right\|_2 = \sqrt{1+B^2}\sqrt{d+1}\varepsilon_{\texttt{eval}} = \bar{\varepsilon}_{\texttt{eval}}$$

□

Lemma 3.3 ($\tilde{\Delta}_{s,a}$ concentrates even more (Line 16))**.** *Similarly, for all s, a where we call the oracle, with probability $1-\delta$, we have $\left\|\tilde{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}/(2\sqrt{E_d})$, and thus, $\forall \theta \in \texttt{Ball}_{\ell_2}(B)$, $|\langle 1 \oplus \theta, \tilde{\Delta}_{s,a} - \Delta_{s,a}\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/(2\sqrt{E_d})$.*

Proof. Follows from the same proof as Lemma 3.2, just replace $n_{\texttt{eval}}$ by $4E_d n_{\texttt{eval}}$. □

A.2 Part 2: Optimism

Lemma 3.4 (θ° not eliminated)**.** *With probability $\geq 1-\delta$, $\theta^\circ \in \Theta_t$ for all iterations $t \in [E_d+1]$.*

Proof. Let $\tilde{\Delta}_t$ be the TD vector which is added to Θ_t at time $t \in [E_d]$. Note that, by Lemma 3.3, for any given t, we have

$$|\langle 1 \oplus \theta^\circ, \tilde{\Delta}_t - \Delta_t\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/(2\sqrt{E_d}), \tag{17}$$

for all t, w.p. $\geq 1-\delta$. The parameter set is defined as

$$\Theta_t = \{\theta : |\langle 1 \oplus \theta, \tilde{\Delta}_i\rangle| \leq \frac{\bar{\varepsilon}_{\texttt{eval}}}{2\sqrt{E_d}} \; \forall i \in [t]\}.$$

Note that, for θ°, this is equivalent to requiring

$$|\langle 1 \oplus \theta^\circ, \tilde{\Delta}_t - \Delta_t\rangle| \leq \frac{\bar{\varepsilon}_{\texttt{eval}}}{2\sqrt{E_d}},$$

since $\langle 1 \oplus \theta^\circ, \Delta_t\rangle = 0$ for all Δ_t (recall Equation 5). This holds by Eq. (17). □

Lemma 3.5 (Optimism)**.** *Under the event of Lemma 3.4, we have $v_t(s_0) \geq v^\circ(s_0)$, $\forall t \in [E_d]$.*

Proof (of Lemma 3.5). Under the event of Lemma 3.4, θ° is never eliminated from Θ_t. Thus, by the optimistic update rule, we have $\theta_{t+1} = \text{argmax}_{\theta \in \Theta_t} \theta^\top \varphi(s_0)$, and since $\theta^\circ \in \Theta_t$, then

$$\theta_{t+1}^\top \varphi(s_0) = v_{\theta_{t+1}}(s_0) \geq (\theta^\circ)^\top \varphi(s_0) = v_{\theta^\circ}(s_0) = v^\circ(s_0)$$

. □

A.3 Part 3: Iteration Complexity

To bound the iteration complexity of our algorithm, we will need to introduce the notion of *Eluder* dimension. We use a simplified form introduced by [LKFS21], although the first definition is due to [RVR13].

Definition A.1 (Eluder dimension [LKFS21])**.** *Let $\mathcal{F}$ be a real-valued function class on domain $\mathcal{X}$. Fix a reference function $f^\star \in \mathcal{F}$, and a scale ε. The **Eluder dimension** of $\mathcal{F}$ at a scale ε, w.r.t. $f^\star$, is the length $\tau \in \mathbb{N}$ of the longest sequence of points $((x_1, f_1), \ldots (x_\tau, f_\tau))$ such that*

$$\forall i \in [\tau]: \quad |f_i(x_i) - f^\star(x_i)| > \varepsilon, \quad \textit{and} \quad \sum_{j<i}(f_i(x_j) - f^\star(x_j))^2 \leq \varepsilon^2. \tag{18}$$

An Eluder sequence of length τ (with respect to $f^\star$) is any sequence $(x_i, f_i)_{i=1}^\tau$ which satisfies Eq (18) for each i.

In other words, the Eluder dimension is the length of the longest sequence of points such that, for each i, we can find a new function f_i which is large with respect to $f^\star$ on x_i but correctly fits $f^\star$ on historical data $x_1, \ldots, x_j$, for $j < i$. We will use the folllowing bound for the Eluder dimension of linear functions with d-dimensional parameters.

Lemma A.2 ([RVR13]). *For any $f^\star$, the Eluder dimension of the function class $\mathcal{F} = \{f_\theta(x) = \theta^\top x\}$, assuming that $\|\theta\|_2 \leq B$ and $\|x\|_2 \leq \gamma$, is*

$$dim_E(\mathcal{F}, \varepsilon) \leq 3d \frac{e}{e-1} \ln\{3 + 3\,(2B/\varepsilon)^2\} + 1 = \mathcal{O}(d \ln(B/\varepsilon)).$$

This gives us enough to prove our iteration bound – we will show that the sequence of linear parameters chosen by our algorithm together with each new TD vector obtained from the oracle forms an Eluder sequence with respect to $\theta^\star$

Lemma 3.6 (Iteration Complexity). *With probability $\geq 1 - 2\delta$, the iteration complexity of the algorithm is at most the Eluder dimension at scale $\bar{\varepsilon}_{\texttt{eval}}$, i.e. $E_d = \mathcal{O}(d \ln(B/\bar{\varepsilon}_{\texttt{eval}}))$.*

Proof (of Lemma 3.6). Assume the events of Lemma 3.2 and Lemma 3.3, which happen together with probability $\geq 1 - 2\delta$. Our function class is $\mathcal{F} = \{f_\theta(x) = \langle 1 \oplus \theta, x\rangle\}$. This is a subset of all linear functions on $d+1$ dimensions, so by Lemma A.2,[7] it has Eluder dimension at least $\mathcal{O}((d+1)\ln(B/\varepsilon)) = \mathcal{O}(d \ln(B/\varepsilon))$. We pick $f^\star = f_{\theta^\circ}$. We will show that $\forall i$, the sequence $(\tilde{\Delta}_i, f_{\theta_i})$ forms an Eluder sequence with respect to $f^\star$, from which it will follow that its length is bounded by E_d. We do this by induction. The base case is obvious. By the constraint set definition (Line 17), we have $|f_{\theta_i}(\tilde{\Delta}_j)| = |\langle 1 \oplus \theta_i, \tilde{\Delta}_j\rangle| \leq \frac{\bar{\varepsilon}_{\texttt{eval}}}{2\sqrt{E_d}}\, \forall j < i$. Assuming the event of Lemma 3.4, we also have $|f_{\theta^\circ}(\tilde{\Delta}_j)| \leq \frac{\bar{\varepsilon}_{\texttt{eval}}}{2\sqrt{E_d}}$, since $\theta^\circ \in \Theta_{i-1}$. Thus

$$|f_{\theta_i}(\tilde{\Delta}_j) - f_{\theta^\circ}(\tilde{\Delta}_j)| \leq 2\frac{\bar{\varepsilon}_{\texttt{eval}}}{2\sqrt{E_d}}\, \forall i \implies \sum_{j<i} (f_{\theta_i}(\tilde{\Delta}_j) - f_{\theta^\circ}(\tilde{\Delta}_j))^2 \leq \bar{\varepsilon}^2_{\texttt{eval}} \frac{i-1}{E_d} \leq \bar{\varepsilon}^2_{\texttt{eval}},$$

where in the last inequality we have used $i - 1 \leq E_d$ by the induction hypothesis. Thus, the second condition of the Eluder dimension is satisfied. For the first condition, we want to show that $|f_{\theta_i}(\tilde{\Delta}_i) - f_{\theta^\circ}(\tilde{\Delta}_i)| > \bar{\varepsilon}_{\texttt{eval}}$. Note that, by Lemma 3.4, we have $|f_{\theta^\circ}(\tilde{\Delta}_i)| \leq \bar{\varepsilon}_{\texttt{eval}}/(2\sqrt{E_d}) \leq \bar{\varepsilon}_{\texttt{eval}}$. Recall from Line 13 that $\left|\langle \hat{\Delta}_i, 1 \oplus \theta_i\rangle\right| = |f_{\theta_i}(\hat{\Delta}_i)| > \varepsilon_{\texttt{tol}}$. Using concentration and linearity, we have $|f_{\theta_i}(\hat{\Delta}_i)| - |f_{\theta_i}(\tilde{\Delta}_i)| \leq 2\bar{\varepsilon}_{\texttt{eval}}$. Thus, $|f_{\theta_i}(\tilde{\Delta}_i)| \geq \varepsilon_{\texttt{tol}} - 2\bar{\varepsilon}_{\texttt{eval}} = 2\bar{\varepsilon}_{\texttt{eval}}$ since $\varepsilon_{\texttt{tol}} = 4\bar{\varepsilon}_{\texttt{eval}}$. Putting this together gives

$$|f_{\theta_i}(\tilde{\Delta}_i) - f_{\theta^\circ}(\tilde{\Delta}_i)| \geq f_{\theta_i}(\tilde{\Delta}_i) - f_{\theta^\circ}(\tilde{\Delta}_i) \geq 2\bar{\varepsilon}_{\texttt{eval}} - \bar{\varepsilon}_{\texttt{eval}} = \bar{\varepsilon}_{\texttt{eval}}.$$

And we are done. □

A.4 Part 4: Consistency, and Putting Everything Together

Lemma 3.7 (Consistency $\implies$ accurate prediction). *If m rollouts have occured without any inconsistencies (i.e., the if statement of Line 13 never gets triggered), then $v^{\pi_\theta}(s_0) > v_\theta(s_0) - 5H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}}$ with probability $\geq 1 - 3\delta$.*

Proof (of Lemma 3.7). Assume the event of Lemmas 3.2 and Lemmas 3.3, which together happen with probability $\geq 1 - 2\delta$. The third event which we assume will be introduced shortly.

For cleanliness, let us write θ for the final parameter which observes m rollouts without consistency break. Observe the following calculation:

[7] (and using that $\mathcal{F} \subseteq \mathcal{F}' \implies dim_E(\mathcal{F}, \varepsilon) \leq dim_E(\mathcal{F}', \varepsilon)$)

$$
\begin{aligned}
v^{\pi_\theta}(s_0) &= \mathbb{E}_{\pi_\theta}[\sum_{j\in[H]} r_{S_j,A_j}] \\
&= \mathbb{E}\left[\left\langle \left(\sum_{j\in[H]} r_{S_j,A_j}\right) \oplus \varphi(s_{H+1}), 1\oplus\theta \right\rangle\right] && (\varphi(s_{H+1})=0) \\
&= \mathbb{E}\left[\langle\varphi(s_0),\theta\rangle + \sum_{j\in[H]} \langle r_{S_j,A_j} \oplus (\varphi(S_{j+1})-\varphi(S_j)), 1\oplus\theta\rangle\right] && \text{(telescoping sum)} \\
&= \langle\varphi(s_0),\theta\rangle + \mathbb{E}\left[\sum_{j\in[H]} \langle r_{S_j,A_j} \oplus \left(P_{S_j,A_j}\varphi(\cdot)-\varphi(S_j)\right), 1\oplus\theta\rangle\right],
\end{aligned}
$$

Now observe that after $n_{\texttt{rollout}}$ number of rollouts, we have $n_{\texttt{rollout}}$ unbiased estimates of the expected trajectories. Thus we can approximate $\mathbb{E}\left[\sum_{j\in[H]}\langle r_{S_j,A_j} \oplus \left(P_{S_j,A_j}\varphi(\cdot)-\varphi(S_j)\right), 1\oplus\theta\rangle\right] \approx \frac{1}{n_{\texttt{rollout}}}\sum_{i=1}^{n_{\texttt{rollout}}} \mathbb{E}[r_{S_j^i,A_j^i}] \oplus \left(P_{S_j^i,A_j^i}\varphi(\cdot)-\varphi(S_j^i)\right)$, where S_j^i and A_j^i are the states and actions in horizon $j \in [H]$ of rollout $i \in [n_{\texttt{rollout}}]$. More precisely, using Hoeffding's and that $\langle\mathbb{E}[r_{S_j^i,A_j^i}] \oplus \left(P_{S_j^i,A_j^i}\varphi(\cdot)-\varphi(S_j)\right)], 1\oplus\theta\rangle \le 1+2B$, we can get

$$
\left|\frac{1}{n_{\texttt{rollout}}}\sum_{i\in[m]}\left(\sum_{j\in[H]}\langle\Delta_j^i, 1\oplus\theta\rangle\right) - \left(\mathbb{E}\sum_{j\in[H]}\langle\Delta_j^i, 1\oplus\theta\rangle\right)\right| \le H(1{+}2B)\sqrt{\frac{\log(2(E_d+1)/\delta)}{2m}} := \varepsilon_{\texttt{roll}},
$$

w.p. $\ge 1-\frac{\delta}{E_d+1}$, where we wrote $\Delta_j^i = \mathbb{E}[r_{S_j^i,A_j^i}] \oplus \left(P_{S_j^i,A_j^i}\varphi(\cdot)-\varphi(S_j)\right)$. By a union bound this happens for all $t \in [E_d+1]$ with probability $\ge 1-\delta$. Picking up where we left off:

$$
\begin{aligned}
v^{\pi_\theta}(s_0) &\ge \langle\varphi(s_0),\theta\rangle + \frac{1}{n_{\texttt{rollout}}}\sum_{i=1}^{n_{\texttt{rollout}}}\sum_{j\in[H]}\langle\mathbb{E}[r_{S_j^i,A_j^i}] \oplus \left(P_{S_j^i,A_j^i}\varphi(\cdot)-\varphi(S_j)\right)], 1\oplus\theta\rangle - \varepsilon_{\texttt{roll}} \\
&\ge \langle\varphi(s_0),\theta\rangle + \frac{1}{m}\sum_{i=1}^{m}\sum_{j\in[H]}\langle\hat{\Delta}_{s_j^i,a_j^i}, 1\oplus\theta\rangle - H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}} && \text{(evaluation error)} \\
&\ge v_\theta(s_0) - \frac{1}{m}\sum_{i=1}^{m}\sum_{j\in[H]} 4\bar{\varepsilon}_{\texttt{eval}} - H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}} && \text{(consistency holds)} \\
&= v_\theta(s_0) - 5H\bar{\varepsilon}_{\texttt{eval}} - \varepsilon_{\texttt{roll}}
\end{aligned}
$$

and we are done. The second inequality follows from Lemma 3.2, which holds for all steps in all m trajectories in all E_d+1 iterations. The third inequality holds since, for every S_j^i, there exists a consistent action, i.e. an action such that $|\langle\hat{\Delta}_j^i, 1\oplus\theta\rangle| \le \varepsilon_{\texttt{tol}} = 4\bar{\varepsilon}_{\texttt{eval}}$. □

A.5 Simulator inaccuracy and misspecification

We start with the case of inaccurate simulators. Recall that we say that a simulator is λ-*innacurate* if the samples obtained are of the form $(\Pi(r+\lambda_{s,a}), s')$, for any (s,a) and for some constant $\lambda_{s,a}$ that satisfies $|\lambda_{s,a}| \le \lambda$.

Theorem 3.8 (DELPHI with misspecification)**.** *Redefine $n'_{\texttt{eval}} = 4n_{\texttt{eval}}$ (Eq. 12) and all subsequent hyperparameters which depend on $n'_{\texttt{eval}}$. Then we have that, for all MDPs that are at most $\frac{\bar{\varepsilon}_{\texttt{eval}}}{8\sqrt{E_d}}$-misspecified or for all simulators that are at most $\frac{\bar{\varepsilon}_{\texttt{eval}}}{4\sqrt{E_d}}$-innaccurate, the conclusions of Theorem 3.1 continue to hold when running* DELPHI *with the new hyperparameters.*

We can repeat the proof of DELPHI , and in fact the only difference will be in Part 1 of the proof (Lemmas 3.2 and 3.3) will continue to hold, which also implies that the rest of the proof will continue to hold.

Lemma A.3 (Concentration of $\hat{\Delta}_{s,a}$ (Line 11))**.** *For any $s, a \in \mathcal{S} \times \mathcal{A}$ that is observed throughout the execution* DELPHI *, with $n_{\texttt{eval}}$ samples in Line 11, we have that with probability $\geq 1 - \delta$, $\left\|\hat{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}$ and thus that $\langle 1 \oplus \theta, \hat{\Delta}_{s,a} - \Delta_{s,a}\rangle \leq \bar{\varepsilon}_{\texttt{eval}}$.*

Lemma A.4 ($\tilde{\Delta}_{s,a}$ concentrates even more (Line 16))**.** *Similarly, for all s, a where we call the oracle, with probability $1 - \delta$, we have $\left\|\tilde{\Delta}_{s,a} - \Delta_{s,a}\right\|_\infty \leq \varepsilon_{\texttt{eval}}/(2\sqrt{E_d})$, and thus, $\forall \theta \in \texttt{Ball}_{\ell_2}(B)$, $|\langle 1 \oplus \theta, \tilde{\Delta}_{s,a} - \Delta_{s,a}\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/(2\sqrt{E_d})$.*

Proof (of Lemmas A.3 and A.4. We repeat the proof of Lemmas 3.2 and 3.3. Note that $\hat{\Delta}_{s,a}$ is an average of i.i.d. random variables with mean $\Delta'_{s,a} = (r(s,a) + \lambda_{s,a}) \oplus P_{s,a}\varphi - \varphi(s)$. As before, by Hoeffdings, we have $|\langle 1 \oplus \theta, \hat{\Delta} - \Delta'\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/2$, where the factor of $1/2$ is due to the definition of $n'_{\texttt{eval}}$. This gives that

$$|\langle 1 \oplus \theta, \hat{\Delta} - \Delta\rangle| \leq |\langle 1 \oplus \theta, \hat{\Delta} - \Delta'\rangle| + |\langle 1 \oplus \theta, \Delta' - \Delta\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/2 + \lambda \leq \bar{\varepsilon}_{\texttt{eval}}/2 + \bar{\varepsilon}_{\texttt{eval}}/4\sqrt{E_d} \leq \bar{\varepsilon}_{\texttt{eval}}$$

The proof of the second part follows similarly, since we have

$$|\langle 1 \oplus \theta, \tilde{\Delta} - \Delta\rangle| \leq |\langle 1 \oplus \theta, \tilde{\Delta} - \Delta'\rangle| + |\langle 1 \oplus \theta, \Delta' - \Delta\rangle| \leq \bar{\varepsilon}_{\texttt{eval}}/4\sqrt{E_d} + \lambda \leq \bar{\varepsilon}_{\texttt{eval}}/2\sqrt{E_d}$$

□

Next we handle the value misspecification case. Recall that the MDP is η-*misspecified* for the expert policy π° and the features φ if there exists θ° such that $\sup_s |v^\circ(s) - \langle \varphi(s), \theta^\circ\rangle| \leq \eta$. Here, we use a reduction argument to show that an η-misspecified MDP can be reduced to a 2η-innacurate simulator of a realizable MDP. Namely, using the same reduction as Appendix D of [WAJAYJS21], we can construct an alternative MDP $\mathcal{M}'$ such that $\mathcal{M}'$ is realizable but is a 2η-inaccurate simulator of $\mathcal{M}$. The result then follows from the first part of the proof.

B Proof of Section 4

B.1 The MDP construction

Theorems 4.1 and 4.3 use the same MDP construction, which is inspired by the recent and remarkable lower bound of [WSG21] (itself an extension of the lower bound of [WAS21]). We give an overview of the MDP and its features, and describe the changes from the original construction of [WSG21]. The state space consists of a hypercube in p dimensions, $\mathcal{S} = \{\pm 1\}^p$, for some $p \in \mathbb{N}$, and assume for simplicity that p is divisible by 4. More specifically, the state space at stage h also contains the history of vectors encountered so far $(s_1, \ldots, s_h)$, which is uniquely defined as the transitions are deterministic. The dimension p will end up being $p \approx \sqrt{d}$ for the value-based lower bound and $p \approx d$ for the policy-based lower bound. We write $\rho(\cdot, \cdot)$ for the *Hamming* distance on $\mathcal{S}$, recalling that is a bilinear function of its arguments, i.e. $\rho(s_1, s_2) = \frac{1}{2}(p - \langle s_1, s_2 \rangle)$. The action set is $\mathcal{A} = [p]$, and each action $a \in [p]$ will correspond to flipping the ath bit of the current state. Each trajectory of horizon H has K *phases* ($K \in \mathbb{N}$), and each phase consists of p steps (thus, $H = Kp$). We write $s_0, s_1, \ldots, s_k, \ldots s_K$ for the states reached at the end of each phase, and when we need to we will use $s_{k,i}$ for a state at step i of phase k. The start state is $s_0 = \vec{1}$, the all-ones vector.

There is a special "goal state" $s^\star$, and reward is given only if a) at the end of any phase, $\rho(s_k, s^\star) \leq p/4$, or b) the learner reaches horizon H, i.e. the end of phase K. The reward function decays geometrically, and is defined according to the sequence of states $s_0, \ldots s_k$ obtained at the end of each phase. Letting $g(s_1, s_2) \coloneqq 1 - \rho(s_1, s_2)/p$ denote one minus the proportion of bits where two states differ, the reward function for reaching a $p/4$-neighbourhood of $s^\star$ at stage k (condition a) just described) is deterministic and has value

$$r_{w^\star}((s_i)_{i=1}^k) = \left(\prod_{i=1}^{k} g(s_{i-1}, s_i) \right) g(s_i, s^\star) \tag{19}$$

The reward function at stage H (condition b) above) is always given and has the same expectation as Eq. (19), but will be a Bernoulli random variable.

Modulo some exceptions (to be described shortly), the transition function is deterministic and is defined via $\mathcal{P}(\tau(s, a)|s, a) = 1$, where $\tau(s, a) = (s_1, \ldots, -s_a, \ldots s_p)$ corresponds to the new vector obtained from flipping the bit at index $a \in [p]$. The exceptions to these transition dynamics are if 1) a state within a $p/4$-neighbourhood of $s^\star$ is reached, or 2) a move is repeated. In the first case, the MDP transitions to a game-over state $\perp$ (after which nothing else is possible and no reward is given). In the second case, if the move is repeated within the first $p/4$ steps of a phase then the MDP similarly transitions to $\perp$, and if the move is repeated in the second $3p/4$ turns then the current state becomes frozen until the end of the current phase (i.e. no further bit flips are allowed). This implies that each bit can only be flipped once in each phase, and further that $g(s_{i-1}, s_i) \leq 3/4$ for each "legal" trajectory in the MDP (thus, the reward decays geometrically, cf. Equation (19)).

So far, the main modification in our construction from that of [WSG21] is the reward function. Their g function is chosen to be 2nd order in $\rho(s_1, s_2)$, while ours is linea in ρ. We now introduce the definition of the expert policy: we simply choose it to be the one which will flip the *earliest* index such that $s_{k,i}$ differs from $s^\star$ and such that its index has not yet been played in the round. If no such index exists, or if $s_{k,i} = s^\star$, the expert policy will simply freeze the current round by repeating the earliest index that has already been played. (Note that there is always a repeated index in this case, and that repeating an index will lead to freezing instead of termination, since at the start of any phase k the trajectory must satisfy $\rho(s_k, s^\star) > p/4$, otherwise the trajectory would have terminated). It will turn out that the expert's trajectory *from the start state* will be identical to that of the optimal policy (and, thus, will be equally difficult to compete with, in the sense of Eq. (2)). While our expert policy might seem optimal, this need not be the case for arbitrary states since it might sometimes be more advantageous for a state in phase k which can no longer reach the $p/4$-neighbourhood of $s^\star$ to simply aim at minimizing the inevitable factor of $g(s_{k-1}, s_k)$ that it will incur. The following lemma (proved in Appendix B.2) gives an expression for the value function corresponding to our expert policy, and shows that the value function satisfies Assumption 2.2.

Lemma B.1. *Let $s_{k,i}$ be a state in round i of phase k. Note that, from $s_{k,i}$, the state s_{k+1} that π° will reach at the end of the current phase is deterministic. The value function of π° is*

$$v^\circ(s_{k,i}) = \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'})\right) g(s_k, s_{k+1}) g(s_{k+1}, s^\star),$$

and furthermore this is linear with features φ_v of dimension $d = \Theta(p^2)$.

The next lemma gives an expression showing that this same expert policy is linear in a different set of features φ_π, and thus that the expert policy satisfies Assumption 4.2.

Lemma B.2. *There exists a feature map φ_π and parameter θ_π of dimension $d = \Theta(p)$ such that*

$$\pi^\circ(s_{k,i}) = \mathrm{argmax}_a\{\langle \varphi_\pi(s,a), \theta_\pi \rangle\}$$

This lemma is also proved in Appendix B.2.

The essence of our lower bound is that each oracle call will reveal one bit of the secret state $s^\star$, and thus without $\tilde{\Omega}(p)$ calls and learner will need exponentially-many exploratory samples to solve this MDP. Thus, for the value-based lower bound, an agent with a d-dimensional feature map must be given an MDP which has a $p = \Theta(\sqrt{d})$-dimensional state space. However, for the policy-based lower bound, we can give the agent a $p = \Theta(d)$-dimensional MDP.

Information-theoretic lower bound Information-theoretically, a learning algorithm can solve this MDP only if they recover the secret state $s^\star$. We let $\mathrm{MDP}(p, K)$ denote an instance of the above MDP with dimension p and parameters K. Following the approach of [WSG21], we prove the sample complexity hardness by reducing each $\mathrm{MDP}(p, K)$ to an instance of an abstract game called $\textsc{CubeGame}(p, K)$. Details on $\textsc{CubeGame}$, and the proof of the following theorem, are deferred to Appendix B.3. The main result is the following:

Theorem B.3. *Any learner which solves $\mathrm{MDP}(p, K)$ can be used to solve $\textsc{CubeGame}(p, K)$. Unless the number of oracle calls is $\Omega(\frac{p}{\log p})$, any learner which is 0.01-optimal on $\textsc{CubeGame}(p, K)$, with a sample complexity of N, will need*

$$N \geq 2^{\Omega(p \wedge K)}.$$

Combined with the fact that a learner provided with d-dimensional features is given an MDP with parameter $p \approx \sqrt{d}$, this gives the result of Theorem 4.1.

B.2 Proofs of Lemma B.1, B.2

We need some more notation in order to linearize the value function. Let $s_{k,i} \neq \perp$ be a state in step i of phase k. We define the variable $\mathrm{ct}^{\mathrm{flip}}_{k,i} = \rho(s_{k,0}, s_{k,i})$, which simply measures the number of components flipped so far in round k of step i, and $\mathrm{fix}_{k,i} \in \{0,1\}^p$ which is a vector with 1 at a component j if said component is currently frozen (i.e. if it has been played or if the entire state has been frozen), and 0 otherwise. Similarly, there are two scalars $e^{\mathrm{fix}}_{k,i}$ and $e^{\neg\mathrm{fix}}_{k,i}$ which simply count the number of components which disagree with $s^\star$ that are currently frozen (for $e^{\mathrm{fix}}_{k,i}$) or not frozen (for $e^{\neg\mathrm{fix}}_{k,i}$). Note that only $e^{\mathrm{fix}}_{k,i}$ and $e^{\neg\mathrm{fix}}_{k,i}$ depend on $s^\star$, and in fact we have:

$$e^{\mathrm{fix}}_{k,i} = \frac{1}{2}(\langle \vec{1}, \mathrm{fix}_{k,i} \rangle - \langle \mathrm{fix}_{k,i} \cdot s_{k,i}, s^\star \rangle) \tag{20}$$

$$e^{\neg\mathrm{fix}}_{k,i} = \frac{1}{2}(\langle \vec{1}, \neg\mathrm{fix}_{k,i} \rangle - \langle \neg\mathrm{fix}_{k,i} \cdot s_{k,i}, s^\star \rangle), \tag{21}$$

where $\mathrm{fix}_{k,i} \cdot s_{k,i}$ denotes component-wise multiplication, i.e. $(\mathrm{fix}_{k,i} \cdot s_{k,i})_j = (\mathrm{fix}_{k,i})_j \cdot (s_{k,i})_j$. Now, we have that:

Lemma B.4 (Value of v°)**.** *Let $s_{k,i}$ be a state in round i of phase k. Let s_{k+1} denote the state that π° will reach at the end of the current phase when starting from $s_{k,i}$ (and note that this choice is deterministic given $s_{k,i}$, and that it may not be in the $p/4$-neighbourhood of $s^\star$). Then we have:*

$$v^\circ(s_{k,i}) = \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'})\right) g(s_k, s_{k+1}) g(s_{k+1}, s^\star),$$

or, overloading notation and letting $g(x) = 1 - x/p$, we have

$$v^\circ(s_{k,i}) = \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) g(ct_{k,i}^{flip} + e_{k,i}^{\neg fix}) g(e_{k,i}^{fix}). \tag{22}$$

Proof. Identical to [WSG21, Lemma 4.9] □

Lemma B.5. *The value function v° is linear in features φ_v with dimension $d = \Theta(p^2)$.*

Proof. Starting from Equation 22, we observe that only $e_{k,i}^{\text{fix}}$ and $e_{k,i}^{\neg\text{fix}}$ depend on $s^\star$. The first term in parentheses in simply a scalar which multiplies the features. We thus calculate linear expressions for $x = \text{ct}_{k,i}^{\text{flip}} + e_{k,i}^{\neg\text{fix}}$ and $y = e_{k,i}^{\text{fix}}$. Starting with $y = e_{k,i}^{\text{fix}}$, we have:

$$y = \frac{1}{2}\left(\langle \vec{1}, \text{fix}_{k,i} \rangle - \langle \text{fix}_{k,i} \cdot s_{k,i}, s^\star \rangle \right) = a + \langle b, s^\star \rangle,$$

where $a = \frac{1}{2}\langle \vec{1}, \text{fix}_{k,i} \rangle$ and $b = -\frac{1}{2}\text{fix}_{k,i} \cdot s_{k,i}$. Similarly:

$$x = \text{ct}_{k,i}^{\text{flip}} + \frac{1}{2}\left(\langle \vec{1}, \neg\text{fix}_{k,i} \rangle - \langle \neg\text{fix}_{k,i} \cdot s_{k,i}, w^\star \rangle \right) = c + \langle d, s^\star \rangle,$$

where $c = \text{ct}_{k,i}^{\text{flip}} + \frac{1}{2}(\langle \vec{1}, \neg\text{fix}_{k,i} \rangle)$ and $d = -\frac{1}{2}\neg\text{fix}_{k,i} \cdot s_{k,i}$. Thus we have: $g(y) = 1 - y/p = 1 - (a + \langle d, s^\star \rangle)/p = (1 - a/p) - \langle \bar{b}, \bar{s^\star} \rangle = a' + \langle \bar{b}, \bar{s^\star} \rangle$, where $a' = (1 - a/p)$, $\bar{b} = -b/\sqrt{p}$ and $\bar{s^\star} = s^\star/\sqrt{p}$. Similarly, $g(x) = 1 - x/p = 1 - (c + \langle d, s^\star \rangle)/p = c' - \langle \bar{d}, \bar{s^\star} \rangle$, where $c' = 1 - c/p$ and $\bar{d} = d/\sqrt{p}$. Putting this together we have that

$$\begin{aligned}
v^\circ(s_{k,i}) &= \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) g(x) g(y) \\
&= \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) (a' + \langle \bar{b}, \bar{s}^\star \rangle)(c' + \langle \bar{d}, \bar{s}^\star \rangle) \\
&= \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) \left(a'c' + c'\langle \bar{b}, \bar{s}^\star \rangle + a'\langle \bar{d}, \bar{s}^\star \rangle + \langle \bar{b}, \bar{s}^\star \rangle \langle \bar{d}, \bar{s}^\star \rangle \right) \\
&= \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) \left(a'c' + \langle c'\bar{b} + a'\bar{d}, \bar{s}^\star \rangle + \langle \bar{b} \otimes \bar{d}, \bar{s}^\star \otimes \bar{s}^\star \rangle \right),
\end{aligned}$$

where we have use a property of the tensor product that $\langle a_1, b_1 \rangle \langle a_2, b_2 \rangle = \langle a_1 \otimes a_2, b_1 \otimes b_2 \rangle$, where $a_1 \otimes a_2, b_1 \otimes b_2 \in \mathbb{R}^{p \times p}$ and their inner product is interpreted as the inner product between the vectorized matrices. Thus, if we take

$$\theta_v = 1 \oplus \bar{s}^\star \oplus (\bar{s}^\star \otimes \bar{s}^\star) \in \mathbb{R}^{1+p+p^2}$$

and

$$\varphi(s_{k,i}) = \left(\prod_{k' \in [k]} g(s_{k'-1}, s_{k'}) \right) \left(a'c' \oplus \left(c'\bar{b} + a'\bar{d} \right) \oplus \left(\bar{b} \otimes \bar{d} \right) \right) \in \mathbb{R}^{1+p+p^2},$$

then we have $v^\circ(s_{k,i}) = \langle \varphi(s_{k,i}), \theta_v \rangle$ as desired. Thus v° is linear with features in dimension $1 + p + p^2$. Note that the norm of the features and the parameter θ_v is also bounded by constants. □

This completes the proof for v°-linearity. We next tackle the analogous statement for π°-linearity.

Lemma B.6. *There exists a feature map φ_π and parameter θ_π of dimension $d \approx p$ such that*

$$\pi^\circ(s_{k,i}) = \text{argmax}_a \{ \langle \varphi_\pi(s, a), \theta_\pi \rangle \}$$

Proof. Let $s_{k,i}$ be a state of interest. Recall that π° will either 1) flip the earliest index which has not been fixed such that the value of $s_{k,i}$ at that index disagrees with $s^\star$, or 2) if no such index exists, freeze the current round by playing a frozen index. First consider a state $s_{k,i}$ such that $\pi^\circ(s_{k,i})$ will flip an index. Since the index flipped was previously incorrect and will thereafter agree with $s^\star$ on that bit, this corresponds to minimizing the distance between s' and $s^\star$ amongst all possible s' which can be reached in one step from $s_{k,i}$ (i.e. amongst all possible $\{\tau(s_{k,i}, a)\}_a$, recalling that $\tau(s_{k,i}, a)$ is our notation for the transition function of the MDP). Thus, $\pi^\circ(s_{k,i}) \in \operatorname{argmin}_a\{\rho(\tau(s_{k,i}, a), s^\star)\}$. This can be written linearly as $\operatorname{argmin}_a\{\frac{1}{2}(p - \langle\tau(s_{k,i}, a), s^\star\rangle)\} = \operatorname{argmax}_a\{\frac{1}{2}(\langle\tau(s_{k,i}, a), s^\star\rangle - p)\} = \operatorname{argmax}_a\{\frac{1}{2}(\langle p \oplus \langle\tau(s_{k,i}, a), s^\star\rangle, 1 \oplus s^\star\rangle)\}$. The second case is that π° will freeze the round at the state $s_{k,i}$. This means that there are no indices which are incorrect that have not been frozen in this round. Again, this corresponds to minimizing the distance between $\tau(s_{k,i}, a)$ and $s^\star$: all other choices will either result in a game over (which has 0 value) or will flip an incorrect bit (which increases the distance). Thus, again we have $\pi^\circ(s_{k,i}) \in \operatorname{argmin}_a\{\rho(\tau(s_{k,i}, a), s^\star)\} = \operatorname{argmax}_a\{\frac{1}{2}(\langle p \oplus \langle\tau(s_{k,i}, a), s^\star\rangle, 1 \oplus s^\star\rangle)\}$.

Thus, in either case, we have that π° is linear with features $\varphi_\pi(s, a) = p \oplus \tau(s_{k,i}, a)$ and $\theta^\circ = 1 \oplus s^\star$. Since the definition of the argmax is scale-insensitive, we can further normalize to obtain that the features are bounded in magnitude by a constant. □

B.3 CUBEGAME with expert advice, and Proof of Theorem B.3

Following the approach of [WSG21], we give our lower bound by providing a reduction to an abstract game called CUBEGAME. Any learning algorithm which can solve the MDPs in our construction can also be used to solve CUBEGAME, and thus it follows that the learner will be subject to the same lower bound. For our setting, we modify the reward function CUBEGAME and augment the learner with the ability to query an expert, which will behave identically to the expert policy which we have defined in our MDPs (that is, it will flip the first bit which is incorrect or give a special actions to indicate if all of the bits are correct). For the rest of this section, when referring to CUBEGAME we are referring to our modified version.

In [WSG21] it is shown that any algorithm which outputs a $0.01-$optimal answer for the (expert-less) CUBEGAME will need a query complexity of $N \geq 2^{\Omega(p \wedge K)}$. In what follows, we will provide the analogous proof of this for our modified game. Thus, the main result of this section is the following, which states that if the learner is not given a budget of $\Omega(p/\log p)$ expert queries, the sample complexity remains exponential. A learning algorithm for CUBEGAME will be called a *planner*, and a planner which returns a 0.01-optimal answer at the end of CUBEGAME will be called *sound*.

Rules of CUBEGAME CUBEGAME is a bandit-like game which is defined by two parameters: a length $K \in \mathbb{N}_+$ and a dimension $p \in \mathbb{N}_+$. The "action space" is $W = \{\pm 1\}^p$. Recall that $\rho(x, y) = \frac{1}{2}(p - x^\top y)$ is the Hamming distance between two vectors in W. The secret parameter which solves the game is housed in the set $W^\star = \{w \in W \mid \frac{p}{4} \leq \rho(\vec{1}, w) \leq \frac{3p}{4}\}$. The planner can only input sequences of vectors where each vector is sufficiently far from the previous one. Formally, for any $k \in [K]$, we let $W^{\circ k} = \{(w_i)_{i\in[k]} \in W^k \mid \forall i \in [k] : \rho(w_{i-1}, w_i) \geq p/4\}$, with $w_0 := \vec{1}$. The action space is: $\mathcal{A} = \cup_{k\in[K]} W^{\circ k}$, thus the planner can input any sequence of length $k \leq K$ satisfying that $(w_i) \in W^{\circ k}$.

The reward function is defined by

$$f_{w^\star}\left((w_i)_{i\in[k]}\right) = \left(\prod_{i\in[k]} g(\rho(w_{i-1}, w_i))\right) g(\rho(w_k, w^\star)),$$
$$\text{where: } g(x) = 1 - \frac{x}{p},$$

with base case $f(()) = g(\rho(w_0, w^\star))$.

While the planner plays, it chooses sequence lengths $L_t \in [K]$ and input sequences $S_t = (w_i^t)_{i\in[L_t]} \in W^{\circ L_t}$. If it chooses to stop playing, it chooses an output $S_t = (w_i^t)_{i\in[8]} \in W^{\circ 8}$ (we distinguish this

case by letting $L_t = 0$ denote that the planner has chosen to terminate). Thus the number of actions taken is $N = \min\{t \in \mathbb{N}_+ \mid L_t = 0\}$.

After any action S_t, the reward is given if either $\rho(w^t_{L_t}, w^\star) < p/4$ or if $L_t = K$. In both cases the reward is $\texttt{Ber}(f_{w^\star}(S_t))$, a Bernoulli random variable with mean $f_{w^\star}(\cdot)$. Similarly, if the planner is done (i.e. the input is $S_N \in W^{\circ 8}$), then the reward given is $R = f_{w^\star}((w^N_i)_{i\in[k^\star]})$, where $k^\star = \min\{8, \min\{k \mid \rho(w^N_k, w^\star) < p/4\}\}$.

The last thing to specify is the oracle. Here, we allow the planner to query the oracle *part-way* through a sequence. Namely, if the planner chooses to input a sequence of length $L_t < K$, then the oracle can be queried at the end of the sequence, and a second sequence of length $L^2_t \leq K - L_t$ can be inputted. This can be repeated as many times as desired, given that the total sequence length remains $\leq K$. The oracle will simulate the expert policy from before: upon being called at a vector $w_{k,i}$, it will either return the index of the first bit which does not agree with $w^\star$, or it will return a special action indicating that all bits are correct.

We are now ready for the main theorem.

Theorem B.7. *Any sound planner for* CUBEGAME *which has less than* $\Omega(p/\log p)$ *oracle queries, will have a sample complexity*

$$N \geq 2^{\Omega(p \wedge K)}$$

The proof comes in 5 lemmas, two of which (Lemmas B.8 and B.10) are analogous to properties from the expert-less CUBEGAME. The other 3 lemmas are information-theoretic and are specific to the oracle setting.

First, some properties about the reward function of CUBEGAME.

Lemma B.8 (Properties of $f_{w^\star}$)**.** *For any* $w^\star \in W^\star, k \in \mathbb{N}$, $s = (w_{k'})_{k'\in[k]} \in W^{\circ k}$, *we have*

$$\frac{1}{4} \leq f_{w^\star}(()) \leq \frac{3}{4}$$
$$0 < f_{w^\star}(s) \leq \left(\frac{3}{4}\right)^{k + \mathbb{1}[\rho(w_k, w^\star) \geq p/4]}$$

Proof. The proof is analogous to [WSG21, Lemma 4.2], substituting our first-order g function. □

The following parameters will control our sample complexity:

$$n = \min\left\{\exp(p/8)p^{-x}/20 - 5, \left(\tfrac{1}{\varepsilon} - 1\right)/9.9\right\}, \quad \text{where} \tag{23}$$
$$x = \frac{p}{16}\log p, \quad \text{and} \tag{24}$$
$$\varepsilon = \left(\frac{3}{4}\right)^{K+1} \tag{25}$$

We will see that $N = \Omega(n)$ for any sound planner.

In the original game of [WSG21], the interaction protocol is captured by $(X_t, Y_t)_{t\in[N]}$, where

- $N = \min\{t \in \mathbb{N}_+ \mid L_t = 0\}$ is the interaction length
- $L_t \in [K]$ is the input length chosen,
- S_t is the sequence $(w_i)_{i\in[L_t]}$ inputted, satisfying $\rho(w_{i-1}, w_i) \geq \frac{p}{4}$
- $X_t = (L_t, S_t)$,
- $U_t = \mathbb{1}\{\rho(w_{L_t-1}, w^\star) < p/4\}$,
- $V_t = \mathbb{1}\{\rho(w_{L_t}, w^\star) < p/4\}$, and
- $Z_t = 0$ unless $V_t = 1$ or $L_t = K$, in which case $Z_t = \text{Ber}(f_{w^\star}(S_t))$,
- $Y_t = (U_t, V_t, Z_t)$.

In our case, we need the sequence S_t to include every bit flip, thus $S_t = (w_{k,i})_{k\in[L_t],i\in[p]}$We also have two new variables, namely $O_t = (o_{k,i})$ which is an indicator that the oracle was called at step i of phrase k and $E_t = (e_{k,i})$ which is the answer returned by the oracle. Thus our new interaction protocol is defined by $X_t = (L_t, S_t, O_t, E_t)$ and $Y_t = (U_t, V_t, Z_t)$, where Y_t remains unchanged.

The planner A, with n interactions, in the environment defined by $w^\star$, defines a distribution over the environment:

$$P^{A,n}_{w^\star}((X_t, O_t, E_t, Y_t)_t) = \prod_{i=1}^{n} p(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p(e_i|x_i, o_i)p(y_i|x_i). \tag{26}$$

Note that $p(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})$ and $p(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})$ are decisions made by the planner and $p(e_i|x_i, o_i)p(y_i|x_i)$ and $p(y_i|x_i)$ are obtained by querying the environment. We define the "abstract game $(0, w^\star)$" to always yield reward 0, and which has the same oracle as environment $w^\star$. It's distribution will be written as $P^A_{(0,w^\star)}$. Let $E^{w^\star}_n$ be the event that in the first n steps the planner does not hit on any vector that is close to $w^\star$:

$$E^{w^\star}_n = \cap_{t\in[n]} \left\{ t > N \text{ or } (t = N \text{ and } \min_{i\in[8]} \rho(w^N_i, w^\star) \geq \tfrac{p}{4}) \text{ or } (t < N \text{ and } \rho(w_{L_t-1}, w^\star) \geq \tfrac{p}{4} \text{ and } \rho(w_{L_t}, w^\star) \geq \tfrac{p}{4}) \right\}$$

Lemma B.9 (A first change of measure)**.** *For any planner A and any $w^\star \in W$, we have*

$$P^A_{w^\star}(E^{w^\star}_n) \geq \tfrac{9}{10} P^A_{(0,w^\star)}(E^{w^\star}_n)$$

Proof. It will be shown that

$$P^A_{w^\star}(E^{w^\star}_n) \geq (1-\varepsilon)^n P^A_{(0,w^\star)}(E^{w^\star}_n). \tag{27}$$

Since $1 - \varepsilon \geq 1 - \frac{1}{1+9.9n}$ by our definition of n we have that $(1-\varepsilon)^n \geq (1 - \frac{1}{1+9.9n})^n \geq \lim_{n\to\infty}(1 - \frac{1}{1+9.9.n})^n > 9/10$, and thus Equation (27) implies our result.

Let $\mathcal{H}$ be the set of all possible histories (x_t, o_t, e_t, y_t) of length n, and $E_h = E^{w^\star}_n \cap \{H = h\}$. Note that $E^{w^\star}_n$ is the disjoint union of E_h, so it is enough to show that for each h we have:

$$\rho = \frac{P^A_{w^\star}[E_h]}{P^A_{(0,w^\star)}[E_h]} \geq (1-\varepsilon)^n,$$

for each h such that $P^A_{(0,w^\star)}(E_h) > 0$. So, let $h = (x_t, o_t, e_t, y_t)$ be such that $P^A_{(0,w^\star)}(E_h) > 0$. This implies in particular that $y_t = (0,0,0)\,\forall t$. Now, both $P^A_{w^\star}$ and $P^A_{(0,w^\star)}$ factorize according to Eq. (26), giving:

$$\frac{P^A_{w^\star}[E_h]}{P^A_{(0,w^\star)}[E_h]} = \prod_i \frac{p(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p^{w^\star}(e_i|x_i, o_i)p^{w^\star}(y_i|x_i)}{p^A(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p^A(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p^{(0,w^\star)}(e_i|x_i, o_i)p^{(0,w^\star)}(y_i|x_i)}.$$

Since we are conditioning on the same fixed history, the terms involving decisions made by the planner will cancel, and similarly the variable E_i also behaves the same in both environments (since the oracle for $w^\star$ is the same). We are left with:

$$\rho = \prod_i \frac{p^{w^\star}(y_i = (0,0,0)|x_i)}{p^{(0,w^\star)}(y_i = (0,0,0)|x_i)}$$

The denominator always has probability 1 in the environment $(0, w^\star)$, and since $U_t = V_t = 0$ under the set $E^{w^\star}_n$ (the planner is never close to $w^\star$), we have $Y_t = 0 \iff Z_t = 0$, so it remains to control

$$\rho = \prod_{i=1}^{n} P^A_{w^\star}(Z_t = 0|x_t).$$

Again since the planner is never close to $w^\star$, $Z_t = 1$ only if $l_t = K$, in which case we have $P^A_{w^\star}(Z_t = 1|x_t) \geq (3/4)^{K+1} = \varepsilon$ by definition of the reward $f_{w^\star}$ obtained from reaching level K. □

The next lemma simply bounds the number of vectors in W which are close to any fixed vector.

Lemma B.10 (Hypercube counting)**.** *For $\tilde{w} \in W$, let $\mathrm{W}_{close}(\tilde{w}) = \{w \in W \mid \rho(w, \tilde{w}) < p/4\}$. Then $|\mathrm{W}_{close}(\tilde{w})| \leq 2^p \exp(-p/8)$*

Proof. Identical to [WSG21, Lemma 4.4]. □

Recall that $E_n^{w^\star}$ is the "bad event" for the planner. We study its complement, $(E_n^{w^\star})^c$, which satisfies $(E_n^{w^\star})^c \subset \{w^\star \in Z\}$, where

$$Z \coloneqq \cup_{t \in [n \wedge (N-1)]} \left(\mathrm{W}_{\text{close}}(w^t_{L_t - 1}) \cup \mathrm{W}_{\text{close}}(w^t_{L_t}\right) \cup (\cup_{i \in [8]} \mathrm{W}_{\text{close}}(w_i^N)),$$

i.e. the event that for some t we have $\rho(w^t_{L_t-1}, w^\star) < p/4$ or $\rho(w^t_{L_t}, w^\star) < p/4$ or that for some $i \in [8]$ we have $\rho(w_i^N, w^\star) < p/4$. We define the "abstract game" $(0, 0)$, where the planner has access to an oracle but, when queried, rather than giving information about the "true" $w^\star$, the oracle will simply return a uniformly random bit in $[p]$.

Lemma B.11 (A second change of measure)**.** *For any planner A with an oracle budget of x, we have that:*

$$P^A_{0,\hat{w}}(\hat{w} \in Z) = p^x P^A_{0,0}(\hat{w} \in Z)$$

Proof. As before, consider the set of histories $\mathcal{H}$, and let $Z_h = Z \cap \{H = h\}$. Writing out the importance ratio gives:

$$\rho = \frac{P^A_{0,w^\star}[Z_h]}{P^A_{(0,0)}[Z_h]} = \prod_i \frac{p(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p^{0,w^\star}(e_i|x_i, o_i)p^{(0,w^\star)}(y_i|x_i)}{p(x_i|x_{1:i-1}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p(o_i|x_{1:i}, o_{1:i-1}, e_{1:i-1}, y_{1:i-1})p^{(0,0)}(e_i|x_i, o_i)p^{(0,0)}(y_i|x_i)}$$

Again, as before, all the terms involving the planner will cancel, since they are conditioned on the same history and thus the planner will make the same decisions. Similarly, in both games the reward is deterministically 0 thus the $p^{(0,w^\star)}(y_i|x_i) = p^{(0,0)}(y_i|x_i)$. We are left with

$$\rho = \prod_{i=1}^n \frac{p^{(0,w^\star)}(E_i = e_i \mid x_i, o_i)}{p^{(0,0)}(E_i = e_i \mid x_i, o_i)}$$

Note that the top probability is deterministic (since the true expert is) and is only equal to 1 at most x times (recalling that x is the total number of oracle calls allowed). We are simply left with the (inverse of the) probability that the random oracle returns any given answer, which is $1/p$. Thus we end up with $\rho = p^x$. □

Lemma B.12 (Finding a bad $w^\star$ for planner A)**.** *For any abstract planner A there exists $w^\star \in W^\star$ such that*

$$P^A_{w^\star}(E_n^{w^\star}) \geq (9/10)^2$$

Proof. Note that, by Lemma B.9, it is sufficient to show that $P^A_{0,w^\star}((E_n^{w^\star})^c) \leq \frac{1}{10}$. Recall that for any $\hat{w} \in W^\star$ we have $(E_n^{\hat{w}})^c \subseteq \{\hat{w} \in Z\}$, where

$$Z \coloneqq \cup_{t \in [n \wedge (N-1)]} \left(\mathrm{W}_{\text{close}}(w^t_{L_t - 1}) \cup \mathrm{W}_{\text{close}}(w^t_{L_t}\right) \cup (\cup_{i \in [8]} \mathrm{W}_{\text{close}}(w_i^N))$$

By a union bound and Lemma B.10 we have that $|Z| \leq (2n+8)2^p \exp(-p/8)$. Since $W^\star = W \setminus \mathrm{W}_{\text{close}}(1) \setminus \mathrm{W}_{\text{close}}(-1)$, this also gives that $|W^\star| \geq 2^p(1 - 2\exp(-p/8))$. We pick $w^\star$ according to:

$$w^\star = \operatorname{argmin}_{\hat{w} \in W^\star} P^A_{0,w^\star}(\hat{w} \in Z).$$

Putting things together and using Lemma B.11 gives:

$$\begin{aligned}
&2^p(1 - 2\exp(-p/8))P^A_{0,w^\star}(w^\star \in Z) \leq |W^\star| P^A_{0,\hat{w}}(w^\star \in Z) \\
&\leq \sum_{\hat{w} \in W^\star} P^A_{0,\hat{w}}(\hat{w} \in Z) \leq \sum_{\hat{w} \in W} P^A_{0,\hat{w}}(\hat{w} \in Z) \leq \sum_{\hat{w} \in W} p^x P^A_{0,0}(\hat{w} \in Z) \\
&= p^x \sum_{\hat{w} \in W} P^A_{0,0}(\hat{w} \in Z) = p^x \sum_{\hat{w} \in W} \mathbb{E}^A_{0,0}\left[\mathbb{1}[\hat{w} \in Z]\right] = p^x \mathbb{E}^A_{0,0}\left[\sum_{\hat{w} \in W} \mathbb{1}[\hat{w} \in Z]\right] \\
&= p^x \mathbb{E}^A_{0,0}[|Z|] \leq p^x(2n+8)2^p \exp(-p/8)
\end{aligned}$$

Rearranging gives that

$$P^A_{0,w^\star}((E^{w^\star}_n)^c) \le P^A_{0,w^\star}(w^\star \in Z) \le \frac{(2n+8)2^p \exp(-p/8)p^x}{2^p(1-2\exp(-p/8)} \le 2(n+5)p^x \exp(-p/8) \le \frac{1}{10},$$

where the last line followed from our bound on n (Eq. (23)). □

We are now ready to prove Theorem B.7. In fact, there is not much left to do.

Proof (of Theorem B.7). Let the planner A be sound and have an expected query cost $\bar{N}$, and $w^\star$ the vector from the previous lemma. Then by Markov's inequality we have:

$$P^A_{(0,w^\star)}[N-1 \ge n] \le \frac{\bar{N}}{n}$$

Letting $E' = E^{w^\star}_n \cap \{N-1 < n\}$ we have

$$P^A_{0,w^\star}[E'] \ge (9/10)^2 - \frac{\bar{N}}{n}$$

Under event E', the output of the planner satisfies $\rho(w^N_i, w^\star) \ge p/4$ for all $i \in [8]$, so the reward at the end of the game is $R < (3/4)^9$. Combined with soundness this gives

$$\begin{aligned}\frac{1}{4} - 0.01 \le f_{w^\star}(()) - 0.01 \le \mathbb{E}^A_{w^\star}[R] &\le (\frac{3}{4})^9 + (1 - P^A_{0,w^\star}[E'])\frac{3}{4} \\ &\le (\frac{3}{4})^9 + (1-(9/10)^2)\frac{3}{4} + \frac{\bar{N}}{n}\frac{3}{4},\end{aligned}$$

which requires $\bar{N} > 0.02n$, namely

$$N > 0.02 \min\left\{\exp(p/8)p^{-x}/16 - 5, \left(\tfrac{1}{\varepsilon} - 1\right)/7.5\right\}$$

Lastly, note that when $x \le \frac{p}{16\log p}$ we have

$$\log n = \log(\exp(p/8)) - \log(p^x) = \frac{p}{8} - x\log p \ge \frac{p}{8} - \frac{p}{16\log p}\log p = \frac{p}{16},$$

thus $n = \Omega(\exp(p/16))$ and in particular $N = \Omega\{2^{p\wedge K}\}$. □

To prove Theorem B.3, the last thing to show is that a learner which solves the MDP can be used to solve CUBEGAME. This reduction follows exactly as in Section 4.8 of [WSG21]

C On π° linearity

This section shows that, when $\pi^\circ \neq \pi^\star$, q° can be linear with $d-$dimensional features yet these features do not realize π°-linearity.

The MDP is as follows: the states are arranged in a binary tree of length H. The action space is $\{\ell, \mathtt{r}\}$, corresponding to the ℓeft and $\mathtt{r}$ight actions. Transitions are deterministic. The reward for every ℓeft action is -1, the reward for every $\mathtt{r}$ight action is $+1$. See Figure 1.

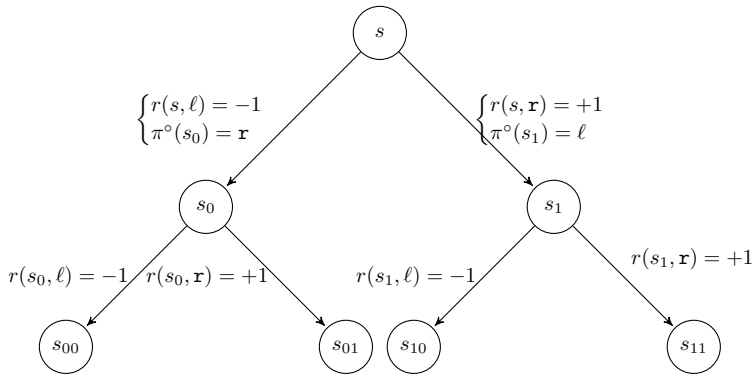

Figure 1: q°-linearity does not imply π°-linearity

Note that we can identify every state with the action sequence that led to it (with the starting state corresponding to the empty sequence). The policy π° is defined such that, if $s = (a_0, \ldots, \ell)$ then $\pi^\circ(s) = \mathtt{r}$ and otherwise if $s = (a_0, \ldots, \mathtt{r})$ then $\pi^\circ(s) = \ell$. Thus the policy will alternate the action taken at every step. This defines the q° function:

$$q^\circ(s,a) = \begin{cases} 0, & \text{if } h = 0 \bmod 2 \\ -1, & \text{if } h = 1 \bmod 2 \text{ and } a = \ell \\ +1, & \text{if } h = 1 \bmod 2 \text{ and } a = \mathtt{r} \end{cases}$$

Note that we can linearize this in one dimension via the features $\varphi^\circ(s,a) = q^\circ(s,a)$ and $\theta = 1$. However, these features do not linearly-realize π°: since $\theta > 1$ then the argmax at every odd horizon will always be the right action (since $\varphi^\circ(s,\mathtt{r}) = 1$ and $\varphi^\circ(s,\ell) = -1$).

Note that $\pi' = \operatorname{argmax}_a \{q^\circ(s,a)\}$, the greedy policy derived from q° is by definition linear with those features. For the special case where $\pi^\circ = \pi^\star$ then the greedy policy π' lines up with the policy π°, so we get linearity for free in that case.