Exponential Lower Bounds for Batch Reinforcement Learning: Batch RL can be Exponentially Harder than Online RL

Andrea Zanette [1]

Abstract

Several practical applications of reinforcement learning involve an agent learning from past data without the possibility of further exploration. Often these applications require us to 1) identify a near optimal policy or to 2) estimate the value of a target policy. For both tasks we derive *exponential* information-theoretic lower bounds in discounted infinite horizon MDPs with a linear function representation for the action value function even if 1) *realizability* holds, 2) the batch algorithm observes the exact reward and transition *functions*, and 3) the batch algorithm is given the *best* a priori data distribution for the problem class. Our work introduces a new 'oracle + batch algorithm' framework to prove lower bounds that hold for every distribution. The work shows an exponential separation between batch and online reinforcement learning.

1. Introduction

While the grand goal of reinforcement learning (RL) is to design fully autonomous agents capable of improving their performance over time by learning from past mistakes, oftentimes a *batch* approach — which predicts some quantity of interest using past observations only — is preferable. For example, past data may be available in large quantities and should not be disregarded by adopting a purely online method. In other applications, safety concerns require that the dataset be collected by a carefully monitored procedure, involving a human or a safe controller. In addition, batch algorithms are key tools to build complex online procedures.

These considerations motivate us to investigate whether there exists any *fundamental limitation* to using a batch approach for RL compared to online learning. Concretely,

we consider two classical batch RL problems: 1) the *off-policy evaluation* (OPE) problem, where the batch algorithm needs to predict the performance of a *target policy* and 2) the *best policy identification* (BPI) problem, where the batch algorithm needs to identify a near optimal policy.

Linear function approximation As many applications of RL require very large state and action spaces, the hope is that by leveraging strong prior knowledge, for example on the form of the optimal solution, we can learn the critical features of a Markov decision process (MDP) to solve the task at hand without probing the full state-action space. For the OPE problem, we assume that the action-value function Q^π of the target policy π can be written at any state action pair (s, a) as an inner product $\phi(s, a)^\top \theta$ between a known feature extractor ϕ and an unknown parameter θ that the learner seeks to identify. For the BPI problem, this representation condition applies to the action-value function $Q^\star$ of an optimal policy.

Quality of the dataset and assumptions Batch algorithms are limited by the quality and quantity of the available data: for example, in a tabular MDP, without adequate coverage of the area of the MDP that the optimal policy tends to visit, there is little hope that we can identify such policy or even predict its performance. However, given enough samples in each state-action pairs, it becomes easy to identify the optimal policy (Azar et al., 2012) or to evaluate the value of another (Yin et al., 2020). In addition, without any prior knowledge about the problem or information about the target policy, a uniform distribution over the state-action space is a priori optimal.

Very recently, some authors have derived exponential lower bounds for the off-policy evaluation problem with linear function approximations (Wang et al., 2020a; Amortila et al., 2020). They discover that even if a sampling distribution induces the best-conditioned covariance matrix, at least exponentially many samples are needed to estimate the value of a target policy reasonably well. As their hard instances are tabular MDPs (i.e., with small state-action spaces), certainly *there exists a better batch distribution* for their setting: the OPE problem in tabular MDPs is easily solvable by using a uniform distribution over the state-

[1]Institute for Computational and Mathematical Engineering, Stanford University, Stanford, USA. Correspondence to: Andrea Zanette <zanette@stanford.edu>.

actions coupled with the policy evaluation algorithm on the empirical model. Therefore, their works show that the assumption on the condition number of the covariance matrix is insufficient in batch RL. However, their works leave open the question of whether the OPE problem with linear value functions can *always* be solved by using a better sampling distribution that generates a dataset of polynomial size (in the horizon and feature dimension), leading to the following, more fundamental question in the context of batch RL with function approximation:

> #### What can a batch algorithm learn using the best a priori distribution for the problem class at hand?

In particular, we wonder whether there is any penalty in using an entirely batch approach in place of an online (and adaptive) method if a good dataset is provided to the batch algorithm. To answer this question for both the OPE and BPI problems in the strongest possible way, we consider an auxiliary process or *oracle* that provides the *batch algorithm* with the best data distribution for the task; together they form a *learning algorithm*. However, in contrast to fully online / adaptive algorithms, the oracle is not allowed to change its data acquisition strategy while information is being acquired. As we explain in Appendix A, this framework allows us to derive strong batch RL lower bounds, because they *will hold for every batch distribution*. Thanks to this framework, we can recover the concurrent lower bound by (Wang et al., 2020a) — which holds for one specific choice of the sampling distribution — in infinite horizon RL as a corollary.

Learning process and assumptions   We consider oracles that can decide on a set of strategic policies to gather data. Alternatively, the oracles can directly specify the state-actions where they want to observe the rewards and transitions (for example, with a simulator). In either case, the oracle selects a sampling strategy and the batch algorithm later receives a *dataset* of states, actions, rewards and transitions. Using the dataset, the batch algorithm makes a prediction, i.e., it estimates the value of a target policy (OPE problem) or returns a near optimal policy (BPI problem). We make two other assumptions that favor the learner:

- *Realizability*, i.e., there is no misspecification.

- *Exact feedback*: the exact reward and transition *function* is observed for each point in the dataset. This is equivalent to observing *infinite data*.

1.1. Contributions

With such strong assumptions and the freedom of selecting any state-action, identifying a near optimal policy or predicting the value of another should be easy tasks for the learner. For example, in tabular MDPs the oracle could prescribe one query in each state-action pair. This way, the reward and transition functions would become known everywhere, giving the batch algorithm complete knowledge of the MDP. For linear bandits with feature vectors spanning $\mathbb{R}^d$, we know that d queries along a basis suffice to identify the full bandit instance. For H horizon MDPs with linear representations, backward induction with dH queries and exact feedback precisely identifies the MDP at hand. In all these cases the oracle can find a good sampling distribution for the batch learner, and there is little or no advantage to using an oracle that adapts the query selection strategy as feedback is received. However, in infinite horizon problems the situation is quite different; we show that

1. there exists OPE and BPI problems where any batch algorithm must receive an *exponential* $\approx (\frac{1}{1-\gamma})^d$ dataset to return a good answer,

2. if the dataset does not originate from policy rollouts then the lower bounds hold even if the action-value function of *every policy* admits a linear representation,

3. there exist exponentially hard batch BPI problems (even under the best data distribution) which are easy to solve with online / adaptive algorithms, showing *an exponential separation between batch and online RL*,

4. there exist exponentially hard problems for infinite horizon batch RL which cannot arise in finite horizon problems, showing *exponential separation between finite and infinite horizon batch RL*.

As a corollary, our work recovers (Wang et al., 2020a)'s lower bound in infinite horizon RL and also shows that the classical globally optimal experimental design yields *provably bad* distributions for infinite horizon RL, making learning impossible even in the limit of infinite data; we discuss this in Appendix A.

We make the following *technical* contributions:

1. we introduce a new 'oracle + batch algorithm' framework to derive lower bounds for *every a priori distribution*; this automatically yields fixed distribution lower bounds (e.g., (Wang et al., 2020a)) as a special case,

2. we help formalize the hardness induced by the *deadly triad* (Sutton & Barto, 2018), i.e., the combination of off-policy learning, bootstrapping and function approximation; in particular, we explain that the *bootstrapping* problem is fundamentally different and potentially more severe than the *extrapolation* problem,

3. we present new classes of hard MDPs where critical MDP information is 'deferred — through bootstrapping — and hidden in an unknown and exponentially small region *in feature space*, too small to be covered by a batch dataset but easy to locate using an online algorithm.

1.2. Literature

Polynomial lower bounds are often derived to certify that a certain algorithm is sample efficient (Jiang & Li, 2016; Duan & Wang, 2020; Hao et al., 2020); in this work we are interested in exponential lower bounds. Divergence of dynamic programing algorithms with function approximation is well known (Baird, 1995; Tsitsiklis & Van Roy, 1996) and has prompted researchers to look for information-theoretic lower bounds for generic predictors (Chen & Jiang, 2019) and in presence of misspecification (Du et al., 2019). Concurrently, Weisz et al. (2020); Wang et al. (2020a) also show information-theoretic lower bounds highlighting the danger of *extrapolation*; for additional literature, please see app. B.

2. Preliminaries

Discounted infinite horizon MDPs (Puterman, 1994) are defined by a tuple $M = \langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}_s$ is the action space in state $s \in \mathcal{S}$ and $\mathcal{A} = \cup_{s \in \mathcal{S}} \{\mathcal{A}_s\}$ is the set of all state-dependent action spaces. We assume that for each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}_s$ there exists a measure $p(\cdot \mid s, a)$ representing the transition dynamics and a scalar reward function $r(s, a) \in [-1, 1]$. The discount factor γ is assumed to be in $(0, 1)$. We consider the discounted return of any policy to be $\in [-1, +1]$ (without loss of generality upon rescaling the reward function). A deterministic policy π maps every state $s \in \mathcal{S}$ to an action $a \in \mathcal{A}_s$. The value function of policy π in state $s \in \mathcal{S}$ is defined as $V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{x_t \sim \pi | x_0 = s} r(x_t, \pi(x_t))$; the limit always exists and it is finite under our assumptions as long as the expectation is well defined. The action-value function Q^{π} of policy π in (s, a) also exists and is defined as $Q^{\pi}(s, a) = r(s, a) + \sum_{t=1}^{\infty} \gamma^t \mathbb{E}_{x_t \sim \pi | (s, a)} r(x_t, \pi(x_t))$. An optimal policy $\pi^{\star}$, when it exists, is defined in every state as $\pi^{\star}(s) = \arg\max_{\pi} V^{\pi}(s)$ and the corresponding value function and action-value function are denoted with $V^{\star} = V^{\pi^{\star}}$ and $Q^{\star} = Q^{\pi^{\star}}$. We sometime add the subscript M to indicate that a certain quantity depends on the MDP M under consideration (e.g., we write $V_M^{\star}$). The Euclidean ball in $\mathbb{R}^d$ is defined as $\mathcal{B} = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$. The *Bellman optimality operator* $\mathcal{T}$ and the *Bellman evaluation*

operator $\mathcal{T}^{\pi}$ are mappings between action value functions:

$$(\mathcal{T}Q)(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim p(s, a)} \sup_{a' \in \mathcal{A}_{s'}} Q(s', a')$$

$$(\mathcal{T}^{\pi}Q)(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim p(s, a)} Q(s', \pi(s')).$$

3. Batch Reinforcement Learning

We first formally define the two *learning problems*, namely the task of returning a near optimal policy, also known as *best policy identification* (BPI) problem, and the task of predicting the value of a *target policy*, also known as *off-policy evaluation* (OPE) problem. Then, in the next four sub-sections we describe the learning process in more detail. The lower bounds that we later derive will hold for all algorithms of the form described in this section.

A *BPI problem* $(s^{\star}, \{M \in \mathcal{M}\})$ is defined by a starting state $s^{\star}$ and a class $\mathcal{M}$ of MDPs sharing the same state space, action space and discount factor. An *OPE problem* $(s^{\star}, \{(M, \pi_M) \mid M \in \mathcal{M}\})$ additionally requires us to identify one or more target policies π_M for each $M \in \mathcal{M}$.

Depending on the problem, the oracle selects a sampling strategy such that for every *problem instance* $(s^{\star}, M, \pi_M)$ (of an OPE problem) or $(s^{\star}, M)$ (of a BPI problem) the batch algorithm experiences a dataset $\mathcal{D}$ from M and uses it to return an accurate estimate $\widehat{Q}_{\mathcal{D}}$ of the action-value function of π_M at $s^{\star}$ (for the OPE problem) or a near optimal policy $\widehat{\pi}_{\mathcal{D}}^{\star}$ on M from $s^{\star}$ (for the BPI problem).

Protocol 1 BATCH RL WITH A STRATEGIC ORACLE

1: *(Input)* Oracle receives a batch problem and budget n
2: *(Query Selection)* Oracle chooses μ (def. 1) or T (def. 2).
3: *(Data Collection)* Batch algorithm receives the dataset $\mathcal{D}$
4: *(Output)* Batch algorithm returns $\widehat{Q}_{\mathcal{D}}$ or $\widehat{\pi}_{\mathcal{D}}^{\star}$

3.1. Step I: Input

The oracle receives *either* an OPE or a BPI problem together with a *query budget* $n \in \mathbb{N}$. The oracle has access to each instance of the problem it is given, and in particular it knows the state and action spaces $\mathcal{S}, \mathcal{A}$, the discount factor γ, and the full set of transition functions and reward functions for each MDP $M \in \mathcal{M}$, but it does not know which of these MDPs in $\mathcal{M}$ it will interact with or what the target policy will be. In addition, the oracle has full knowledge of the batch algorithm.

3.2. Step II: Query Selection

The purpose of the oracle is to help the batch algorithm by providing it with the best dataset for the specific problem at hand. To capture different mechanisms of data acquisition, we consider two methods to specify the *query set*, i.e., the set of state-action pairs where the oracle wants the batch

algorithm to observe the rewards and the transitions. In the first mechanism the oracle directly selects the state-actions; we place no restriction on the mechanism to obtain these queries as long as the sampling distribution is fixed for all MDPs in the class.

Definition 1 (Policy-Free Queries)**.** *A set μ of state-action pairs is said to be policy-free for an OPE or BPI problem if μ does not depend on the specific MDP instance $M \in \mathcal{M}$.*

Since the batch algorithm observes the exact reward and transition function at the selected query points, the number of policy-free queries is *the size of the support of the distribution μ.*

The second mechanism to collect data is by selecting deterministic[1] policies. The policies will generate random trajectories that the batch algorithm later observes. Since we are interested in what the agent can learn in the limit of infinite data (i.e., under exact feedback) we let the batch algorithm observe *all possible realizations* of such trajectories. With this aim, we define the state-action space reachable in c or less timesteps from s_0 using policy π as $\text{Reach}(s_0, \pi, c) =$

$$\{(s, a) \mid \exists t < c, \text{ s.t. } \mathbf{P}\left((s_t, a_t) = (s, a) \mid \pi, s_0\right) > 0\}$$

where (s_t, a_t) is the random state-action encountered at timestep t upon following π from s_0. We let the oracle select the best combination of trajectory lengths and number of distinct policies.

Definition 2 (Policy-Induced Queries)**.** *Consider an OPE or BPI problem and fix a set $T = \{(s_{0i}, \pi_i, c_i)\}_{i=1}^{\kappa}$ of triplets, each containing a starting state s_{0i}, a deterministic policy π_i and a trajectory length c_i such that $\sum_{i=1}^{\kappa} c_i \leq n$. Then the query set μ induced by T is defined as*

$$\mu \overset{def}{=} \bigcup_{(s_0, \pi, c) \in T} \text{Reach}(s_0, \pi, c).$$

The condition $\sum_{i=1}^{\kappa} c_i \leq n$ attempts to make the amount of information acquired using policy-free queries comparable to the policy-induced query method, although the latter generates $|\mu| \geq n$ if the dynamics are stochastic. In other words, the batch learner always observes at least n state-actions if these are induced by policies.

Remark 1. *A policy-induced query set is also policy-free whenever the dynamics of each MDP $M \in \mathcal{M}$ are the same.*

Remark 2. *When prescribing a set T the oracle has knowledge of the induced dataset μ for each choice of T (e.g., by having access to a connectivity graph of the MDP).*

---

[1]This is not a restriction: for any given stochastic policy the agent can sample an action from the distribution of actions in every state before deploying the policy.

While a policy-free query set may seem less constrained than a policy-induced query set, the latter may implicitly reveal additional information about the dynamics of the MDP whenever the induced trajectories are all different across different MDPs.

3.3. Step III: Data Collection

After the oracle has submitted the sampling strategy, the batch learner receives a dataset $\mathcal{D}$ which contains the *exact* reward and transition *function*, $r(s, a)$ and $p(s, a)$, from the MDP $M \in \mathcal{M}$ for each (s, a) in the query set μ.

3.4. Step IV: Output

The batch algorithm is finally required to make a prediction using the acquired dataset $\mathcal{D}$. For the OPE problem, the batch algorithm also receives the target policy π_M and it is required to output an estimator $\widehat{Q}_{\mathcal{D}}$ for the action-value function of the target policy $Q_M^{\pi_M}(s^\star, \cdot)$; for the BPI problem, it must output a near-optimal policy $\widehat{\pi}_{\mathcal{D}}^\star$ at $s^\star$.

3.5. Evaluation Criterion

The *oracle* and the *batch algorithm* together form a *learning algorithm*. This framework allows us to derive batch lower bounds in a strong form as they will hold for any data distribution. We say that the learning algorithm is (ϵ, δ)-*sound* for an OPE problem if for every instance $(s^\star, M, \pi_M)$ of the problem the returned estimator $\widehat{Q}_{\mathcal{D}}$ is accurate w.h.p.:

$$\mathbf{P}\left(\sup_{a \in \mathcal{A}_{s^\star}} |(Q_M^{\pi_M} - \widehat{Q}_{\mathcal{D}})(s^\star, \cdot)| < \epsilon\right) > 1 - \delta.$$

Similarly, we say that that the learning algorithm is (ϵ, δ)-*sound* for a BPI problem if for every instance $(s^\star, M)$ it holds that the returned policy $\widehat{\pi}_{\mathcal{D}}^\star$ is near optimal w.h.p.:

$$\mathbf{P}\left((V_M^\star - V_M^{\widehat{\pi}_{\mathcal{D}}^\star})(s^\star) < \epsilon\right) > 1 - \delta.$$

As the query set is always non-random and the batch algorithm experiences the exact reward and transition function, the only randomness lies in the possible randomization internal to the batch algorithm when it returns an answer.

3.6. Adaptive and Online Algorithms

Consider a policy-free mechanism. We say that a learning algorithm is *adaptive* if every time the oracle submits a state-action it receives the feedback from the environment and can use it to select the next state-action to query the MDP.

Likewise, consider a policy-induced mechanism. We say that a learning algorithm is acting *online* if every time the

oracle selects a policy and trajectory length, it can use the acquired feedback to select the next combination of policy and trajectory length (its position is reset in every episode).

4. Intuition

The mechanism that induces hardness for infinite horizon problems must be different than that in finite horizon: the constructions from Weisz et al. (2020); Wang et al. (2020b) rely on the *extrapolation* issue that compounds the errors multiplicatively. In our case, the reward and transition functions are observed exactly, so there is no error to extrapolate in the first place. Instead, *bootstrapping*, i.e., the fact that the value function in one state depends on the same value function at successor states (Sutton & Barto, 2018), is the root cause of hardness; here we provide some intuition.

While each of our theorems need a different construction, they all build on the intuition presented in this section; at a high level, bootstrapping can "erase" the information gained along certain directions in feature space.

Suppose that the oracle is trying to find a good policy-free query set for the off-policy problem on a class of MDPs with feature vectors $\phi(\cdot, \cdot)$ anywhere in the unit Euclidean ball $\mathcal{B}$. Denote with $(s_1, a_1), \ldots, (s_n, a_n)$ the state-actions chosen by the oracle and with $(s_1^+, a_1^+), \ldots, (s_n^+, a_n^+)$ the corresponding successor states and actions[2]. Intuitively, choosing a set $\phi(s_1, a_1), \ldots, \phi(s_n, a_n)$ of orthogonal feature vectors of maximal length seems to be the best the oracle can do because it gives rise to a covariance matrix $\Phi^\top \Phi$ (where Φ is described below) that is a multiple of the identity. In Fig. 1 (left) we represent the feature space of an RL problem where the learner can choose any feature vector in the Euclidean ball in $\mathbb{R}^2$; the agent's choice of the feature vectors $\phi_i = \phi(s_i, a_i)$ arise from globally optimal experimental design (Pukelsheim, 2006). Define

$$\Phi = \begin{bmatrix} \phi(s_1, a_1)^\top \\ \ldots \\ \phi(s_n, a_n)^\top \end{bmatrix}, r = \begin{bmatrix} r(s_1, a_1) \\ \ldots \\ r(s_n, a_n) \end{bmatrix}, \Phi^+ = \begin{bmatrix} \phi(s_1^+, a_1^+)^\top \\ \ldots \\ \phi(s_n^+, a_n^+)^\top \end{bmatrix}$$

After observing the reward and successor states, we can write down the local (i.e., at the state-actions chosen by the oracle) Bellman equations. If we leverage the known functional form of the action-value function, we can write the following linear system, whose solution (in terms of $\theta \in \mathbb{R}^d$ or action-value function $\Phi\theta$) the batch algorithm seeks to discover:

$$\Phi\theta = r + \gamma\Phi^+\theta \quad \longrightarrow \quad (\Phi - \gamma\Phi^+)\theta = r. \quad (1)$$

Unlike machine learning or bandit problems where the

agent directly observes a response from $\phi(s_i, a_i)$, bootstrapping the future returns makes the agent observe a response (i.e., the reward) corresponding not to $\phi(s_i, a_i)$ but to $\phi(s_i, a_i) - \gamma\phi(s_i^+, a_i^+)$, which we call "effective feature vector", see Eq. (1). Unfortunately, $\phi(s_i^+, a_i^+)$ can act adversarially. As an example, consider Fig. 1 (left) in $\mathbb{R}^2$ where the orthogonal feature vectors chosen by the agent are projected along the vertical direction by bootstrapping: the net effect is that the agent only learns along one axis. Mathematically, when this happens the final system of linear equations in Eq. (1) admits infinitely-many solutions since $\Phi - \gamma\Phi^+$ is rank deficient[3]. From a reinforcement learning perspective, the local Bellman equations in Eq. (1) admit multiple linear value functions as possible solutions. This happens because the rewards and transitions that the batch algorithm has received could have originated from any MDP whose action-value function is a solution to the local Bellman equations.

Unfortunately, choosing more feature vectors does not easily solve the problem (Fig. 1 (middle)). To acquire information in all directions (i.e., to ensure that $\Phi - \gamma\Phi^+$ in Eq. (1) is full rank) the oracle must probe the small spherical cap in Fig. 1 (right). Which spherical cap to probe is unknown ahead of sampling (as the target policy or the optimal policy are unknown), and thus the oracle needs to probe all of them if it wants the batch algorithm to confidently solve the problem. Using the fact that there are exponentially many spherical caps in an Euclidean ball in $\mathbb{R}^d$, we conclude.

Exponential separation with online learning An online algorithm can detect that the next-state feature matrix Φ^+ is acting adversarially. In addition, Φ^+ reveals the location of the spherical cap in Fig. 1 (right). The online algorithm can then probe the spherical cap to ensure that the linear system in Eq. (1) is full rank.

5. Exponential Lower Bounds

We define the *query complexity to (ϵ, δ)-soundness of a problem* (OPE or BPI) to be the minimum value for n (as in Definitions 1 and 2) such that there exists a (ϵ, δ)-sound learning algorithm for that problem. In particular, the query complexity depends on the MDP class $\mathcal{M}$ and can be different for the OPE and BPI tasks and for the policy-free and policy-induced query mechanism. Our lower bounds on the query complexity are *significantly stronger* than typical sample complexity lower bounds: they are really lower bounds on the *size of the support* of the distribution μ and automatically imply *infinite sample complexity* lower bounds since the batch algorithm already observes the ex-

---

[2] Assume deterministic successor states; the actions in the successor states are determined by the target policy (OPE problem) or by an arg max function (BPI problem).

[3] This follows from the fundamental theorem of linear algebra. In particular, since the row space of $\Phi - \gamma\Phi^+$ does not span $\mathbb{R}^d$, the nullspace of $\Phi - \gamma\Phi^+$ must have dimension at least 1.
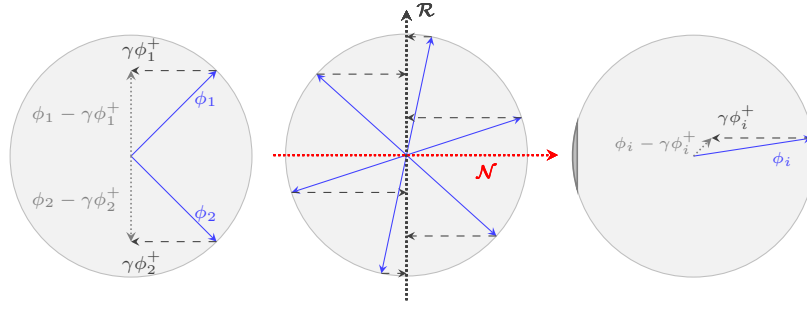
Figure 1. *Left*: the orthogonal blue vectors ϕ_1, ϕ_2 represent the choice of the algorithm, while ϕ_1^+, ϕ_2^+ are the corresponding successor features. The learner thus acquires information only along the vertical direction. *Middle*: in the figure $\mathcal{R} = \text{Range}(\Phi^\top - \gamma(\Phi^+)^\top)$ and $\mathcal{N} = \text{Null}(\Phi - \gamma\Phi^+)$. *Right*: in this case the next-state discounted feature $\gamma\phi_i^+$ cannot project ϕ_i onto the vertical plane because of the discount factor $\gamma < 1$.

act reward and transition functions where μ is supported.

The lower bounds are expressed in terms of the regularized incomplete beta function $I_x(a, b) = B(x, a, b)/B(1, a, b)$ where $B(x, a, b)$ is the incomplete beta function for some positive real numbers a, b and $x \in [0, 1]$; for the precise definitions, please see Appendix D.1. For brevity, define $\mathcal{N}(\gamma, d) = I_{1-\gamma^2}^{-1}\left(\frac{d-1}{2}, \frac{1}{2}\right)$. Corollary 2 ensures $\mathcal{N}(\gamma, d)$ is exponential in the dimension d for $d \geq 5$ (here the $\approx$ symbol highlights an approximate dependence without a formal definition):

$$\mathcal{N}(\gamma, d) > 2^{-d}\mathcal{N}(\gamma, d) \geq \gamma\sqrt{d}\left(\frac{1}{2^{2.5}(1-\gamma)}\right)^{\frac{d-1}{2}}$$
$$\approx \left(\frac{1}{1-\gamma}\right)^d.$$

5.1. Realizability Assumptions

Unless additional assumptions are made regarding the MDP class $\mathcal{M}$ that defines the OPE and BPI problems, the query complexity of a reasonably sound learner is exactly the size of the state and action space (we assume exact feedback). One hopes that by restricting the MDP class $\mathcal{M}$, the query complexity of the OPE and BPI problems can be brought down to a more manageable level, in particular, independent of the state-action spaces.

We make one of the following three assumptions (only one assumption will hold at any given time, depending on the theorem); two are known as realizability, and the third is strictly stronger than the first two. The first concerns the OPE problem; $\mathcal{B}$ is the unit Euclidean ball.

Assumption 1 (Q^π is Realizable). *Given an OPE problem $(s^\star, \{(M, \pi_M), M \in \mathcal{M}\})$, there exists a d-dimensional map $\phi(\cdot, \cdot)$ such that $\|\phi(\cdot, \cdot)\|_2 \leq 1$ and for any $M \in \mathcal{M}$ the action-value function of the target policy π_M satisfies $\forall(s, a), Q_M^{\pi_M}(s, a) = \phi(s, a)^\top \theta_M$ for some $\theta_M \in \mathcal{B}$.*

For the BPI problem the representation condition applies to the action-value function of an optimal policy.

Assumption 2 ($Q^\star$ is Realizable). *Given a BPI problem $(s^\star, \{M \in \mathcal{M}\})$, there exists a d-dimensional feature map $\phi(\cdot, \cdot)$ such that $\|\phi(\cdot, \cdot)\|_2 \leq 1$ and for any $M \in \mathcal{M}$ there exists $\theta_M^\star \in \mathcal{B}$ such that the optimal action-value function is linear: $\forall(s, a), Q_M^\star(s, a) = \phi(s, a)^\top \theta_M^\star$.*

A stronger assumption we make is that the action-value function of *every* policy has a linear representation.

Assumption 3 (Q^π is Realizable for every Policy). *Given a BPI problem $(s^\star, \{M \in \mathcal{M}\})$ or an OPE problem $(s^\star, \{(M, \pi_M), M \in \mathcal{M}\})$, there exists a d-dimensional feature map $\phi(\cdot, \cdot)$ such that $\|\phi(\cdot, \cdot)\|_2 \leq 1$ and for any MDP $M \in \mathcal{M}$ the value of every policy π satisfies $\forall(s, a), Q_M^\pi(s, a) = \phi(s, a)^\top \theta_M^\pi$ for some $\theta_M^\pi \in \mathcal{B}$.*

The learners that we consider are aware of these assumptions because they can examine each MDP in the class $\mathcal{M}$ they receive (see Section 3.1).

5.2. Off-Policy Evaluation

The first result of this work is contained in the following lower bound for the OPE problem; since all MDPs in $\mathcal{M}$ share the same dynamics, the oracle has full knowledge of the actions it needs to take to visit any state-action it desires.

Theorem 1 (OPE Policy-Induced Lower Bound). *There exists an OPE problem $(s^\star, \{(M, \pi_M), M \in \mathcal{M}\})$ satisfying Assumption 1 such that its policy-induced query complexity to $(1, 1/2)$-soundness is at least $\mathcal{N}(\gamma, d)$.*

It is useful to compare the *form* of the above lower bound with that of some concurrent results for the off-policy evaluation problem: for example, (Wang et al., 2020a; Amortila et al., 2020) show that *there exist* a sampling distribution μ

(that induces the best-conditioned covariance matrix), a target policy π and an MDP class $\widetilde{\mathcal{M}}$, each satisfying certain properties, such that the best estimator on the most difficult MDP in $\widetilde{\mathcal{M}}$ performs poorly if less than exponentially many samples are used. However, in their case there exists a better batch distribution of polynomial size that solves the problem. Our instances are instead much harder, as they command an exponential dataset even for the *best* distribution for the task: since the oracle can prescribe any data distribution, we can claim that *for all* distributions of polynomial size we can find an MDP subclass $\widetilde{\mathcal{M}} \subseteq \mathcal{M}$ and a target policy such that all MDPs in $\widetilde{\mathcal{M}}$ generate similar datasets but the value of the target policy is very different on these MDPs in $\widetilde{\mathcal{M}}$, giving lower bounds in a *stronger form*.

5.3. Best Policy Identification

As in Theorem 1, in the next lower bound the oracle knows the set μ induced by any choice of T (see Definition 2).

Theorem 2 (BPI Policy-Induced Lower Bound). *There exists a BPI problem* $(s^\star, \{M \in \mathcal{M}\})$ *satisfying Assumption 2 with features in dimension $d+1$ such that its policy-induced query complexity to $(1/2, 1/2)$-soundness is at least $2^{-d}\mathcal{N}(\gamma, d)$.*

Hard BPI problems for adaptive and online algorithms are given by Weisz et al. (2020). Clearly, their construction can be embedded in an infinite horizon MDP, giving a lower bound under their assumptions. However, our framework allows the learner to observe the *exact* reward and transition function, which is equivalent to having infinite data at the selected state-actions. In such case, the construction of Weisz et al. (2020) no longer gives rise to hard instances. In particular, the BPI problem with our assumptions becomes straightforward in finite horizon, showing an exponential separation between finite and infinite horizon batch RL.

5.4. Lower Bounds with Stronger Representation

We wonder what is achievable if the oracle can specify the queries anywhere in the state-action space without the restriction imposed by following policies (i.e., using a policy-free query set). Under this assumption the situation gets surprisingly worse, as the lower bounds now hold even if the action-value function of *every* policy is linear.

Theorem 3 (Policy-Free Lower Bounds). *There exist an OPE problem $(s^\star, \{(M, \pi_M), M \in \mathcal{M}\})$ and a BPI problem $(s^\star, \{M \in \mathcal{M}\})$, which satisfy Assumption 3 and share the same $s^\star$ and $\mathcal{M}$, such that their policy-free query complexity to $(1, 1/2)$-soundness is at least $\mathcal{N}(\gamma, d)$. In addition, an MDP class $\mathcal{M}$ that yields the lower bounds (but with $\frac{1}{\sqrt{d}}, 1/2)$-soundness) can be constucted with at most $|\mathcal{A}| = 2d$ actions.*

Contrasting Theorem 3 with Theorems 1 and 2 shows that there is a sharp distinction in what can be achieved depending on the *assumptions on the mechanism* that generates the batch dataset, a distinction which is absent in tabular RL.

6. Exponential Separation with Online Learning

Theorem 4 (Exponential Separation with Online Learning). *Consider the same BPI problem as in Theorem 2. There exists an online algorithm that can identify an optimal policy with probability one by observing trajectories of length one with exact feedback from $d+1$ distinct policies from $s^\star$.*

This result shows *exponential separation with online learning* even when the best batch distribution is used. The key information is hidden in an exponentially small area of the feature space whose position is a priori unknown. This region is too small to be covered by a batch dataset. However, an online algorithm can learn where this information is hidden and then probe such region as we show in Section 7.

A similar exponential separation with online learning is available for the BPI problem in Theorem 3 (see the end of the proof in the appendix). In addition, assuming access to a generative model an even stronger result is readily available in the literature using the Least-Square Policy Iteration (LSPI) algorithm (Lagoudakis & Parr, 2003): with a generative model (Lattimore et al., 2020) show that with probability at least $1 - \delta$, LSPI finds an ϵ optimal policy *for any BPI problem $(s^\star, \{M \in \mathcal{M}\})$ that satisfies Assumption 3* using at most $\text{poly}(d, \frac{1}{1-\gamma}, \frac{1}{\epsilon}, \ln \frac{1}{\delta})$ samples. However, LSPI is non-batch as it relies on Monte-Carlo rollouts at every iteration. Our result thus shows that it is not possible to start from a batch dataset obtained from a generative model and run LSPI (or any algorithm) successfully without acquiring further data even if a strong representation holds.

7. Proof Sketch of Theorem 2 and Theorem 4

We first describe the state and action space which are fixed across all MDPs in the class $\mathcal{M}$. Then we describe the instance-dependent reward and transition functions. Finally we prove the theorems.

At a high level, each MDP contains a two-armed bandit instance in $s^\star$ (the starting state). There, the learner has two choices: 1) take the special action $a^\star$ that gives a known return or 2) take any other action in the positive orthant $\mathcal{B}^+ = \{x \in \mathcal{B} \mid x_i \geq 0, i \in [d]\}$, see Fig. 2.

Crucially, on $\mathcal{B}^+$ the reward function is almost everywhere zero except inside the exponentially small spherical cap

$\mathcal{C}_\gamma(w) = \{\|x\|_2 \leq 1 \mid \frac{x^\top w}{\|w\|_2} \geq \gamma\}$ (Fig. 2) for some $w \in \mathcal{B}$. Unless the oracle prescribes an action inside $\mathcal{C}_\gamma(w)$, *the batch algorithm only observes a zero reward function* and is unable to distinguish different MDPs using this information.

7.1. Setup: State-Action Space and Feature Extractor

State space   The state space $\mathcal{S} = \{s^\star, \overline{s}, s^\dagger\}$ consists of a start state $s^\star$, an intermediate state $\overline{s}$ and a terminal state $s^\dagger$.

Action space   In the starting state $s^\star$ the special action $a^\star$ is available in addition to any action $a \in \mathcal{B}^+$. In the intermediate state $\overline{s}$ any action in $\mathcal{B}^+$ is available but not $a^\star$. Finally, in the terminal state $s^\dagger$ only $\vec{0} \in \mathcal{B}^+$ is available. Mathematically $\mathcal{A}_{s^\star} = \mathcal{B}^+ \cup \{a^\star\}$; $\mathcal{A}_{\overline{s}} = \mathcal{B}^+$; $\mathcal{A}_{s^\dagger} = \{\vec{0}\}$.

Feature map   The feature map only depends on the action:

$$\forall s \in \mathcal{S}: \ \phi(s, a) = \begin{cases} [\vec{0}, 1] & \text{if } a = a^\star \quad \text{(only available in } s^\star\text{)} \\ [a, 0] & \text{if } a \in \mathcal{B}^+. \end{cases}$$

7.2. Setup: MDP-specific Rewards and Transitions

Every MDP $M \in \mathcal{M}$ is identified by a vector w in the outer portion of the positive orthant $\partial\mathcal{B}^+ = \{x \in \mathcal{B}^+ \mid \|x\|_2 = 1\}$ and by a $\pm$ sign, and is denoted with $M_{w,+}$ or $M_{w,-}$.

Transition function   The transition function p_w depends on the vector w that identifies each MDP in the class, *but not on the sign $+$ or $-$*. Fix the MDP by fixing $w \in \partial\mathcal{B}^+$ (two MDPs correspond to a given choice of w). If the agent plays the special action $a^\star$, which is only available in the starting state $s^\star$, it transitions with probability one to the terminal state $s^\dagger$. If the agent plays $a \neq a^\star$, the transition function only depends on the action a (and not on the current state s) and the successor state is $\overline{s}$ with some probability, and is otherwise the absorbing state $s^\dagger$.

Mathematically, if $a = a^\star$ then $p_w(s^\dagger \mid (s^\star, a^\star)) = 1$ and
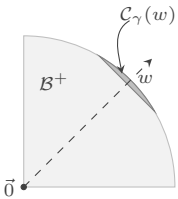

Figure 2. Action space $\mathcal{B}^+$. On $M_{w,-}$ the reward function is zero for any action $a \in \mathcal{B}^+$; on $M_{w,+}$ it is nonzero only inside $\mathcal{C}_\gamma(w)$.

if conversely $a \in \mathcal{B}^+$:

$$\begin{cases} p_w(\overline{s} \mid s, a) & = \min\{(1/\gamma)a^\top w, 1\}, \\ p_w(s^\dagger \mid s, a) & = 1 - p_w(\overline{s} \mid s, a). \end{cases} \tag{2}$$

The definition implies that the successor state is always either $\overline{s}$ or the terminal state $s^\dagger$.

Reward function   The reward function $r_{w,\pm}$ depends on both the vector $w \in \partial\mathcal{B}^+$ and on the sign $+$ or $-$ that identifyies the MDP. It is always $\frac{1}{2}$ if the special action $a^\star$ is taken and otherwise it is everywhere 0 on $M_{w,-}$ or is positive only in the spherical cap on $M_{w,+}$. Mathematically:

$$\text{on } M_{w,+}: \ r_{w,+}(s, a) \overset{def}{=} \begin{cases} \frac{1}{2} & \text{if } (s, a) = (s^\star, a^\star) \\ \max\{a^\top w - \gamma, 0\} & \text{otherwise,} \end{cases}$$

$$\text{on } M_{w,-}: \ r_{w,-}(s, a) \overset{def}{=} \begin{cases} \frac{1}{2} & \text{if } (s, a) = (s^\star, a^\star) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

7.3. Proof Sketch of Theorem 2 (Batch Lower Bound)

The steps for the proof are the following: we show that 1) $Q^\star$ is linear 2) policy-induced queries are also policy-free for this problem 3) using less than exponentially many queries ensures that at least one spherical cap $\mathcal{C}_\gamma(\widetilde{w})$ is not probed 4) the corresponding MDPs $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$ look the same outside the spherical cap $\mathcal{C}_\gamma(\widetilde{w})$ 5) the agent does not have enough information to distinguish $M_{\widetilde{w},+}$ from $M_{\widetilde{w},-}$.

Realizability   By inspection we can verify realizability.

Lemma 1 ($Q^\star$ is Realizable; Lemma 8 in appendix). *For any $w \in \partial\mathcal{B}^+$ let $Q^\star_{w,+}$ and $Q^\star_{w,-}$ be the optimal $Q^\star$ values on $M_{w,+}$ and $M_{w,-}$, respectively. It holds that*

$$\forall(s, a), \quad \begin{cases} Q^\star_{w,+}(s, a) = \phi(s, a)^\top[w, \frac{1}{2}] & \text{on } M_{w,+}, \\ Q^\star_{w,-}(s, a) = \phi(s, a)^\top[\vec{0}, \frac{1}{2}] & \text{on } M_{w,-}. \end{cases}$$

Policy-free vs policy-induced queries   Notice that although the dynamics are different for different w's (that identify the MDP), any set T (Definition 2) induces the same set μ of state-actions (possibly with the exception of $(s^\star, a^\star)$ and $(s^\dagger, \vec{0})$) regardless of the vector w and the sign $\pm$. We can therefore consider the case that the oracle has chosen a policy-free query set $\mu = \{(s_i, a_i)\}_{i=1}^n \cup \{(s^\star, a^\star), (s^\dagger, \vec{0})\}$.

Existence of exponentially many spherical caps

Lemma 2 (Follows from Lemma 5 in appendix). *Assume that less than $2^{-d}\mathcal{N}(\gamma, d)$ actions $a_1, \ldots, a_n$ on $\mathcal{B}^+$ are selected. Then there exists a spherical cap $\mathcal{C}_\gamma(\widetilde{w})$ that no action has probed, i.e., $\exists \widetilde{w} \in \partial\mathcal{B}^+ s.t. \forall i \in [n], a_i \notin \mathcal{C}_\gamma(\widetilde{w})$.*

In this case, the agent does not have any information originating from inside the dark gray spherical cap in Fig. 2.

$M_{w,+}$, $M_{w,-}$ are identical outside of the spherical cap
If $n < 2^{-d}\mathcal{N}(\gamma, d)$, consider $\widetilde{w}$ given by the above lemma and the two associated MDPs $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$. Notice that the transition function (Eq. (2)) is by construction identical on $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$ while the reward function (Eq. (3)) is zero at any action $a \notin \mathcal{C}_\gamma(\widetilde{w})$ selected by the oracle on both $M_{w,+}, M_{w,-}$. Then for any $(s,a) \in \{(s_i, a_i)\}_{i=1}^n$ that the batch algorithm receives it holds that

$$r_{\widetilde{w},+}(s,a) = r_{\widetilde{w},-}(s,a) \quad \text{and} \quad p_{\widetilde{w},+}(s,a) = p_{\widetilde{w},-}(s,a).$$

Batch algorithm does not have enough information
The above equation implies that the transitions and the rewards in the dataset could have originated from either $M_{\widetilde{w},+}$ or $M_{\widetilde{w},-}$. In $s^\star$, the batch algorithm has two choices to determine the policy $\widehat{\pi}_\mathcal{D}^\star$ to return: choose action $a^\star$ and get a total return of $\frac{1}{2}$ or choose an action $a \neq a^\star$. The second choice is $\frac{1}{2}$-suboptimal on $M_{\widetilde{w},-}$, while the first is at least $\frac{1}{2}$-suboptimal on $M_{\widetilde{w},+}$. At best, the batch algorithm can randomize between the two, showing the result.

7.4. Proof Sketch of Theorem 4 (Online Upper Bound)

It remains to exhibit an online algorithm that can solve every problem instance from this MDP class using $d+1$ queries. The algorithm proceeds as follows: 1) it tries to locate the position of the spherical cap by learning the vector w and 2) it probes the spherical cap to learn the $\pm$ sign of the MDP, precisely identifying the MDP.

Identifying the position w of the spherical cap Consider the following adaptive algorithm that submits policy-induced queries. The algorithm first plays the actions $\gamma e_1, \ldots, \gamma e_d$ in $s^\star$ where e_i is the vector of all zeros and 1 in position i (these are d policies that generate trajectories of length one where γe_i is the only action).

Upon receiving the transition functions $p_w(\overline{s} \mid s^\star, \gamma e_i) = \min\{\frac{1}{\gamma}(\gamma e_i)^\top w, 1\} = e_i^\top w$ for all $i \in [d]$ (see Eq. (2)), the agent can determine each component of the vector w. By construction, w identifies the spherical cap $\mathcal{C}_\gamma(w)$.

Identifying the sign $\pm$ of the MDP Next, the algorithm plays the state-action $(s^\star, w)$ to probe the spherical cap and observe the reward ($1 - \gamma$ on $M_{w,+}$ and 0 on $M_{w,-}$) which identifies the sign of the MDP. Since the MDP is now precisely identified, the agent can predict the value of any policy, and in particular, it can return the optimal policy.

8. Discussion

This work presents exponential lower bounds for batch RL. In general, models are never correct, observations are noisy, and a batch algorithm needs to return an answer using whatever dataset is available; clearly the lower bounds

continue to hold in these more general settings as much as they do when using more general predictors (like neural networks) which contain the linear setting as a special case. As these hard instances can only arise in infinite horizon settings, there is an exponential separation between finite and infinite horizon batch RL.

The strength of our results arise from the 'oracle + batch algorithm' protocol which allows us to derive lower bounds for *every* a priori data distribution; as a special case, we recover the concurrent lower bound of (Wang et al., 2020a) for the infinite horizon setting. We highlight that our lower bounds always imply an infinite sample complexity.

Beyond the exponential lower bounds, an important result is that online exploration may be required to achieve polynomial sample efficiency on certain RL problems. This is surprising, because online RL has the additional exploration burden compared to batch RL with a good dataset.

Finally, this work helps formalize some of the dangers of the deadly triad, which has long been known to cause algorithmic instabilities and divergence of dynamic programming algorithms. In a sense, the bootstrapping problem is for infinite horizon what the extrapolation problem is for finite horizon MDPs (and finite-steps algorithms), but unlike extrapolation, it cannot be mitigated by adding more samples.

Acknowledgment

References

Agarwal, A., Henaff, M., Kakade, S., and Sun, W. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *arXiv preprint arXiv:2007.08459*, 2020a.

Agarwal, A., Kakade, S., and Yang, L. F. Model-based reinforcement learning with a generative model is minimax optimal. In *Conference on Learning Theory*, pp. 67–83, 2020b.

Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. Optimality and approximation with policy gradient methods in markov decision processes. In *Conference on Learning Theory*, pp. 64–66, 2020c.

Amortila, P., Jiang, N., and Xie, T. A variant of the wang-

foster-kakade lower bound for the discounted setting, 2020.

Antos, A., Szepesvári, C., and Munos, R. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

Ayoub, A., Jia, Z., Szepesvari, C., Wang, M., and Yang, L. F. Model-based reinforcement learning with value-targeted regression. *arXiv preprint arXiv:2006.01107*, 2020.

Azar, M., Munos, R., and Kappen, H. J. On the sample complexity of reinforcement learning with a generative model. In *International Conference on Machine Learning (ICML)*, 2012.

Baird, L. Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning (ICML)*. 1995.

Bradtke, S. J. and Barto, A. G. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1-3):33–57, 1996.

Chen, J. and Jiang, N. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pp. 1042–1051, 2019.

Cui, Q. and Yang, L. F. Is plug-in solver sample-efficient for feature-based reinforcement learning? *arXiv preprint arXiv:2010.05673*, 2020.

Du, S. S., Kakade, S. M., Wang, R., and Yang, L. F. Is a good representation sufficient for sample efficient reinforcement learning? *arXiv preprint arXiv:1910.03016*, 2019.

Du, S. S., Lee, J. D., Mahajan, G., and Wang, R. Agnostic q-learning with function approximation in deterministic systems: Tight bounds on approximation error and sample complexity, 2020.

Duan, Y. and Wang, M. Minimax-optimal off-policy evaluation with linear function approximation. *arXiv preprint arXiv:2002.09516*, 2020.

Hao, B., Duan, Y., Lattimore, T., Szepesvári, C., and Wang, M. Sparse feature selection makes batch reinforcement learning more sample efficient, 2020.

Jiang, N. and Li, L. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pp. 652–661. PMLR, 2016.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. Contextual decision processes with low Bellman rank are PAC-learnable. In Precup, D.

and Teh, Y. W. (eds.), *International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1704–1713, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/jiang17c.html.

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, 2020.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of machine learning research*, 4(Dec):1107–1149, 2003.

Lattimore, T. and Szepesvári, C. *Bandit Algorithms*. Cambridge University Press, 2020.

Lattimore, T., Szepesvari, C., and Weisz, G. Learning with good feature representations in bandits and in RL with a generative model. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5662–5670. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/lattimore20a.html.

Lazaric, A., Ghavamzadeh, M., and Munos, R. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13(Oct):3041–3074, 2012.

Li, G., Wei, Y., Chi, Y., Gu, Y., and Chen, Y. Breaking the sample size barrier in model-based reinforcement learning with a generative model. *arXiv preprint arXiv:2005.12900*, 2020.

Li, L., Munos, R., and Szepesvári, C. Toward minimax off-policy value estimation. 2015.

Li, S. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics and Statistics*, 4(1):66–70, 2011.

Liu, Q., Li, L., Tang, Z., and Zhou, D. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, pp. 5356–5366, 2018.

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*, 2020.

Marjani, A. A. and Proutiere, A. Best policy identification in discounted mdps: Problem-specific sample complexity. *arXiv preprint arXiv:2009.13405*, 2020.

Munos, R. Error bounds for approximate policy iteration. In *ICML*, volume 3, pp. 560–567, 2003.

Munos, R. Error bounds for approximate value iteration. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2005.

Munos, R. and Szepesvári, C. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

Precup, D. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, pp. 80, 2000.

Pukelsheim, F. *Optimal design of experiments*. SIAM, 2006.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1994. ISBN 0471619779.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT Press, 2018.

Thomas, P. and Brunskill, E. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pp. 2139–2148, 2016.

Tsitsiklis, J. N. and Van Roy, B. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1-3):59–94, 1996.

Wang, R., Foster, D. P., and Kakade, S. M. What are the statistical limits of offline rl with linear function approximation? *arXiv preprint arXiv:2010.11895*, 2020a.

Wang, R., Salakhutdinov, R., and Yang, L. F. Provably efficient reinforcement learning with general value function approximation, 2020b.

Weisz, G., Amortila, P., and Szepesvári, C. Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions. *arXiv preprint arXiv:2010.01374*, 2020.

Wen, Z. and Van Roy, B. Efficient exploration and value function generalization in deterministic systems. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Xie, T. and Jiang, N. Q* approximation schemes for batch reinforcement learning: A theoretical comparison. volume 124 of *Proceedings of Machine Learning Research*, pp. 550–559, Virtual, 03–06 Aug 2020a. PMLR. URL http://proceedings.mlr.press/v124/xie20a.html.

Xie, T. and Jiang, N. Batch value-function approximation with only realizability. *arXiv preprint arXiv:2008.04990*, 2020b.

Xie, T., Ma, Y., and Wang, Y.-X. Towards optimal off-policy evaluation for reinforcement learning with marginalized importance sampling. In *Advances in Neural Information Processing Systems*, pp. 9668–9678, 2019.

Yang, L. F. and Wang, M. Reinforcement leaning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning (ICML)*, 2020.

Yin, M., Bai, Y., and Wang, Y.-X. Near optimal provable uniform convergence in off-policy evaluation for reinforcement learning. *arXiv preprint arXiv:2007.03760*, 2020.

Zanette, A., Brunskill, E., and J. Kochenderfer, M. Almost horizon-free structure-aware best policy identification with a generative model. In *Advances in Neural Information Processing Systems*, 2019a.

Zanette, A., Lazaric, A., J. Kochenderfer, M., and Brunskill, E. Limiting extrapolation in linear approximate value iteration. In *Advances in Neural Information Processing Systems*, 2019b.

Zanette, A., Brandfonbrener, D., Pirotta, M., and Lazaric, A. Frequentist regret bounds for randomized least-squares value iteration. In *AISTATS*, 2020a.

Zanette, A., Lazaric, A., Kochenderfer, M., and Brunskill, E. Learning near optimal policies with low inherent bellman error. In *International Conference on Machine Learning (ICML)*, 2020b.

Zanette, A., Lazaric, A., Kochenderfer, M. J., and Brunskill, E. Provably efficient reward-agnostic navigation with linear value iteration. In *Advances in Neural Information Processing Systems*, 2020c.

Zhou, D., He, J., and Gu, Q. Provably efficient reinforcement learning for discounted mdps with feature mapping. *arXiv preprint arXiv:2006.13165*, 2020.

A. Benefits of the Setup

One of the technical innovation of this work lies in the 'oracle + batch algorithm' protocol, which allows us to obtain lower bounds that hold for *every* distribution that the oracle can choose; this automatically yields lower bounds for fixed distributions as a special case. The key idea is that if the batch algorithm cannot return a good answer for *any* distribution chosen by the oracle, it certainly cannot return a good answer for an a priori fixed distribution, otherwise the oracle would have chosen it! To formally highlight the strength of the oracle setup, we derive (Wang et al., 2020a)'s lower bound (theorem 4.1 in their paper) for infinite horizon RL as a special case and with an additional remark about infinite data.

Corollary 1 (Well Conditioned Covariance Matrix is Insufficient). *There exist an MDP class $\mathcal{M}$ and a feature extractor ϕ that satisfy Assumption 3 (Q^π is Realizable for every Policy) together with a target policy π and a distribution $\mu = \{(s_i, a_i)\}_{i=1}^n$ that induces a covariance matrix*

$$\frac{1}{d}I = \frac{1}{|\mu|} \sum_{(s,a) \in \mu} \phi(s,a)\phi(s,a)^\top$$

such that no algorithm can predict the value of the target policy π with probability $> \frac{1}{2}$ and additive error < 1 (or return a policy with suboptimality < 1) even in the limit of infinite data (i.e., sampled rewards and transitions) generated from μ.

Proof. Consider the MDP class $\mathcal{M}$ described in the proof of theorem 4. Since any feature vector in the unit Euclidean ball is available, simply choose a distribution μ that samples the feature vectors $e_1, \ldots, e_d$ (concretely, we can choose $\mu = \{(s^\star, e_1), \ldots, (s^\star, e_d)\}$).

Since μ is a distribution that the oracle could have chosen and consists of just $|\mu| = d \leq \mathcal{N}(\gamma, d)$ queries, apply Theorem 3 *(Policy-Free Lower Bounds)* to deduce that even if infinite data is generated from μ, no batch algorithm can return the value of $Q^\pi(s^\star, \cdot)$ for some action with additive error < 1 and with probability $> \frac{1}{2}$; likewise it cannot return a policy with suboptimality error < 1 from $s^\star$ with probability $> \frac{1}{2}$. □

In summary, there exist problems where the batch algorithm cannot return a reasonable answer even with infinite data, the best conditioned covariance matrix and a strong representation for the action value function. We highlight that the exponential lower bound in (Wang et al., 2020a)'s construction can potentially be avoided by choosing a better batch distribution since their MDPs are tabular with small state-action spaces; our MDPs are instead much harder and cannot be solved even if one chooses — through an oracle — the best distribution for such problem class. Similar results can be derived for any assumption on the covariance matrix.

Finally, we highlight that all theorems that we present in this work are expressed as a function of the number of state-action queries, i.e., the size of the support of μ, instead of the number of sampled rewards and transitions. This way, if the support of the batch distribution is small $|\mu| \leq 2^{-d}\mathcal{N}(\gamma, d)$ then our theorems ensure that even infinite samples from the state-actions where μ is supported do not suffice to make accurate predictions, because the batch algorithm already receives the full reward and transition functions in our interaction protocol. As a clarifying example, consider choosing a sampling distribution μ according to globally-optimal design of experiments (Pukelsheim, 2006), which is an optimal procedure in linear bandits to recover the unknown parameter (Lattimore & Szepesvári, 2020). Design of experiments fails to be effective in our hard instances because it prescribes only $O(d^2)$ distinct state-actions, i.e., queries, where to (repeatedly) acquire samples, but to escape the lower bound one does not need more samples at the same state-actions; instead *the support of the sampling distribution needs to grow*.

B. Related Literature

For tabular MDPs one query in each state and action pair with exact feedback is sufficient to identify the MDP. The available results for learning in tabular MDPs consider a similar setting, where only noisy observations are available (Azar et al., 2012; Li et al., 2020; Agarwal et al., 2020b). The algorithms examined by these authors are batch algorithms which specify a uniform distribution over all the state-actions and are minimax optimal. Adaptive algorithms with a generative model (Zanette et al., 2019a; Marjani & Proutiere, 2020) exist but offer little minimax advantage.[4] In the following discussion we primarily consider the function approximation setting with linear value functions.

---

[4]They can however remove the explicit dependence on the action space from the main rate even in minimax problems, see (Zanette et al., 2019a), section 6.

Negative results: lower bounds Polynomial lower bounds are routinely derived to certify that a certain algorithm is sample efficient. For example, (Jiang & Li, 2016) present lower bounds for tabular RL and (Duan & Wang, 2020; Hao et al., 2020) for the linear setting under additional closure assumptions on the Bellman operator. Instead, in this work we are interested in exponential lower bounds.

Dynamic programming algorithms like least-square value and policy iteration (Bradtke & Barto, 1996; Lagoudakis & Parr, 2003) are widely adopted but they are prone to divergence (Baird, 1995; Tsitsiklis & Van Roy, 1996). The hardness of obtaining statistically efficient algorithms has motivated researchers to look for information-theoretic lower bounds. In particular, (Chen & Jiang, 2019) provide a generic lower bound in absence of concentrability; such lower bound does not apply to our setting as we further assume a linear structure. Du et al. (2019) show that if a linear predictor is highly misspecified then an exponential number of queries is needed, and Lattimore et al. (2020) generalize this result through a corollary to show that using ϵ-accurate linear predictors in dimension d can only give $\sqrt{d}\epsilon$-optimal policies for linear bandits and RL. While these results are relevant to our setting, they becomes vacuous in absence of misspecification, i.e., when realizability holds. We examine the relation with concurrent work (Weisz et al., 2020; Wang et al., 2020a; Amortila et al., 2020) in Section 5; however, here we mention that their constructions are easily solvable under our assumptions.

Positive results: upper bounds When the action value function has a linear parameterization and the transitions are low rank, Yang & Wang (2020); Jin et al. (2020); Zanette et al. (2020a) propose regret-minimizing online algorithms; these works implicitly imply that both the BPI and OPE batch problems are easily solvable under low-rank dynamics. For the more general setting that the value function is closed under the Bellman operator, batch upper bounds for the BPI problem exist (Munos, 2005; Munos & Szepesvári, 2008) and these have been recently generalized to the online setting (Zanette et al., 2020b) with a computationally tractable algorithm (Zanette et al., 2020c); collectively, these works show that online RL can operate under batch assumptions, and our work shows that online RL is in fact exponentially easier than batch RL on certain problems. Minimax results for batch OPE problems are also available (Duan & Wang, 2020) for a setting essentially equivalent to low inherent Bellman error; for the BPI problem see instead (Xie & Jiang, 2020a). Batch methods based on minimizing the Bellman residual also exist (Antos et al., 2008). When the action value function of all policies can be linearly represented Munos (2003); Lazaric et al. (2012); Lattimore et al. (2020); Agarwal et al. (2020c) provide algorithms to learn a good policy using a simulator, and online using stronger conditions (Agarwal et al., 2020a); all these methods are non-batch, and our Theorem 3 rules out the existence of batch algorithms operating under the same assumptions. Other linear models recently considered include (Ayoub et al., 2020; Zhou et al., 2020) and when the Bellman equations are linearly representable up to a low-rank error (Jiang et al., 2017); although the focus of these works is online exploration, learning should be possible in the batch setting as well.

The optimal policy (as opposed to a near optimal one) can be identified if there exists some separation between the best and the second best policy (Du et al., 2020), although a sample complexity proportional to the inverse gap (which can be exponentially small) must be suffered; learning in such setting is also possible with an entirely batch algorithm. Near-convexity also ensures that both the BPI and OPE learning problems are solvable in the batch setting (Zanette et al., 2019b); under exact convexity (Cui & Yang, 2020) give an optimal sample complexity for the BPI problem. Deterministic systems with linear value functions are also learnable (Wen & Van Roy, 2013) in finite horizon, and it is easy to derive a batch algorithm for both the BPI and OPE problem under this assumption. Xie & Jiang (2020b) show that under strong concentrability assumptions the BPI problem is solvable using only realizability and general (non-linear) function approximators and Liu et al. (2020) show how to find the best-in-class solution to the BPI problem while avoiding concetrability for general approximators. Finally, importance sampling estimators (Precup, 2000; Li et al., 2015) make very little assumptions but as such they do not leverage the linear structure of the problem and can exhibit unbounded variance. However, methods to reduce the variance exist (Jiang & Li, 2016; Thomas & Brunskill, 2016; Liu et al., 2018; Xie et al., 2019).

C. Globally Optimal Experimental Design is Not a Good Sampling Strategy in RL

We wonder if the classical globally optimal design of experiment (DoE) (Pukelsheim, 2006) from statistics produces acceptable results in light of our lower bounds, for example, whether it yields a matching exponential upper bound. DoE prescribes a set of states and actions (the design) where to acquire several samples; multiple samples are acquired at the selected state-actions, according to the design. The procedure thus prescribes policy-free queries. In finite horizon DoE is successful: Weisz et al. (2020) conclude that DoE with value iteration can identify a near optimal policy in finite horizon given exponentially many samples.

Dishearteningly, we find that in infinite horizon this is no longer true: in the limit of infinite samples over the feature vectors selected by DoE the learner receives exact feedback, but since DoE always prescribes at most $d(d+1)/2$ distinct vectors where samples are acquired (see (Lattimore & Szepesvári, 2020; Pukelsheim, 2006)), the support of the query set is always at most $d(d+1)/2$ regardless of the number of actual samples along these feature vectors. For large enough d and γ close to 1 we have $d(d+1)/2 \leq I^{-1}_{1-\gamma^2}(\frac{d-1}{2}, \frac{1}{2})$ as the left hand side is polynomial and the right hand side is exponential. In light of Theorem 3, we conclude that even if infinite data are collected over the feature vectors chosen by DoE and all policies have a linear representation, no batch algorithm can output a good policy or estimate the value of another with good enough probability. This happens because traditional DoE fails to account for the effect of bootstrapping, as we explain in Section 4. Thus, in infinite horizon the challenge is much greater than in finite horizon, because a good policy (or the value of a target policy) cannot eventually be learned by just collecting more samples, instead, *the support of the sampling distribution needs to grow*.

D. Preliminaries

D.1. Additional Notation

We indicate with $\mathcal{B}$ the Euclidean ball in dimension d, i.e., $\mathcal{B} \stackrel{def}{=} \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$. We denote with $\partial\mathcal{B} \stackrel{def}{=} \{x \in \mathbb{R}^d \mid \|x\|_2 = 1\}$ the surface of the unit ball. In addition, let $e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]$ be the i-th canonical vector in $\mathbb{R}^d$, i.e., the vector of all zeros with a 1 in the i-th position. We define the unit Euclidean ball restricted to the positive orthant: $\mathcal{B}^+ \stackrel{def}{=} \{x \in \mathcal{B} \mid x^\top e_i \geq 0, \forall i \in [d]\}$ and its outer surface $\partial\mathcal{B}^+ \stackrel{def}{=} \{x \in \partial\mathcal{B} \mid x^\top e_i \geq 0, \forall i \in [d]\}$. Sometimes we indicate with $\vec{0}$ the zero vector in $\mathbb{R}^d$ to distinguish it from the $0 \in \mathbb{R}$. We occasionally write $r_M, p_M, V_M, Q_M, \mathcal{T}_M, \pi_M$ to highlight the dependence on a certain MDP M.

In the proofs for the lower bounds we consider a class $\mathcal{M}$ of MDPs containing MDP instances M with certain properties. Every MDP in the class $\mathcal{M}$ can be identified by a vector w and a sign $+$ or $-$, and is denoted by $M_{w,+}$ or $M_{w,-}$, respectively. We overload the notation slightly and write M_w in a statement to indicate the statement holds for both $M_{w,+}$ and $M_{w,-}$. This allows us to write, for example, $\mathcal{M} = \{M_{w,+} \mid w \in \partial\mathcal{B}\} \cup \{M_{w,-} \mid w \in \partial\mathcal{B}\} = \{M_w \mid w \in \partial\mathcal{B}\}$.

The volume of a (measurable) set $\Omega \in \mathbb{R}^d$ is defined as $\mathrm{Vol}(\Omega) = \int_{x \in \Omega} dx$.

The *gamma function* for a non-negative $z \in \mathbb{R}$ is defined as

$$\Gamma(z) \stackrel{def}{=} \int_0^\infty x^{z-1} e^{-x} dx$$

and the *beta function* for non-negative $a, b \in \mathbb{R}$ is defined as

$$B(a, b) \stackrel{def}{=} \int_0^1 t^{a-1}(1-t)^{b-1} dt.$$

If $x \in [0, 1]$ we can define the *incomplete beta function*

$$B(x, a, b) = \int_0^x t^{a-1}(1-t)^{b-1} dt$$

which allows us to define the regularized incomplete beta function

$$I_x(a, b) \stackrel{def}{=} \frac{B(x, a, b)}{B(a, b)}.$$

Useful bounds are provided in Lemma 3 *(Upper Bound on I)*.

D.2. Learning Finite Horizon MDPs with Policy-Free Queries

In this section we describe the process to learn a finite horizon MDPs with a linear representation $Q_t^\pi(s, a) = \phi_t(s, a)^\top \theta_t^\pi$ for the target policy π and $Q_t^\star(s, a) = \phi_t(s, a)^\top \theta_t^\star$ for the optimal action-value function, $\forall t \in [H]$. We make the same assumptions as in the main text.

Consider the following learner that submits policy-free queries. First, in every timestep $t \in [H]$ the oracle selects appropriate state-actions $(s_1, a_1), \ldots, (s_d, a_d)$ such that the corresponding feature vectors $\phi_t(s_1, a_1), \ldots, \phi_t(s_d, a_d)$ form a basis

for $\mathbb{R}^d$ (if a basis does not exist it is possible to reduce the dimensionality of the problem). Then for the BPI problem, the batch algorithm runs least-square value iteration backward. By induction, the algorithm identifies $\theta_t^\star$ for all t. In particular, assume $\theta_{t+1}^\star$ is known, which is certainly true for $t + 1 = H + 1$ (the base case). Then the batch algorithm solves in every timestep t the following linear system of equations ($\forall i \in [d]$):

$$\phi_t(s_i, a_i)\theta_t = r(s_i, a_i) + \mathbb{E}_{s' \sim p(s_i, a_i)} \max_{a' \in \mathcal{A}_{s'}} \phi(s', a')^\top \theta_{t+1}^\star \tag{4}$$

$$= r(s_i, a_i) + \mathbb{E}_{s' \sim p(s_i, a_i)} V_{t+1}^\star(s') \tag{5}$$

$$= Q_t^\star(s_i, a_i). \tag{6}$$

Since $\phi_t(s_i, a_i), i = 1, \ldots, d$ form a basis for $\mathbb{R}^d$, $\theta_t = \theta_t^\star$ is the unique solution. By induction, we conclude that the BPI problem can be solved in finite horizon MDPs under our assumptions.

The OPE problem is analogous. The batch algorithm solves the following linear system of equations for $i \in [d]$ at timestep t.

$$\phi_t(s_i, a_i)\theta_t = r(s_i, a_i) + \mathbb{E}_{s' \sim p(s_i, a_i)} \phi(s', \pi(s'))^\top \theta_{t+1}^\pi \tag{7}$$

$$= r(s_i, a_i) + \mathbb{E}_{s' \sim p(s_i, a_i)} V_{t+1}^\pi(s') \tag{8}$$

$$= Q_t^\pi(s_i, a_i). \tag{9}$$

Since $\phi_t(s_i, a_i), i = 1, \ldots, d$ form a basis for $\mathbb{R}^d$, $\theta_t = \theta_t^\pi$ is the unique solution. By induction, we conclude that the OPE problem can be solved in finite horizon MDPs under our assumptions.

D.3. Hypersphere and Hyperspherical Sectors

Definition 3 (Hypersphere, hyperspherical cap and hyperspherical sector). *Fix $b \geq 0$ and define the b-hyperspherical cap in direction $w \in \mathcal{B}$ as*

$$\mathcal{C}_b(w) \overset{def}{=} \{x \in \mathcal{B} \mid \frac{x^\top w}{\|w\|_2} \geq b\} \tag{10}$$

and the b-hyperspherical sector in direction w as

$$\overset{\triangle}{\mathcal{C}}_b(w) \overset{def}{=} \left\{x \in \mathcal{B} \mid \frac{x^\top w}{\|x\|_2 \|w\|_2} \geq b\right\}. \tag{11}$$

The following formulas are useful to compute the volume of the hypersphere $\mathcal{B}$ and of a spherical sector $\overset{\triangle}{\mathcal{C}}_b(w)$.

Fact 1 (Volume of a Hyperspherical Sector). *The volume of an spherical sector is given by the formula*

$$\mathrm{Vol}(\overset{\triangle}{\mathcal{C}}_b(w)) = \frac{\mathrm{Vol}(\mathcal{B})}{2} I_{1-b^2}\left(\frac{d-1}{2}, \frac{1}{2}\right). \tag{12}$$

where the volume of the ball $\mathcal{B}$ is

$$\mathrm{Vol}(\mathcal{B}) = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2} + 1)}. \tag{13}$$

Proof. For the proof, see (Li, 2011) where ϕ in their notation is the half-angle of the hypersector and $\sin^2 \phi = 1 - \cos^2 \phi = 1 - b^2$. □

D.4. Bounds on the Regularized Incomplete Beta Function

Lemma 3 (Upper Bound on I). *The following upper bound holds true for $\gamma \in (0, 1)$ and $d \geq 3$:*

$$I_{1-\gamma^2}(\frac{d-1}{2}, \frac{1}{2}) \leq \frac{1}{\gamma} \times \frac{2}{d-1} \times (1 - \gamma^2)^{\frac{d-1}{2}} \times \sqrt{\frac{d}{2\pi}}. \tag{14}$$

Proof. We need to compute an upper bound on

$$I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right) \stackrel{def}{=} \frac{B(1-\gamma^2, \frac{d-1}{2}, \frac{1}{2})}{B(\frac{d-1}{2}, \frac{1}{2})}. \tag{15}$$

We first compute an upper bound on the incomplete Beta function

$$B(1-\gamma^2, \frac{d-1}{2}, \frac{1}{2}) \stackrel{def}{=} \int_0^{1-\gamma^2} t^{\frac{d-1}{2}-1}(1-t)^{\frac{1}{2}-1}dt \tag{16}$$

$$= \int_0^{1-\gamma^2} \sqrt{\frac{t^{d-3}}{(1-t)}}dt. \tag{17}$$

Notice that $t \leq 1-\gamma^2$ and so $1-t \geq \gamma^2$ and finally $\frac{1}{1-t} \leq \frac{1}{\gamma^2}$. Therefore the following upper bound follows:

$$B(1-\gamma^2, \frac{d-1}{2}, \frac{1}{2}) \leq \frac{1}{\gamma}\int_0^{1-\gamma^2} t^{\frac{d-1}{2}-1}dt \tag{18}$$

$$= \frac{1}{\gamma} \times \frac{2}{d-1} \times t^{\frac{d-1}{2}}\big|_{t=1-\gamma^2} \tag{19}$$

$$= \frac{1}{\gamma} \times \frac{2}{d-1} \times (1-\gamma^2)^{\frac{d-1}{2}}. \tag{20}$$

It remains to compute a lower bound on the Beta function:

$$B(\frac{d-1}{2}, \frac{1}{2}) = \frac{\Gamma(\frac{d-1}{2})\Gamma(\frac{1}{2})}{\Gamma(\frac{d-1}{2}+\frac{1}{2})}. \tag{21}$$

We have $\Gamma(\frac{1}{2}) = \sqrt{\pi}$; we thus focus on the ratio

$$\left[\frac{\Gamma(\frac{d-1}{2})}{\Gamma(\frac{d-1}{2}+\frac{1}{2})}\right]^{-1} = \frac{\Gamma(\frac{d-1}{2}+\frac{1}{2})}{\Gamma(\frac{d-1}{2})} \tag{22}$$

$$= \frac{\Gamma(\frac{d-2}{2}+1)}{\Gamma(\frac{d-2}{2}+\frac{1}{2})} \tag{23}$$

$$\leq \left(\frac{d-2}{2}+1\right)^{1-\frac{1}{2}} \tag{24}$$

$$= \sqrt{\frac{d}{2}}. \tag{25}$$

The inequality is Gautschi's inequality for the gamma function. Therefore, we conclude

$$B(\frac{d-1}{2}, \frac{1}{2}) \geq \sqrt{\frac{2\pi}{d}}. \tag{26}$$

The bounds just derived for the beta functions together yield an upper bound on the regularized incomplete beta function:

$$I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right) \leq \frac{1}{\gamma} \times \frac{2}{d-1} \times (1-\gamma^2)^{\frac{d-1}{2}} \times \sqrt{\frac{d}{2\pi}}. \tag{27}$$

$$\square$$

Using the above result we can derive the following easily interpretable lower bounds.

Corollary 2 (Lower Bounds on I^{-1}). *If $d \geq 5$ then the following lower bounds holds true for $\gamma \in (0, 1)$:*

$$I_{1-\gamma^2}^{-1}\left(\frac{d-1}{2}, \frac{1}{2}\right) \geq \gamma\sqrt{d}\left(\left(\frac{1}{2}\right) \cdot \frac{1}{(1-\gamma)}\right)^{\frac{d-1}{2}}, \tag{28}$$

$$2^{-d}I_{1-\gamma^2}^{-1}\left(\frac{d-1}{2}, \frac{1}{2}\right) \geq \gamma\sqrt{d}\left(\left(\frac{1}{2}\right)^{2.5} \cdot \frac{1}{(1-\gamma)}\right)^{\frac{d-1}{2}}. \tag{29}$$

Proof. Using Lemma 3 we can derive the following crude lower bound for $d \geq 5$

$$I_{1-\gamma^2}^{-1}\left(\frac{d-1}{2}, \frac{1}{2}\right) \geq \gamma\sqrt{\frac{\pi}{2}} \times \frac{d-1}{\sqrt{d}} \times \left(\frac{1}{(1+\gamma)(1-\gamma)}\right)^{\frac{d-1}{2}} \tag{30}$$

$$\geq \gamma\sqrt{d}\left(\left(\frac{1}{2}\right) \cdot \frac{1}{(1-\gamma)}\right)^{\frac{d-1}{2}}. \tag{31}$$

Under the same conditions we have the following lower bound

$$2^{-d}I_{1-\gamma^2}^{-1}\left(\frac{d-1}{2}, \frac{1}{2}\right) \geq \gamma\sqrt{d}\left(\left(\frac{1}{2}\right)^{\frac{2d}{d-1}} \cdot \left(\frac{1}{2}\right)\frac{1}{(1-\gamma)}\right)^{\frac{d-1}{2}} \tag{32}$$

$$\geq \gamma\sqrt{d}\left(\left(\frac{1}{2}\right)^{3.5} \cdot \frac{1}{(1-\gamma)}\right)^{\frac{d-1}{2}}. \tag{33}$$

$\square$

E. Existence of a Lonely Hyperpherical Cap

Consider a set of n points in the unit ball $\{y_1, \ldots, y_n\} \subseteq \mathcal{B}$. In this section we show that if n is less than $N_\gamma \approx \left(\frac{1}{1-\gamma}\right)^d$ then there exists an hyperspherical cap $\mathcal{C}_\gamma(w)$ (identified by its direction w) such that none of the y_i's is in $\mathcal{C}_\gamma(w)$ or its symmetric counterpart $\mathcal{C}_\gamma(-w)$. For short, define

$$N_\gamma \stackrel{def}{=} \frac{2}{I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right)} = 2\mathcal{N}(\gamma, d). \tag{34}$$

Lemma 4 (Existence of a Lonely Hyperspherical Cone). *Let $\mu = \{y_1, \ldots, y_n\} \subseteq \mathcal{B}$ be a collection of n points. If $n < N_\gamma$ then there exists a point $\widetilde{w} \in \partial\mathcal{B}$ such that its γ-spherical cone does not contain any of the y_i's, i.e.,*

$$\forall \mu = \{y_1, \ldots, y_n\} \subseteq \mathcal{B} \quad s.t. \quad n < N_\gamma, \ \exists \widetilde{w} \in \partial\mathcal{B} \quad such\ that \quad \forall y \in \{y_1, \ldots, y_n\}, \ y \notin \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w}). \tag{35}$$

Proof. This follows from a geometrical argument; the idea is that the hyperspherical cones around $y_1, \ldots, y_n$ with parameter γ are not sufficient to cover the whole hypersphere, leaving a "gap". A point in that gap, which we denote[5] with $w \neq 0$ is not covered by any of the hyperspherical cones $\{\overset{\triangle}{\mathcal{C}}_\gamma(y_1), \ldots, \overset{\triangle}{\mathcal{C}}_\gamma(y_n)\} \subseteq \mathcal{B}$, which in turn means that the hyperspherical cone $\overset{\triangle}{\mathcal{C}}_\gamma(w)$ around such point w cannot cover any of the y's.

More formally, consider the hyperspherical cones $\overset{\triangle}{\mathcal{C}}_\gamma(y_1), \ldots, \overset{\triangle}{\mathcal{C}}_\gamma(y_n)$. The volume of their union is at most

$$\mathrm{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \cup \cdots \cup \overset{\triangle}{\mathcal{C}}_\gamma(y_n)\right) \leq \sum_{i=1}^{n} \mathrm{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_i)\right) = n\mathrm{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1)\right). \tag{36}$$

We have

$$\frac{\mathrm{Vol}(\mathcal{B})}{\mathrm{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1)\right)} = \frac{2}{I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right)} = N_\gamma \tag{37}$$

where the first equality follows from Fact 1 *(Volume of a Hyperspherical Sector)*. It is easily seen that if $n < N_\gamma$ then there must exist a point $w \neq 0, w \in \mathcal{B}$ (in fact, a whole subset of $\mathcal{B}$ of non-zero volume) not covered by any spherical sector around the y's, i.e.,

$$\exists w \in \mathcal{B} \quad such\ that \quad w \notin \bigcup_{i=1}^{n} \overset{\triangle}{\mathcal{C}}_\gamma(y_i). \tag{38}$$

Now denote with $\widetilde{w} \stackrel{def}{=} \frac{w}{\|w\|_2}$ its normalization; it follows by definition that $\widetilde{w} \notin \bigcup_{i=1}^{n} \overset{\triangle}{\mathcal{C}}_\gamma(y_i)$ and $\widetilde{w} \in \partial B$. Consider the spherical cone around $\widetilde{w}$, i.e., consider $\overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$. By symmetry we have that none of the y_i's can be in $\overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$. This is because $\widetilde{w} \notin \overset{\triangle}{\mathcal{C}}_\gamma(y_i)$ means

$$\frac{y_i^\top \widetilde{w}}{\|y_i\|_2 \|\widetilde{w}\|_2} < \gamma \tag{39}$$

which is equivalent to saying $y_i \notin \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$, and this can be repeated for every $i \in [n]$. $\qquad\square$

We also need the following closely related result in the positive orthant.

---

[5]We save the notation $\widetilde{w}$ for its normalization, i.e., $\widetilde{w} = \frac{w}{\|w\|_2}$.

Lemma 5 (Existence of a Lonely Spherical Cone in the Positive Orthant). *Let $\mu = \{y_1, \ldots, y_n\} \subseteq \mathcal{B}^+$ be a collection of n points. If $n < 2^{-d} N_\gamma$ then there exists a point $\widetilde{w} \in \partial \mathcal{B}^+$ that also satisfies $e_i^\top \widetilde{w} > 0$, $\forall i \in [d]$ such that its γ-spherical cone does not contain any of the y_i's, i.e.,*

$$\forall \mu = \{y_1, \ldots, y_n\} \subseteq \mathcal{B}^+, \ n < 2^{-d} N_\gamma, \ \exists \widetilde{w} \in \partial \mathcal{B}^+, \ e_i^\top \widetilde{w} > 0 \ \forall i \in [d], \tag{40}$$

$$\text{such that} \quad \forall y \in \{y_1, \ldots, y_n\}, \ y \notin \mathcal{C}_\gamma(\widetilde{w}). \tag{41}$$

Proof. The proof for the positive orthant (i.e., restricted to $\mathcal{B}^+$) is nearly identical to Lemma 4 *(Existence of a Lonely Hyperspherical Cone)*. Consider the hyperspherical sectors intersected with $\mathcal{B}^+$, i.e., $\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \cap \mathcal{B}^+, \ldots, \overset{\triangle}{\mathcal{C}}_\gamma(y_n) \cap \mathcal{B}^+$. The volume of their union is at most

$$\text{Vol}\left((\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \cap \mathcal{B}^+) \cup \cdots \cup (\overset{\triangle}{\mathcal{C}}_\gamma(y_n) \cap \mathcal{B}^+) \right) \leq \sum_{i=1}^n \text{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_i) \cap \mathcal{B}^+ \right) \tag{42}$$

$$\leq \sum_{i=1}^n \text{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_i) \right) \tag{43}$$

$$= n \text{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \right). \tag{44}$$

From geometry we know that the volume of the hypersphere in its positive orthant is 2^{-d} times the volume of the hypersphere:

$$\frac{\text{Vol}(\mathcal{B}^+)}{\text{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \right)} = \frac{2^{-d} \text{Vol}(\mathcal{B})}{\text{Vol}\left(\overset{\triangle}{\mathcal{C}}_\gamma(y_1) \right)} = 2^{-d} \frac{2}{I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2} \right)} = 2^{-d} \cdot N_\gamma. \tag{45}$$

where the second equality follows from Fact 1 *(Volume of a Hyperspherical Sector)*. It is easily seen that if $n < 2^{-d} N_\gamma$ then there must exist a whole subset of $\mathcal{B}^+$ of non-zero volume not covered by any hyperspherical cone around the y's, which allows us to claim

$$\exists w \in \mathcal{B}^+, \ w^\top e_i > 0, \forall i \in [d], \quad \text{such that} \quad w \notin \bigcup_{i=1}^n \overset{\triangle}{\mathcal{C}}_\gamma(y_i). \tag{46}$$

Now denote with $\widetilde{w} \overset{def}{=} \frac{w}{\|w\|_2}$ its normalization; it follows by definition that $\widetilde{w} \notin \bigcup_{i=1}^n \overset{\triangle}{\mathcal{C}}_\gamma(y_i)$ and $\widetilde{w} \in \partial \mathcal{B}^+$, $\widetilde{w}^\top e_i > 0, \forall i \in [d]$. Consider the hyperspherical cone around $\widetilde{w}$, i.e., consider $\overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$. By symmetry we have that none of the y_i's can be in $\overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$. This is because $\widetilde{w} \notin \overset{\triangle}{\mathcal{C}}_\gamma(y_i)$ means

$$\frac{y_i}{\|y_i\|_2}^\top \frac{\widetilde{w}}{\|\widetilde{w}\|_2} < \gamma \tag{47}$$

which is equivalent to saying $y_i \notin \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$, and this can be repeated for every $i \in [n]$. Since $\mathcal{C}_\gamma(\widetilde{w}) \subset \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w})$, the thesis follows. $\qquad\square$

Building on Lemma 4 *(Existence of a Lonely Hyperspherical Cone)*, we can actually show there exists *two symmetric* hyperspherical sectors that do not contain any of y's.

Lemma 6 (Existence of two Symmetric Hyperspherical Cones). *Let $\{y_1, \ldots, y_n\} \subseteq \mathcal{B}$ be a collection of n points. If $n < N_\gamma/2$ then there exists a point $\widetilde{w} \in \partial \mathcal{B}$ such that the γ-hyperspherical cones around $\widetilde{w}$ and $-\widetilde{w}$ do not contain any of the y's, i.e.,*

$$\exists \widetilde{w} \in \partial \mathcal{B} \quad \text{such that} \quad y_i \notin \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w}) \cup \overset{\triangle}{\mathcal{C}}_\gamma(-\widetilde{w}), \qquad \forall i \in [n], \tag{48}$$

so in particular, $y_i \notin \mathcal{C}_\gamma(\widetilde{w}) \cup \mathcal{C}_\gamma(-\widetilde{w}), \forall i \in [n]$.

Proof. Consider the augmented set $\{y_1, \ldots, y_n, -y_1, \ldots, -y_n\}$. Then Lemma 4 *(Existence of a Lonely Hyperspherical Cone)* applied to this set ensures that if $2n < N_\gamma$ then there exists a point $\widetilde{w} \in \mathcal{B}$ with unit norm $\|\widetilde{w}\|_2 = 1$ such that

$$y \notin \overset{\triangle}{\mathcal{C}}_\gamma(\widetilde{w}), \qquad \forall y \in \{y_1, \ldots, y_n, -y_1, \ldots, -y_n\}. \tag{49}$$

This means

$$\frac{y_i^\top \widetilde{w}}{\|y_i\|_2} < \gamma \rightarrow y_i^\top \widetilde{w} < \|y_i\|_2 \gamma < \gamma \tag{50}$$

$$\frac{-y_i^\top \widetilde{w}}{\|y_i\|_2} < \gamma \rightarrow -y_i^\top \widetilde{w} < \|y_i\|_2 \gamma < \gamma. \tag{51}$$

and so in particular, $y_i \notin \mathcal{C}_\gamma(\widetilde{w}) \cup \mathcal{C}_\gamma(-\widetilde{w}), \forall i \in [n]$. $\qquad \square$

F. Proof of Theorem 1

F.1. MDP Class

We consider a class $\mathcal{M}$ of MDPs sharing the same state-space $\mathcal{S}$, action space $\mathcal{A}$, discount factor γ, and transition function p. The MDPs differ only in the reward function r and the prescribed target policy. We first describe the state and action space, the discount factor and the transition probabilities, which are fixed across all MDPs in the class $\mathcal{M}$.

State space Each state s in the state space $\mathcal{S}$ can be identified by a point in the Euclidean ball, i.e., we write $\mathcal{S} = \mathcal{B}$. The starting state is the origin $s^\star = \vec{0} \in \mathcal{B}$.

Action space In each state $s \in \mathcal{S}$ the action set coincides with the unit ball, i.e., $\forall s \in \mathcal{S}, \mathcal{A}_s = \mathcal{B}$.

Discount factor The discount factor γ is in the interval $(0, 1)$.

Feature map The feature extractor returns the point in the Euclidean ball corresponding to the action chosen in the selected state. Mathematically

$$\forall (s, a), \ \phi(s, a) = a.$$

Since the available actions are always a subset of $\mathcal{B}$, it is easy to see that the image of the feature map is the set $\mathcal{B}$ and $\|\phi(\cdot, \cdot)\|_2 \le 1$ holds.

Transition Function The transition function is deterministic and is only a function of the action chosen in the current state; it is thus convenient to denote with $s^+(a)$ the unique successor state reached upon taking action a (in any state s). The successor state is equivalent to the action taken, i.e.,

$$s^+(a) = a. \tag{52}$$

Since $a \in \mathcal{B}$, we have that $s^+(a) \in \mathcal{B}$, which is a valid state, and so the above display identifies a valid transition function.

F.2. Instance of the Class

Every MDP in the class $\mathcal{M}$ can be identified by a vector $w \in \partial \mathcal{B}$ and a sign $+$ or $-$, and is denoted by $M_{w,+}$ or $M_{w,-}$, respectively. We now describe the MDP-specific reward function and target policy.

Target policy The target policy depends on the vector $w \in \partial \mathcal{B}$ that identifies the MDP M_w, but not on the sign $+$ or $-$ that distinguishes $M_{w,+}$ from $M_{w,-}$ (therefore, knowledge of the target policy does not reveal the exact MDP in the class). In addition, the target policy is deterministic. Let $\mathcal{C}_\gamma(w)$ be a γ-hyperspherical cap in direction w (see Definition 3 *(Hypersphere, hyperspherical cap and hyperspherical sector)*). Then the target policy is defined as

$$\pi_w(s) = \begin{cases} \frac{1}{\gamma}(s^\top w)w, & \text{if } s \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w), \\ s & \text{otherwise.} \end{cases} \tag{53}$$

Since $s \in \mathcal{B}$ and $w \in \partial \mathcal{B}$, the linear algebra operations in the above definition are all well defined. In addition, if $s \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ we have $\|\frac{1}{\gamma}(s^\top w)w\|_2 = \frac{|s^\top w|}{\gamma}\|w\|_2 \le \frac{\gamma}{\gamma}\|w\|_2 = 1$ using the definition of hyperspherical cap in Definition 3 *(Hypersphere, hyperspherical cap and hyperspherical sector)*, so $\frac{1}{\gamma}(s^\top w)w \in \mathcal{B}$ as well. This means the action chosen by the target policy $\pi_w(s)$ lives in $\mathcal{B}$ and it is thus a valid action, so the definition of the target policy in the above display is well posed.

Reward function The reward function depends on both the vector $w \in \partial \mathcal{B}$ and on the sign $+$ or $-$ identifying the MDP and it is defined as follows:

$$\text{on } M_{w,+}: \quad r_{w,+}(s,a) \stackrel{def}{=} \begin{cases} 0 & \text{if } a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w) \\ (1-\gamma)a^\top(+w) & \text{otherwise} \end{cases} \tag{54}$$

$$\text{on } M_{w,-}: \quad r_{w,-}(s,a) \stackrel{def}{=} \begin{cases} 0 & \text{if } a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w) \\ (1-\gamma)a^\top(-w) & \text{otherwise} \end{cases} \tag{55}$$

$$\tag{56}$$

Notice that this is a valid definition as the linear algebra operations are well defined.

Remark 3 (Opposite MDPs). *The prescribed target policy is the same on $M_{w,+}$ and $M_{w,-}$. In particular, the two MDPs have identical transition functions but opposite reward functions. In addition, notice that $M_{w,+} = M_{-w,-}$ and $M_{w,-} = M_{-w,+}$.*

Remark 4 (Planning to Explore). *Since the dynamics are the same for all MDPs $\in \mathcal{M}$, and the oracle has access to each one of them, it must know the dynamics, i.e., the oracle knows how to navigate the environment.*

F.3. Realizability

We compute the action value function for each MDP (and its associated policy) in the class, showing realizability.

Lemma 7 (Q^π Realizability). *For any $w \in \partial \mathcal{B}$ let $Q_{w,+}$ and $Q_{w,-}$ be the action-value functions of π_w (the target policy) on $M_{w,+}$ and $M_{w,-}$, respectively. Then it holds that*

$$\forall(s,a), \quad \begin{cases} Q_{w,+}(s,a) = \phi(s,a)^\top(+w) & \text{on } M_{w,+}, \\ Q_{w,-}(s,a) = \phi(s,a)^\top(-w) & \text{on } M_{w,-}. \end{cases} \tag{57}$$

Proof. Consider $M_{w,+}$. At all (s,a), $Q_{w,+}$ must satisfy the Bellman evaluation equations for π_w. In particular, consider applying $\mathcal{T}_{w,+}^{\pi_w}$ to $Q_{w,+}$

$$(\mathcal{T}_{w,+}^{\pi_w} Q_{w,+})(s,a) = r_{w,+}(s,a) + \gamma Q_{w,+}(s_w^+(a), \pi_w(s_w^+(a)))$$
$$= r_{w,+}(s,a) + \gamma Q_{w,+}(a, \pi_w(a)) \tag{58}$$

since $s_w^+(a) = a$ is the only possible successor state. We now evaluate the RHS. Two cases are possible: if $a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then the reward function is zero and the rhs of the above equation reads

$$\gamma Q_{w,+}(a, \pi_w(a)) = \gamma \phi(a, \pi_w(a))^\top(+w) \tag{59}$$
$$= \gamma \left(\frac{1}{\gamma}(a^\top w)w \right)^\top (+w) \tag{60}$$
$$= a^\top w \tag{61}$$
$$= \phi(s,a)^\top w \tag{62}$$
$$= Q_{w,+}(s,a). \tag{63}$$

If conversely $a \in \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then the rhs reads

$$(1-\gamma)a^\top w + \gamma Q_{w,+}(a, \pi_w(a)) = (1-\gamma)a^\top w + \gamma \phi(a, \pi_w(a))^\top(+w) \tag{64}$$
$$= (1-\gamma)a^\top w + \gamma a^\top w \tag{65}$$
$$= a^\top w \tag{66}$$
$$= \phi(s,a)^\top w \tag{67}$$
$$= Q_{w,+}(s,a). \tag{68}$$

Thus $Q_{w,+}$ is the action value function of the target policy on $M_{w,+}$.

Similarly we verify that $Q_{w,-}$ solves the Bellman evaluation equation on $M_{w,-}$.

$$(\mathcal{T}_{w,-}^{\pi_w} Q_{w,-})(s,a) = r_{w,-}(s,a) + \gamma Q_{w,-}(s_w^+(a), \pi_w(s_w^+(a))) \tag{69}$$

$$= r_{w,-}(s,a) + \gamma Q_{w,-}(a, \pi_w(a)). \tag{70}$$

We now evaluate the RHS. Two cases are possible: if $a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then the reward function is zero and the rhs of the above equation reads

$$\gamma Q_{w,-}(a, \pi_w(a)) = \gamma \phi(a, \pi_w(a))^\top(-w) \tag{71}$$

$$= \gamma \left(\frac{1}{\gamma}(a^\top w)w\right)^\top (-w) \tag{72}$$

$$= -a^\top w \tag{73}$$

$$= \phi(s,a)^\top(-w) \tag{74}$$

$$= Q_{w,-}(s,a). \tag{75}$$

If conversely $a \in \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then the rhs reads

$$(1-\gamma)a^\top(-w) + \gamma Q_{w,-}(a, \pi_w(a)) = (1-\gamma)a^\top(-w) + \gamma \phi(a, \pi_w(a))^\top(-w) \tag{76}$$

$$= (1-\gamma)a^\top(-w) + \gamma a^\top(-w) \tag{77}$$

$$= -a^\top w \tag{78}$$

$$= \phi(s,a)^\top(-w) \tag{79}$$

$$= Q_{w,-}(s,a). \tag{80}$$

$\square$

F.4. Proof of Theorem 1

Proof. Consider the MDP class described in Appendix F.1 and Appendix F.2. First, from Lemma 7 *(Q^π Realizability)* we know that every instance of the OPE problem satisfies Assumption 1 *(Q^π is Realizable)* with a given feature map $\phi(\cdot, \cdot)$.

Next, we know that the set of query points is induced by one or more policies. However notice that any fixed policy visits the same states regardless of the MDP because the transition function does not depend on the particular MDP at hand. Therefore, the policy-induced query set is also policy-free. Thus, it suffices to do the proof for a policy free query set μ. In addition, the size of the policy-free query set μ is exactly n since the MDP is deterministic.

From Lemma 6 *(Existence of two Symmetric Hyperspherical Cones)* we know that if $|\mu| < I_{1-\gamma^2}^{-1}(\frac{d-1}{2}, \frac{1}{2})$ then $\exists \widetilde{w} \in \partial \mathcal{B}$ such that $\forall (s,a) \in \mu, a \notin \mathcal{C}_\gamma(\widetilde{w}) \cup \mathcal{C}_\gamma(-\widetilde{w})$. Consider the two associated MDPs $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$. Notice that the transition function is by construction identical on $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$ (and so is the target policy) while their reward functions are zero at any $(s,a) \in \mu$:

$$\begin{cases} r_{\widetilde{w},+}(s,a) &= r_{\widetilde{w},-}(s,a) \\ p_{\widetilde{w},+}(s,a) &= p_{\widetilde{w},-}(s,a). \end{cases} \tag{81}$$

This implies that the reward and transition functions in the dataset could have originated from either $M_{\widetilde{w},+}$ or $M_{\widetilde{w},-}$, and likewise the target policy does not indicate whether the prediction concerns $M_{\widetilde{w},+}$ or $M_{\widetilde{w},-}$.

In $s^\star = \vec{0}$ the value of $Q^{\pi_{\widetilde{w}}}(s^\star, \widetilde{w})$ is $+1$ on $M_{\widetilde{w},+}$ and -1 on $M_{\widetilde{w},-}$. The agent has thus two choices: either predict a positive value or a negative one. At best, it can randomize between the two, showing the result.

Learning with an online algorithm with knowledge of the target policy Now consider the following online algorithm that submits policy-induced queries. The algorithm first asks the target policy to output an action on the states $\frac{\gamma}{2}e_1, \ldots, \frac{\gamma}{2}e_d$, which is a basis for $\mathbb{R}^d$. This reveals the vector w of the MDP and the position of each hyperspherical cap. Then the agent probes either hyperspherical cap to gain knowledge about the reward function, which identifies the sign of the MDP.

By construction, notice that none of the state vectors $\frac{\gamma}{2}e_1, \ldots, \frac{\gamma}{2}e_d$ is located in an hyperspherical cap of height γ since $|(\frac{\gamma}{2}e_i)^\top w| \le \frac{\gamma}{2}\|e_i\|_2\|w\|_2 < \gamma$. Since $e_1, \ldots, e_d$ is a basis for $\mathbb{R}^d$, we have that $w \ne 0$ cannot be orthogonal to all of them. Therefore, there must exist an index $j \in [d]$ such that $e_j^\top w \ne 0$. The action that the target policy outputs $\pi_w(\frac{\gamma}{2}e_j)$ for any such vector that satisfies $\frac{\gamma}{2}e_j^\top w \ne 0$ and is not in $\mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ must be non-zero and must be aligned with w. Thus, the agent knows the vector w of the MDP (up to a sign). Next, it can play $(s^\star, w)$ (a behavioral policy of length 1 from the initial state $s^\star$) to probe one of the two hyperspherical caps and to observe the reward $\pm(1 - \gamma)$ which identifies the sign of the MDP. Since the specific MDP is now precisely identified, the agent can predict the value of any policy, and in particular, the action-value function of the target policy. $\qquad\square$

G. Proof of Theorem 2 and Theorem 4

G.1. MDP Class

We consider a class $\mathcal{M}$ of MDPs sharing the same state-space $\mathcal{S}$, action space $\mathcal{A}$ and discount factor $\gamma > 0$, but different transition function p and reward function r. We first describe the state and action space and the discount factor which are fixed across all MDPs in the class $\mathcal{M}$.

At a high level, each MDP contains a two-armed bandit instance in $s^\star$ (the starting state). There, the learner has two choices: either take the special action $a^\star$ that leads to the terminal state $s^\dagger$ and gives an immediate reward of $\frac{1}{2}$, or take any other action that leads to either an intermediate state $\overline{s}$ or the terminal state $s^\dagger$. In any case, the agent never gets back to $s^\star$.

State space   The state space can written as the union of three states, a starting state $s^\star$, an intermediate state $\overline{s}$ and a terminal state $s^\dagger$. Mathematically:

$$\mathcal{S} = \{s^\star, \overline{s}, s^\dagger\}. \tag{82}$$

Action space   The action space is as follows. In the starting state $s^\star$ the special action $a^\star$ is available in addition to any action in $\mathcal{B}^+$. In $\overline{s}$, any action in $\mathcal{B}^+$ is available. Finally, in the terminal state $s^\dagger$ only $\vec{0} \in \mathcal{B}^+$ is available. Mathematically:

$$\mathcal{A}_s \stackrel{def}{=} \begin{cases} \mathcal{B}^+ \cup \{a^\star\} & \text{if } s = s^\star \\ \mathcal{B}^+ & \text{if } s = \overline{s} \\ \{\vec{0}\} & \text{if } s = s^\dagger. \end{cases} \tag{83}$$

Notice that $\vec{0} \in \mathcal{B}^+$.

Discount factor   The discount factor γ is in the interval $(0,1)$.

Feature map   For this BPI problem the feature map is the $(d+1)$-dimensional vector defined as follows

$$\phi(s,a) = \begin{cases} [a, 0] & \text{if } a \in \mathcal{B}^+ \\ [\vec{0}, 1] & \text{if } a = a^\star \quad \text{(this action is only available in } s^\star\text{).} \end{cases} \tag{84}$$

We notice that we must have $\|\phi(\cdot, \cdot)\|_2 \leq 1$ and that the feature map only depends on the action.

G.2. Instance of the Class

Next we describe the MDP-specific transition and reward functions. Every MDP is identified by a vector $w \in \partial\mathcal{B}^+$ such that $e_i^\top w > 0, \forall i \in [d]$, and by a $+$ and $-$ sign.

Transition Function   The transition function p_w depends on the vector w that identifies each MDP in the class, but not on the sign $+$ or $-$. Fix the MDP by fixing w (two MDPs correspond to a given choice of w). If the agent plays the special action $a^\star$, which is only available in the starting state $s^\star$, it transitions with probability one to the terminal state $s^\dagger$. Otherwise, the transition function is only a function of the action and the successor state is $\overline{s}$ with some probability, and is otherwise the absorbing state $s^\dagger$.

Mathematically, if $a = a^\star$ (which implies $s = s^\star$) then

$$p_w(s' = s^\dagger \mid (s^\star, a^\star)) = 1$$
$$p_w(s' \neq s^\dagger \mid (s^\star, a^\star)) = 0.$$

This is a valid definition in $(s^\star, a^\star)$. If conversely $a \in \mathcal{B}^+$:

$$p_w(s' = \overline{s} \mid (s,a)) = \min\{\frac{1}{\gamma}a^\top w, 1\}$$
$$p_w(s' = s^\dagger \mid (s,a)) = 1 - p_w(s' = \overline{s} \mid (s,a)). \tag{85}$$

Since the probabilities are positive (notice that in particular $a^\top w \geq 0$ since $a \in \mathcal{B}^+$ and $w \in \mathcal{B}^+$) and add up to one, the definition is well posed. In particular, the definition implies that the successor state is always either $\bar{s}$ or the terminal state $s^\dagger$:

$$\forall (s,a): \qquad p_w(s' = s^\star \mid (s,a)) = 0. \tag{86}$$

Reward function The reward function $r_{w,+}$ or $r_{w,-}$ depends on both the vector $w \in \partial \mathcal{B}^+$ and on the sign $+$ or $-$ that identifies the MDP. It is always $\frac{1}{2}$ if the special action $a^\star$ is taken and otherwise it is everywhere 0 on $M_{w,-}$ or is positive only in the hyperspherical cap on $M_{w,+}$. Mathematically, it is defined as follows:

On $M_{w,+}$:

$$\forall (s,a), \quad r_{w,+}(s,a) \overset{def}{=} \begin{cases} \frac{1}{2} & \text{if } (s,a) = (s^\star, a^\star) \\ \max\{a^\top w - \gamma, 0\} & \text{otherwise.} \end{cases} \tag{87}$$

On $M_{w,-}$:

$$\forall (s,a), \quad r_{w,-}(s,a) \overset{def}{=} \begin{cases} \frac{1}{2} & \text{if } (s,a) = (s^\star, a^\star) \\ 0 & \text{otherwise.} \end{cases} \tag{88}$$

Remark 5. *$M_{w,+}$ and $M_{w,-}$ have identical transition functions but different reward functions.*

Remark 6. *The reward functions on $M_{w,+}$ and $M_{w,-}$ differ only when the chosen action is inside the hyperspherical cap $\mathcal{C}_\gamma(w)$.*

G.3. Realizability

We compute the optimal action value function for each MDP in the class M, showing realizability.

Lemma 8 ($Q^\star$ is Realizable). *For any $w \in \partial \mathcal{B}^+$, let $Q^\star_{w,+}$ and $Q^\star_{w,-}$ be the optimal $Q^\star$ values on $M_{w,+}$ and $M_{w,-}$, respectively. Then it holds that*

$$\forall (s,a), \quad \begin{cases} Q^\star_{w,+}(s,a) = \phi(s,a)^\top [w, \frac{1}{2}] & \text{on } M_{w,+}, \\ Q^\star_{w,-}(s,a) = \phi(s,a)^\top [\vec{0}, \frac{1}{2}] & \text{on } M_{w,-}. \end{cases} \tag{89}$$

Proof. We first consider $M_{w,+}$. On any state $\neq s^\dagger$ the optimal policy is to take action $a = w$, achieving a return of $(1-\gamma)$ at every timestep. This way the agent transitions to the state $s = \bar{s}$ and then stays put there playing action $a = w$. This yields a total return of 1. In the terminal state $s^\dagger$ action $\vec{0}$ is the only available and gives a reward of zero with a self loop. Thus:

$$V^\star_{w,+}(s) = \begin{cases} 1 & \text{if } s \neq s^\dagger \\ 0 & \text{if } s = s^\dagger. \end{cases} \tag{90}$$

Now we apply the Bellman operator to $V^\star_{w,+}$ to compute the optimal action-value function on $M_{w,+}$.

If $(s,a) = (s^\star, a^\star)$ then

$$(\mathcal{T}_{w,+} V^\star_{w,+})(s^\star, a^\star) = r_{w,+}(s^\star, a^\star) + \gamma \, \mathbb{E}_{s' \sim p_w(s^\star, a^\star)} V^\star_{w,+}(s') = \frac{1}{2} + 0. \tag{91}$$

If $(s,a) \neq (s^\star, a^\star)$ and $a \notin \mathcal{C}_\gamma(w)$ then by definition $a^\top w < \gamma$ and so

$$(\mathcal{T}_{w,+} V^\star_{w,+})(s,a) = r_{w,+}(s,a) + \gamma \, \mathbb{E}_{s' \sim p_w(s,a)} V^\star_{w,+}(s') = 0 + \gamma \left(\frac{1}{\gamma} a^\top w \cdot 1 + 0 \right) = a^\top w. \tag{92}$$

If $(s,a) \neq (s^\star, a^\star)$ and $a \in \mathcal{C}_\gamma(w)$ then

$$(\mathcal{T}_{w,+} V^\star_{w,+})(s,a) = r_{w,+}(s,a) + \gamma \, \mathbb{E}_{s' \sim p_w(s,a)} V^\star_{w,+}(s') = (a^\top w - \gamma) + \gamma \cdot 1 = a^\top w. \tag{93}$$

This shows that the optimal action-value function on $M_{w,+}$ is

$$Q^\star_{w,+}(s,a) = \begin{cases} [a,0]^\top [w,\frac{1}{2}] & \text{if } (s,a) \neq (s^\star, a^\star) \\ [\vec{0},1]^\top [w,\frac{1}{2}] & \text{if } (s,a) = (s^\star, a^\star) \end{cases} \tag{94}$$

which equals $\phi(s,a)^\top [w,\frac{1}{2}]$.

Now we reason on $M_{w,-}$. In $s^\star$, the optimal policy is play $a^\star$ once and transition to the terminal state $s^\dagger$. If conversely $s \neq s^\star$, the maximum attainable return by any policy is 0. Thus on $M_{w,-}$ the optimal value function reads

$$V^\star_{w,-}(s) = \begin{cases} \frac{1}{2} & \text{if } s = s^\star, \\ 0 & \text{if } s \neq s^\star. \end{cases} \tag{95}$$

Now we apply the Bellman optimality operator to $V^\star_{w,-}$ to obtain the optimal action value function on $M_{w,-}$.

If $(s,a) = (s^\star, a^\star)$

$$(\mathcal{T}_{w,-} V^\star_{w,-})(s^\star, a^\star) = r_{w,-}(s^\star, a^\star) + \gamma\, \mathbb{E}_{s' \sim p_w(s^\star, a^\star)} V^\star_{w,-}(s') = \frac{1}{2} + 0. \tag{96}$$

Conversely, if $(s,a) \neq (s^\star, a^\star)$

$$(\mathcal{T}_{w,-} V^\star_{w,-})(s,a) = r_{w,-}(s,a) + \gamma\, \mathbb{E}_{s' \sim p_w(s,a)} V^\star_{w,-}(s') = 0 = a^\top \vec{0} \tag{97}$$

This shows that on $M_{w,-}$ the optimal action value function is

$$Q^\star_{w,-}(s,a) = \begin{cases} [a,0]^\top [\vec{0},\frac{1}{2}] & \text{if } (s,a) \neq (s^\star, a^\star) \\ [\vec{0},1]^\top [\vec{0},\frac{1}{2}] & \text{if } (s,a) = (s^\star, a^\star) \end{cases} \tag{98}$$

which equals $\phi(s,a)^\top [\vec{0},\frac{1}{2}]$. This concludes the proof. $\qquad \square$

G.4. Proof of Theorem 2

Proof. Consider the MDP class described in Appendix G.1 and Appendix G.2. First, from Lemma 8 *($Q^\star$ is Realizable)* we know that every member of the class satisfies Assumption 2 *($Q^\star$ is Realizable)* with a given feature map $\phi(\cdot, \cdot)$.

Notice that any policy π induces the same state-actions (possibly with the exception of the singletons $(s^\star, a^\star)$ and $(s^\dagger, \vec{0})$) regardless of the MDP that the learner is interacting with. We can therefore consider the case that the oracle has chosen a policy-free query set $\mu = \{(s,a)\}_{i=1,2,\dots,n}$ of size n in addition to the singletons $(s^\star, a^\star)$ and $(s^\dagger, \vec{0})$ (these singletons do not convey any additional information). Here the (s,a) pairs are the possible state-actions visited by the behavioral policies chosen by the oracle.

From Lemma 5 *(Existence of a Lonely Spherical Cone in the Positive Orthant)* we know that if $|\mu| < 2^{-d} I^{-1}_{1-\gamma^2}(\frac{d-1}{2}, \frac{1}{2})$ then $\exists \widetilde{w} \in \partial \mathcal{B}^+$ such that $e_i^\top w > 0, \forall i \in [d]$ and $\forall (s,a) \in \mu, a \notin \mathcal{C}_\gamma(\widetilde{w})$. Consider the two associated MDPs $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$. Notice that the transition function is by construction identical on $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$ while their reward functions are zero at any $(s,a) \in \mu$:

$$\begin{cases} r_{\widetilde{w},+}(s,a) &= r_{\widetilde{w},-}(s,a) \\ p_{\widetilde{w},+}(s,a) &= p_{\widetilde{w},-}(s,a). \end{cases} \tag{99}$$

This implies that the transitions and the rewards in the dataset could have originated from either $M_{\widetilde{w},+}$ or $M_{\widetilde{w},-}$. Thus in $s^\star$, the batch algorithm has two choices to determine $\widehat{\pi}^\star_{\mathcal{D}}$: choose action $a^\star$ and get a total return of $\frac{1}{2}$ or choose an action $a \neq a^\star$. The second choice is $\frac{1}{2}$-suboptimal on $M_{\widetilde{w},-}$, while the first is $\frac{1}{2}$-suboptimal on $M_{\widetilde{w},+}$. At best, the batch algorithm can randomize between the two, showing the result. $\qquad \square$

G.5. Proof of Theorem 4

Proof. Now consider the following adaptive algorithm that submits policy-induced queries. The algorithm first plays $\gamma e_1, \ldots, \gamma e_d$ in $s^\star$ (these are d policies that generate trajectories of length one) to locate the position of the hyperspherical cap (vector w). Then the agent probes the hyperspherical cap to gain knowledge of the reward function, which identifies the sign of the MDP.

Upon playing $\gamma e_1, \ldots, \gamma e_d$ in $s^\star$ the agent receives the transition functions $p_w(s' = \overline{s} \mid (s^\star, \gamma e_i)) = \min\{\frac{1}{\gamma}(\gamma e_i)^\top w, 1\} = e_i^\top w$ for all $i \in [d]$. From this, the agent can determine each entry of the vector w. Next, it can play the state-action $(s^\star, w)$ to probe the hyperspherical cap and to observe the reward ($1 - \gamma$ on $M_{w,+}$ and 0 on $M_{w,-}$) which identifies the sign of the MDP. Since the specific MDP is now precisely identified, the agent can predict the value of any policy, and in particular, it can return the optimal policy. $\qquad\square$

H. Proof of Theorem 3

H.1. MDP Class Definition

We consider a class $\mathcal{M}$ of MDPs sharing the same state-space $\mathcal{S}$, action space $\mathcal{A}$ and discount factor $\gamma > 0$, but different reward function r, transition function p and target policy (for the OPE problem). A feature map ϕ from the state-action space to $\mathbb{R}^d$ is prescribed and is fixed across different MDPs of the class. We first define the state and action space, the discount factor and the feature extractor which are fixed across all MDPs in the class $\mathcal{M}$.

We present *two versions* of the construction, one with *continuous* action space and one with *small action* space. The proof for either case is the same; the two constructions are reported here to highlight the impact of the action space in such lower bound.

State space The state space $\mathcal{S}$ consists of a starting state $s^\star$ and a set of satellite states which can be identified with the unit ball $\mathcal{B}$. Mathematically we can write

$$\mathcal{S} = \{s^\star\} \cup \mathcal{B}, \quad \{s^\star\} \cap \mathcal{B} = \emptyset. \tag{100}$$

Action space (Continuous $\mathcal{A}$) The starting state $s^\star$ has actions in the Euclidean ball; each satellite state has a unique action. Mathematically

$$\mathcal{A}(s) \stackrel{def}{=} \begin{cases} \mathcal{B} & \text{if } s = s^\star \\ \{s\} & \text{if } s \in \mathcal{B}. \end{cases} \tag{101}$$

In particular $\forall s \in \mathcal{S}, \mathcal{A}(s) \subseteq \mathcal{B}$.

Action space (Small $\mathcal{A}$) The starting state $s^\star$ has the canonical vectors in the Euclidean ball as available actions (along with their 'negative counterpart'); each satellite state has a unique action. Mathematically

$$\mathcal{A}(s) \stackrel{def}{=} \begin{cases} E \stackrel{def}{=} \{e_1, \ldots, e_d, -e_1, \ldots, -e_d\} & \text{if } s = s^\star \\ \{s\} & \text{if } s \in \mathcal{B}. \end{cases} \tag{102}$$

In particular $\forall s \in \mathcal{S}, \mathcal{A}(s) \subseteq \mathcal{B}$.

Discount factor The discount factor γ is in the interval $(0, 1)$.

Feature map The feature extractor returns the action chosen in the selected state. Mathematically

$$\phi(s, a) = a. \tag{103}$$

Since the available actions are always a subset of $\mathcal{B}$ (see e.g., Eq. (101)), it is easy to see that the image of the feature map is (contained in) the set $\mathcal{B}$.

Target Policy In the OPE problem, the target policy is identical in every MDP M in the class, and in particular, it returns the only action available in each state $\neq s^\star$. In $s^\star$ it takes action $\vec{0}$.

H.2. Instance of the Class

Every MDP in the class $\mathcal{M}$ can be identified by a vector $w \in \mathcal{B}$ and a sign $+$ or $-$, and is denoted by $M_{w,+}$ or $M_{w,-}$, respectively. Next we describe the transition function and the reward function on M_w and the target policy.

Transition function The transition function depends only on the vector $w \in \mathcal{B}$ that identifies the MDP M_w, but not on the sign $+$ or $-$ that distinguishes $M_{w,+}$ from $M_{w,-}$. For a given MDP M_w, the transition function is deterministic and depends only on the action chosen; it is convenient to represent the only possible successor state $s' \in \mathcal{S}$ by the function $s' = s_w^+(a)$. Mathematically

$$s_w^+(a) = \begin{cases} \frac{1}{\gamma}(a^\top w)w, & \text{if } a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w), \\ a & \text{otherwise.} \end{cases} \tag{104}$$

It is easy to see that the linear algebra operations in the above display are well defined. In addition, $s_w^+(a) \in \mathcal{B} \subset \mathcal{S}$, because $a \in \mathcal{B}$ and if $a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ we have $\|\frac{1}{\gamma}(a^\top w)w\|_2 \leq \frac{|a^\top w|}{\gamma}\|w\|_2 \leq 1$, so $\frac{1}{\gamma}(a^\top w)w \in \mathcal{B}$ as well. This also means that the successor state is never the starting state $s^\star$.

Reward function   The reward function depends on both the vector $w \in \mathcal{B}$ and on the sign $+$ or $-$ that identifies the MDP and it is defined as follows:

$$\text{on } M_{w,+} : \quad r_{w,+}(s,a) \stackrel{def}{=} \begin{cases} 0 & \text{if } a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w) \\ (1-\gamma)a^\top(+w) & \text{otherwise} \end{cases} \tag{105}$$

$$\text{on } M_{w,-} : \quad r_{w,-}(s,a) \stackrel{def}{=} \begin{cases} 0 & \text{if } a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w) \\ (1-\gamma)a^\top(-w) & \text{otherwise} \end{cases} \tag{106}$$

$$\tag{107}$$

Remark 7. $M_{w,+}$ and $M_{w,-}$ have identical transition functions but opposite reward functions.

Remark 8. $M_{w,+}$ and $M_{-w,-}$ have identical transition and reward functions i.e., they are the same MDP $M_{w,+} = M_{-w,-}$. Likewise, we have $M_{w,-} = M_{-w,+}$.

H.3. Realizability

For each MDP in the class, we compute the action value function of an arbitrary policy π, showing realizability.

Lemma 9 (Realizability: Verifying assumption 3). *For any vector $w \in \partial\mathcal{B}$, and any policy π, let $Q_{w,+}^\pi$ and $Q_{w,-}^\pi$ be the action-value functions of π on $M_{w,+}$ and $M_{w,-}$, respectively. Then it holds that*

$$\forall(s,a), \quad \begin{cases} Q_{w,+}^\pi(s,a) = \phi(s,a)^\top(+w) & \text{on } M_{w,+}, \\ Q_{w,-}^\pi(s,a) = \phi(s,a)^\top(-w) & \text{on } M_{w,-}. \end{cases} \tag{108}$$

Proof. Let $\mathcal{T}_{w,+}^\pi$ and $\mathcal{T}_{w,-}^\pi$ be the Bellman evaluation operators for π on $M_{w,+}$ and $M_{w,-}$, respectively. We need to show that the proposed solutions in Eq. (108) satisfy the Bellman evaluation equations at all (s,a) pairs.

First, we focus on $M_{w,+}$ and apply the Bellman operator $\mathcal{T}_{w,+}^\pi$ to $Q_{w,+}^\pi$.

If $a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then notice that the successor state is $s^+(a) = \frac{1}{\gamma}(a^\top w)w$. Furthermore, π must return the only action available in the successor state (and the successor state is never $s^\star$). Thus we can write

$$(\mathcal{T}_{w,+}^\pi Q_{w,+}^\pi)(s,a) = r_{w,+}(s,a) + \gamma Q_{w,+}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{109}$$

$$= 0 + \gamma Q_{w,+}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{110}$$

$$= \gamma Q_{w,+}^\pi(s_w^+(a), s_w^+(a)) \tag{111}$$

$$= \gamma \phi(s_w^+(a), s_w^+(a))^\top(+w) \tag{112}$$

$$= \gamma(s_w^+(a))^\top(+w) \tag{113}$$

$$= \gamma \left(\frac{1}{\gamma}(a^\top w)w\right)^\top (+w) \tag{114}$$

$$= a^\top w \tag{115}$$

$$= \phi(s,a)^\top w \tag{116}$$

$$= Q_{w,+}^\pi(s,a). \tag{117}$$

If conversely $a \in \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then $s_w^+(a) = a$, and as before the policy can only take the only action available there,

giving

$$(\mathcal{T}_{w,+}^\pi Q_{w,+}^\pi)(s,a) = (1-\gamma)a^\top(+w) + \gamma Q_{w,+}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{118}$$

$$= (1-\gamma)a^\top(+w) + \gamma Q_{w,+}^\pi(a, \pi(a)) \tag{119}$$

$$= (1-\gamma)a^\top(+w) + \gamma\phi(a,a)^\top(+w) \tag{120}$$

$$= (1-\gamma)a^\top(+w) + \gamma a^\top(+w) \tag{121}$$

$$= a^\top(+w) \tag{122}$$

$$= \phi(s,a)^\top(+w) \tag{123}$$

$$= Q_{w,+}^\pi(s,a). \tag{124}$$

This shows that $Q_{w,+}^\pi(s,a) = \phi(s,a)^\top(+w)$ is the value of π on $M_{w,+}$.

The argument to verify that $Q_{w,-}^\pi = \phi(s,a)^\top(-w)$ is the value of π on $M_{w,-}$ is identical, as follows.

If $a \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then

$$(\mathcal{T}_{w,-}^\pi Q_{w,-}^\pi)(s,a) = r_{w,-}(s,a) + \gamma Q_{w,-}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{125}$$

$$= 0 + \gamma Q_{w,-}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{126}$$

$$= \gamma Q_{w,-}^\pi(s_w^+(a), s_w^+(a)) \tag{127}$$

$$= \gamma(s_w^+(a))^\top(-w) \tag{128}$$

$$= \gamma\left(\frac{1}{\gamma}(a^\top w)w\right)^\top(-w) \tag{129}$$

$$= a^\top(-w) \tag{130}$$

$$= \phi(s,a)^\top(-w). \tag{131}$$

Otherwise, if $a \in \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$ then

$$(\mathcal{T}_{w,-}^\pi Q_{w,-}^\pi)(s,a) = (1-\gamma)a^\top(-w) + \gamma Q_{w,-}^\pi(s_w^+(a), \pi(s_w^+(a))) \tag{132}$$

$$= (1-\gamma)a^\top(-w) + \gamma Q_{w,-}^\pi(a, \pi(a)) \tag{133}$$

$$= (1-\gamma)a^\top(-w) + \gamma a^\top(-w) \tag{134}$$

$$= a^\top(-w) \tag{135}$$

$$= \phi(s,a)^\top(-w). \tag{136}$$

$\square$

H.4. Proof of Theorem 3

Proof. Consider the MDP class described in Appendix H.1 and Appendix H.2. First, from Lemma 9 *(Realizability: Verifying assumption 3)* we know that every MDP in $\mathcal{M}$ satisfies Assumption 3 *(Q^π is Realizable for every Policy)* with a given feature map $\phi(\cdot, \cdot)$.

Let μ be the policy-free query set chosen by the oracle. Using Lemma 6 *(Existence of two Symmetric Hyperspherical Cones)* we can claim that if $|\mu| < I_{1-\gamma^2}^{-1}(\frac{d-1}{2}, \frac{1}{2})$ then $\exists \widetilde{w} \in \partial\mathcal{B}$ such that $\forall (s,a) \in \mu, a \notin \mathcal{C}_\gamma(\widetilde{w}) \cup \mathcal{C}_\gamma(-\widetilde{w})$. Consider the two associated MDPs $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$. Notice that the transition function is by construction identical on $M_{\widetilde{w},+}$ and $M_{\widetilde{w},-}$ while their reward function is zero at any $(s,a) \in \mu$:

$$\begin{cases} r_{\widetilde{w},+}(s,a) &= r_{\widetilde{w},-}(s,a) \\ p_{\widetilde{w},+}(s,a) &= p_{\widetilde{w},-}(s,a). \end{cases} \tag{137}$$

This implies that the reward and transition functions in the dataset could have originated from either $M_{\widetilde{w},+}$ or $M_{\widetilde{w},-}$. In addition, the target policy is fixed for all MDPs and so it does not reveal the MDP instance.

Conclusion for continuous action space In this section we focus on the construction with continuous actions space.

Consider the best policy identification problem. In $s^\star$ the algorithm has two choices to determine $\widehat{\pi}_{\mathcal{D}}^\star$: choose an action a such that $a^\top \widetilde{w} > 0$ or $a^\top \widetilde{w} \leq 0$. In the first case, it obtains negative return on $M_{\widetilde{w},-}$ and in the second case it obtains negative return on $M_{\widetilde{w},+}$. However, the value of the optimal policy in $s^\star$ is $+1$ in both cases. Even if the batch algorithm randomizes the output, with probability at least $1/2$ the returned policy is at least 1-suboptimal.

Likewise, consider the off-policy evaluation problem where π is the target policy. The batch algorithm has two choices to estimate $Q_M^\pi(s^\star, \widetilde{w})$: either a positive or a negative value. However $Q_{M_{\widetilde{w},+}}^\pi(s^\star, \widetilde{w})$ equals $+1$ and $Q_{M_{\widetilde{w},-}}^\pi(s^\star, \widetilde{w})$ equals -1. At best, the batch algorithm can randomize between a positive and a negative value, making an error of at least 1 with probability at least $1/2$.

Conclusion for small action space In this section we focus on the construction with small actions space.

Now Lemma 10 *(Sufficient Inner Product)* ensures that that for any $\widetilde{w}, \|\widetilde{w}\|_2 = 1$ there exists an action $e_j \in \mathcal{A}(s^\star)$ such that either $e_j^\top \widetilde{w} \geq \frac{1}{\sqrt{d}}$ or $-e_j^\top \widetilde{w} \geq \frac{1}{\sqrt{d}}$; define the action in $s^\star$ most aligned with $\widetilde{w}$ to be $\widetilde{e} = \arg\max_{e \in \mathcal{A}(s^\star)} e^\top \widetilde{w} \geq \frac{1}{\sqrt{d}}$.

Consider the best policy identification problem. In $s^\star$ the algorithm has two choices to determine $\widehat{\pi}_{\mathcal{D}}^\star$: choose an action a such that $a^\top \widetilde{w} > 0$ or $a^\top \widetilde{w} \leq 0$. In the first case, it obtains negative return on $M_{\widetilde{w},-}$ and in the second case it obtains negative return on $M_{\widetilde{w},+}$. However, the value of the optimal policy in $s^\star$ is at least $\widetilde{e}^\top \widetilde{w} \geq \frac{1}{\sqrt{d}}$ on $M_{w,+}$ and at least $(-\widetilde{e})^\top(-\widetilde{w}) \geq \frac{1}{\sqrt{d}}$ on $M_{w,-}$ (notice that $\widetilde{e}$ is available in the construction with small action space). Even if the batch algorithm randomizes the output, with probability at least $1/2$ the returned policy is at least $\frac{1}{\sqrt{d}}$-suboptimal.

Likewise, consider the off-policy evaluation problem where π is the target policy. The batch algorithm has two choices to estimate $Q_M^\pi(s^\star, \widetilde{e})$: either a positive or a negative value. However $Q_{M_{\widetilde{w},+}}^\pi(s^\star, \widetilde{e}) \geq \frac{1}{\sqrt{d}}$ and $Q_{M_{\widetilde{w},-}}^\pi(s^\star, \widetilde{e}) \leq -\frac{1}{\sqrt{d}}$. At best, the batch algorithm can randomize between a positive and a negative value, making an error of at least $\frac{1}{\sqrt{d}}$ with probability at least $1/2$.

Learning with an adaptive or online algorithm (continuous action space) Now consider the following adaptive algorithm that submits policy-free queries (the algorithm that submits policy-induced queries is analogous). At a high level, the algorithm first plays $\frac{\gamma}{2}e_1, \ldots, \frac{\gamma}{2}e_d$ in $s^\star$. This is a basis for $\mathbb{R}^d$ and by construction it does not intersect any hyperspherical cap. However, this allows the agent to locate the position of the hyperspherical caps in the given MDP (vector w). Then the agent probes either hyperspherical cap to gain knowledge of the reward function, which identifies the sign of the MDP.

More formally, consider an MDP $M_w, w \in \partial\mathcal{B}$ and notice that none of the query actions (described above) is located in an hyperspherical cap of height γ: we have $|(\frac{\gamma}{2}e_i)^\top w| \leq \frac{\gamma}{2}\|e_i\|_2\|w\|_2 < \gamma$. In addition, $e_1, \ldots, e_d$ is a basis for $\mathbb{R}^d$ which implies that $w \neq 0$ cannot be orthogonal to all of them. Therefore, there must exist an index $j \in [d]$ such that $e_j^\top w \neq 0$. Since $\frac{\gamma}{2}e_j \notin \mathcal{C}_\gamma(w) \cup \mathcal{C}_\gamma(-w)$, the successor state $s_w^+(\frac{\gamma}{2}e_j)$ must be non-zero and must be parallel to w. Thus, the agent knows the vector w of the MDP (up to a sign). Next, it can play $(s^\star, w)$ to probe one of the two hyperspherical caps and to observe the reward $\pm(1-\gamma)$ which identifies the sign of the MDP using Remark 8. Since the specific MDP is now precisely identified, the agent can predict the value of any policy, and therefore, it can return an optimal policy. $\square$

H.5. Helper Lemma

To prove Theorem 3 *(Policy-Free Lower Bounds)* with small action space we need the following helper lemma.

Lemma 10 (Sufficient Inner Product). *Let $\{e_1, \ldots, e_d\} \subseteq \mathbb{R}^d$ be the set of canonical vectors and $\widetilde{w} \in \mathbb{R}^d, \|\widetilde{w}\|_2 = 1$. Then there exists a canonical vector e_j such that*

$$|e_j^\top \widetilde{w}| \geq \frac{1}{\sqrt{d}}. \tag{138}$$

Proof. Let $\widetilde{w}_j$ be the jth component of $\widetilde{w}$. From the hypothesis we must have

$$1 = \|\widetilde{w}\|_2^2 = \sum_{j=1}^d \widetilde{w}_j^2. \tag{139}$$

This implies that at least one component $\widetilde{w}_j$ for some $j \in [d]$ must be greater (in absolute value) than $\frac{1}{\sqrt{d}}$, i.e.,

$$\exists j \in [d] \quad \text{such that} \quad |\widetilde{w}_j| \geq \frac{1}{\sqrt{d}} \tag{140}$$

otherwise $\|\widetilde{w}\|_2 < 1$ in Eq. (139), contradiction. Since $e_1, \ldots, e_j$ are the canonical vectors, the thesis $|e_j^\top \widetilde{w}| = |\widetilde{w}_j| \geq \frac{1}{\sqrt{d}}$ now follows. $\qquad\square$