Computationally Efficient PAC RL in POMDPs with Latent Determinism and Conditional Embeddings

Masatoshi Uehara [1] **Ayush Sekhari** [1] **Jason D. Lee** [2] **Nathan Kallus** [1] **Wen Sun** [1]

Abstract

We study reinforcement learning with function approximation for large-scale Partially Observable Markov Decision Processes (POMDPs) where the state space and observation space are large or even continuous. Particularly, we consider Hilbert space embeddings of POMDP where the feature of latent states and the feature of observations admit a conditional Hilbert space embedding of the observation emission process, and the latent state transition is deterministic. Under the function approximation setup where the optimal latent state-action Q-function is linear in the state feature, and the optimal Q-function has a gap in actions, we provide a *computationally and statistically efficient* algorithm for finding the *exact optimal* policy. We show our algorithm's computational and statistical complexities scale polynomially with respect to the horizon and the intrinsic dimension of the feature on the observation space. Furthermore, we show both the deterministic latent transitions and gap assumptions are necessary to avoid statistical complexity exponential in horizon or dimension. Since our guarantee does not have an explicit dependence on the size of the state and observation spaces, our algorithm provably scales to large-scale POMDPs.

1. Introduction

In reinforcement learning (RL), we often encounter partial observability of states (Kaelbling et al., 1998). Partial observability poses a serious challenge in RL from both computational and statistical aspects since observations are no longer Markovian. From a computational perspective, even if we know the dynamics, planning problems in POMDPs (partially observable Markov decision process) are known to be NP-hard (Papadimitriou & Tsitsiklis, 1987). From a statistical perspective, an exponential dependence on horizon in sample complexity is not avoidable without further assumptions (Jin et al., 2020a).

We consider computationally and statistically efficient learning on large-scale POMDPs with deterministic transitions (but *stochastic* emissions). Here large-scale means that the POMDP might have large or even continuous state and observation spaces, but they can be modeled using conditional embeddings. Deterministic transitions and stochastic emissions is a very practically-relevant setting. For example, in robotic control, the dynamics of the robot itself is often deterministic but the observation of its current state is distorted by noise on the sensors (Platt Jr et al., 2010; Platt et al., 2017). In human-robot-interaction, the robots' dynamics is often deterministic, while human's actions can be modeled as uncertain observations emitted from a distribution conditioned on robot's state and human's pre-fixed goal (Javdani et al., 2015). For autonomous driving, the dynamics of the car in the 2d space is deterministic under normal road conditions, while the sensory data (e.g. GPS, IMU data, and Lidar scans) is modeled as stochastic. Besse & Chaib-Draa (2009) offers further practical examples such as diagnosis of systems (Pattipati & Alexandridis, 1990) and sensor management (Ji et al., 2007). With known deterministic transitions, we can obtain positive results from computational perspectives for optimal planning (Littman, 1996; Bonet, 2012). However, when transitions are unknown, learning algorithms that enjoy both computation and statistical efficiency are still limited to the tabular setting (Jin et al., 2020a).

To design provably efficient RL algorithms for POMDPs with large state and observation spaces, we need to leverage function approximation. The key question that we aim to answer here is, *under what structural conditions of the POMDPs, can we perform RL with function approximation with both statistical and computational efficiency?* Specifically, we consider Hilbert space embeddings of POMDPs (HSE-POMDPs), where both features on the observations and latent states live in reproducing kernel Hilbert spaces (RKHSs), which are equipped with conditional embedding operators and the operators have non-zero singular values (Boots et al., 2011). This assumption is similarly used in

*Equal contribution [1]Cornell University [2]Princeton University. Correspondence to: Masatoshi Uehara <uehara-masatoshi136@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

	Computationally Efficient	Deterministic Transition	Non-tabular	Model-based or Model-free
Guo et al. (2016); Azizzadenesheli et al. (2016)				
Jin et al. (2020a); Li et al. (2021)	No	No	No	Model-based
Cai et al. (2022)	No	No	Yes	Model-based
Jin et al. (2020b)	Yes	Yes	No	Model-based
Our work	Yes	Yes	Yes	Model-free

Table 1. Summary of our work and representative existing works tackling statistically efficient learning on POMDPs. For details and additional works, refer to Section 1.1. Note that deterministic transition does not preclude *stochastic* emissions and rewards.

prior works such as learning HSE hidden Markov models (Song et al., 2010) and HSE predictive state representations (HSE-PSRs) (Boots et al., 2013), where both have demonstrate that conditional embeddings are applicable to real-world applications such as estimating the dynamics of car from IMU data and estimating the configurations of a robot arm with raw pixel images. Also, HSE-POMDPs naturally capture undercomplete tabular POMDPs (Jin et al., 2020a). For HSE-POMDPs with deterministic latent transition, we show positive results under the function approximation setting where the optimal Q-function over latent state and action is linear in the state feature, and the optimal Q-function has a non-trivial gap in the action space.

Our key contributions are as follows. Under the aforementioned setting, we propose an algorithm that learns the exact optimal policy with computational complexity and statistical complexity both polynomial in the horizon and intrinsic dimension (information gain) of the features. Notably, the complexity has no explicit dependence on the size of the problem including the sizes of the state and observation space, thus provably scaling to large-scale POMDPs. In particular, to the best of the author's knowledge, our algorithm is the *first* algorithm that enjoys both statistical and computational efficiency in certain large-scale POMDPs as summarized in Table 1. Our algorithm leverages a key novel finding that the linear optimal Q-function in the latent state's feature together with the existence of the conditional embedding operator implies that the Q-function's value can be estimated using new features constructed from the (possibly multiple-step) future observations, which are observable quantities. Our simple model-free algorithm operates completely using observable quantities and never tries to learn latent state transition and observation emission distribution, unlike existing works (Liu et al., 2022b; Guo et al., 2016; Azizzadenesheli et al., 2016). We also provide lower bounds indicating that in order to perform statistically efficient learning in POMDPs under linear function approximation, we need both the gap condition and the deterministic latent transition condition.

1.1. Related Works

We here review related works. A summary is in Table 1.

Computational challenge in POMDPs. The seminal work (Papadimitriou & Tsitsiklis, 1987) showed finding the optimal policy in POMPDPs is PSPACE-hard. Even worse, finding ϵ-near optimal policy is PSPACE-hard, and finding the best memoryless policy is NP-hard (Littman, 1994; Burago et al., 1996). Recently (Golowich et al., 2022) showed that quasi-polynomial time planning is attainable under the weak-observability assumption (Even-Dar et al., 2007), which is similar to our assumption. However, their lower bound suggests that polynomial computation is still infeasible. Representative models that permit us to get polynomial complexity results are POMDPs where transitions are deterministic, but stochastic emissions (Littman, 1996; Besse & Chaib-Draa, 2009; Jin et al., 2020a). We also consider deterministic latent transition with stochastic emission, but with function approximation.

Statistically efficient online learning in POMDPs. (Even-Dar et al., 2005; Kearns et al., 1999) proposed algorithms that have A^H-type sample complexity, which is prohibitively large in the horizon. (Guo et al., 2016; Azizzadenesheli et al., 2016; Xiong et al., 2021; Jin et al., 2020a; Liu et al., 2022b) show favorable sample complexities in the tabular setting using a model-based spectral learning framework (Hsu et al., 2012). Jin et al. (2020a) additionally showed a computationally and statistically efficient algorithm for tabular POMDPs with deterministic transitions. However, their algorithm crucially relies on the discreteness of the latent state space and it is unclear how to extend it to continuous settings. Also, our approach is model-free so does not need to model the emission process, which itself is an extremely challenging task when observations are high dimensional (e.g., raw-pixel images). In the non-tabular setting, there are many works on learning uncontrolled dynamical systems in HSE-POMDPs (Song et al., 2010; Boots et al., 2013). These existing works do not tackle the challenge of strategic exploration in online RL. Recent work (Cai et al., 2022) shows guarantees for related models in which the transition and observation dynamics are modeled by linear mixture models; however, their approach is computationally inefficient. We remark there are further works tackling online RL in other POMDPs, such as LQG (Lale et al., 2021; Simchowitz et al., 2020), latent POMDPs (Kwon et al.,

2021), reactive POMDPs (Krishnamurthy et al., 2016; Jiang et al., 2017) and unified algorithms (Uehara et al., 2022; Zhong et al., 2022; Chen et al., 2022; Zhan et al., 2022). However, their algorithms are not computationally efficient.

RL with linear $Q^\star$ in MDPs. There is a large body of literature on RL under the linear $Q^\star$-assumption with deterministic transitions (Wen & Van Roy, 2013; Du et al., 2019; 2020). The most relevant work is Du et al. (2020). For a detailed comparison, refer to Remark 1.

2. Preliminaries

We introduce POMDPs, HSE-POMDPs, and our primary assumptions.

2.1. Partially Observable Markov Decision Processes

We consider an episodic POMDP given by the tuple $(H, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathbb{T}, \mathbb{O}, \mathbb{Y}, s_0)$. Here, H is the number of steps in each episode, $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions with $|\mathcal{A}| = A$, $\mathcal{O}$ is the set of observations, $\mathbb{T} = \{\mathbb{T}_h\}_{h=0}^{H-1}$ is the transition dynamics such that $\mathbb{T}_h$ is a map from $\mathcal{S} \times \mathcal{A}$ to $\Delta(\mathcal{S})$, $\mathbb{O} = \{\mathbb{O}_h\}_{h=0}^{H-1}$ is the set of emission distributions such that $\mathbb{O}_h$ is a map from $\mathcal{S}$ to $\Delta(\mathcal{O})$, $\mathbb{Y} = \{\mathbb{Y}_h\}_{h=0}^{H-1}$ is the set of reward distributions such that $\mathbb{Y}_h$ is a map from $\mathcal{S} \times \mathcal{A}$ to $\Delta(\mathbb{R})$, and s_0 is a fixed initial state. We denote the conditional mean of the reward distribution by $r_h : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and the noise by τ, so that $r_h(s,a) + \tau$ has law $\mathbb{Y}_h(s,a)$. We suppose that $\mathbb{Y}_h(s,a)$ lies in $[0,1]$.

In a POMDP, states are not observable to agents. Each episode starts from s_0 at $h = 0$. At each step $h \in [H]$, the agent observes $o_h \in \mathcal{O}$ generated from the hidden state $s_h \in \mathcal{S}$ following $\mathbb{O}_h(\cdot \mid s_h)$, the agent picks an action $a_h \in \mathcal{A}$, receives a reward r_h following $\mathbb{Y}_h(\cdot \mid s, a)$, and then transits to the next latent state $s_{h+1} \sim \mathbb{T}_h(\cdot|s_h, a_h)$.

We streamline the notation as follows. We let $o_{0:h}$ denote $o_0, \ldots, o_h$, and similarly for $a_{0:h}$. Given a matrix A, let $\eta(A)$ be its smallest singular value. Given a vector a and matrix A, let $\|a\|_A^2 = a^\top A a$. Give vectors a and b, we define $\langle a, b\rangle = a^\top b$, $[H] := [0, \cdots, H-1]$.

2.2. Hilbert Space Embedding POMDPs

We introduce our model, HSE-POMDPs. For ease of presentation, we first focus on the finite-dimensional setting with 1-step observability, i.e., using one-step future observation for constructing the conditional mean embedding. We extend to infinite-dimensional RKHS in Section B. We extend to multiple-step future observations in Section 5.

Consider two features, one on the observation, $\psi : \mathcal{O} \to \mathbb{R}^d$, and one on latent state, $\phi : \mathcal{S} \to \mathbb{R}^{d_s}$.

Assumption 1 (Existence of linear conditional mean embedding and left invertibility)**.** *Assume $\forall h \in [H]$, there exists a left-invertible matrix $G_h \in \mathbb{R}^{d \times d_s}$ s.t. $\mathbb{E}_{o \sim \mathbb{O}_h(s)}[\psi(o)] = G_h \phi(s)$.($\exists G_h^\dagger \in \mathbb{R}^{d_s \times d}$ s.t. $G_h^\dagger G_h = I$).*

The left invertible condition is equivalent to saying that G_h is full column rank (it also requires that $d \geq d_s$). This assumption is widely used in the existing literature on learning uncontrolled partially observable systems (Song et al., 2010; 2013; Boots et al., 2013). Later, this is relaxed in Section 5 to permit for the case $d \leq d_s$. Furthermore, we permit the case where $d_s = \infty, d = \infty$ as later formalized in Section B. Finally, note our assumption is different from the assumption in block MDPs (Du et al., 2019). In contrast to block MDPs, we cannot generally decode latent states from observations. We present two concrete examples below.

Example 1 (Undercomplete tabular POMDPs)**.** *Let $d_s = |\mathcal{S}|$ and $d = |\mathcal{O}|$, define ϕ and ψ as one-hot encoding vectors over $\mathcal{S}$ and $\mathcal{O}$, respectively. We overload notation and denote $\mathbb{O}_h \in \mathbb{R}^{d \times d_s}$ as a matrix with entry (i,j) equal to $\mathbb{O}_h(o = i|s = j)$. This $\mathbb{O}_h$ corresponds to G_h. Assumption 1 is satisfied if $\mathbb{O}_h$ is full column rank. This assumption is used in Hsu et al. (2012) for learning tabular undercomplete HMMs. We discuss the overcomplete case $|\mathcal{O}| \leq |\mathcal{S}|$ in Section 5.*

Example 2 (Gaussian POMDPs with discrete latent states (Liu et al., 2022a))**.** *Suppose $\mathcal{S}$ is discrete but $\mathcal{O}$ is continuous, and $\mathbb{O}_h(\cdot \mid s) = \mathcal{N}(\mu_{s,h}, I)$. Then, letting $O_h = [\mu_{1:h}, \cdots, \mu_{|\mathcal{S}|:h}] \in \mathbb{R}^{d \times |\mathcal{S}|}$ and $\phi(\cdot)$ be a one-hot encoding vector over $\mathcal{S}$, we have $\mathbb{E}_{o \sim \mathbb{O}_h(\cdot|s)}[o] = O_h \phi(s)$. Assumption 1 is satisfied when O_h is full-column rank, i.e., the means of the Gaussian distributions are linearly independent.*

2.3. Assumptions and function approximation

We introduce three additional assumptions: deterministic transitions, linear $Q^\star$, and the existence of an optimality gap. The first assumption is as follows.

Assumption 2 (Systems with deterministic state transitions and initial distributions)**.** *The transition dynamics $\mathbb{T}_h$ is deterministic, i.e., there exists a mapping $p_h : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ s.t. $\mathbb{T}_h(\cdot \mid s, a)$ is Dirac at $p_h(s,a)$. The initial state is deterministic.*

Notice Assumption 2 ensures the globally optimal policy $\pi^\star = \operatorname{argmax}_\pi \mathbb{E}_\pi[\sum_h r_h]$ is given as a sequence of (non-history-dependent and deterministic) actions $a^\star_{0:H-1}$.

Here, we stress three points. First, rewards and emission probabilities can still be *stochastic*. Second, Assumption 2 is standard in the literature on MDPs (Wen & Van Roy, 2013; Krishnamurthy et al., 2016; Du et al., 2020; Dann et al., 2018) and POMDPs (Bonet, 2012; Littman, 1996) . As mentioned in Section 1, this setting is practical in many real-world applications. Third, while we can consider learning about POMDPs with stochastic transitions, even if we know the transitions and focus on planning, computing a near-optimal policy is PSPACE-hard (Papadimitriou & Tsitsiklis, 1987). This implies we must need additional

conditions. Deterministic transitions can be regarded as one such possible condition, in particular one that is relevant to many real-world applications as discussed in Section 1.

Next, we suppose the optimal latent Q-function is linear in the state feature. Given a level $h \in [H]$ and state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, the optimal Q-function on the latent state is recursively defined as $Q_h^\star(s,a) = r(s,a) + \max_{a'} Q_{h+1}^\star(p_h(s,a),a')$ starting from $Q_H^\star(s,a) = 0, \forall s,a$. We define $V_h^\star(s) = \max_a Q_h^\star(s,a)$. Now we are ready to introduce the linearity assumption.

Assumption 3 (Linear $Q^\star$)**.** *Given the state feature $\phi : \mathcal{S} \to \mathbb{R}^{d_s}$, for any $h \in [H]$ and any $a \in [A]$, there exists $w_{a;h}^\star \in \mathbb{R}^{d_s}$ such that $Q_h^\star(s,a) = \langle w_{a;h}^\star, \phi(s)\rangle$ for any $s \in \mathcal{S}$ and $\|w_{a;h}^\star\| \le W$.*

This assumption is widely used in RL (Du et al., 2019; 2020; Li et al., 2021; Du et al., 2021). We further consider the infinite-dimensional case in Section B.

Next, we assume an optimality gap. For any $h \in [H]$, define $\mathrm{gap}_h(s,a) = V_h^\star(s) - Q_h^\star(s,a)$.

Assumption 4 (Optimality Gap)**.** $\min_{(h,s,a)}\{\mathrm{gap}_h(s,a) : \mathrm{gap}_h(s,a) > 0\} \ge \Delta$ *for some* $\Delta > 0$.

This assumption is extensively used in bandits (Auer et al., 2002; Dani et al., 2008) and RL (Du et al., 2019; 2020; Simchowitz & Jamieson, 2019; Li et al., 2021; He et al., 2021; Lykouris et al., 2021; Hu et al., 2021; Wu et al., 2022). For its plausibility refer to Du et al. (2019).

3. Lower Bounds

Before presenting our algorithm, we show via lower bounds that Assumptions 2 and 4 are each minimal by themselves, meaning that if we omit just one of them and make no further assumptions then we cannot learn with polynomial sample complexity. All details are deferred to Appendix A.

We first consider the role of Assumption 4 and show that one can *not* hope to learn with sample complexity that scales as $\mathrm{poly}(\min(d,H))$ under latent determinism and linear $Q^\star$, but no gap.

Theorem 1 (Optimality-gap assumption is minimal)**.** *Let d, H be sufficiently large constants, and consider any online learning algorithm ALG. Then, there exists state and observation feature vectors $\phi : \mathcal{S} \to R^d$ and $\psi : \mathcal{O} \to R^d$ with $\max_{s,o}\{\|\phi(s)\|, \|\psi(o)\|\} \le 1$, and a POMDP $(H, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathbb{T}, \mathbb{O}, \mathbb{Y}, s_0)$ that satisfies Assumptions 1, 2, and 3 with respect to features ψ and ϕ such that with probability at least $1/10$ ALG requires at least $\frac{1}{d^{1/5} \wedge H^{1/2}} 2^{\Omega\left(d^{1/5} \wedge H^{1/2}\right)}$ many samples to return a $1/10$ suboptimal policy for this POMDP.*

Theorem 1 is proved by lifting the construction in (Weisz et al., 2021) to the POMDP setting, and consists of an underlying deterministic dynamics (on the state space) with stochastic rewards.

In our next result, we consider the role of Assumption 2 and show that one cannot hope to learn with sample complexity that scales as $\mathrm{poly}(\min(d,H))$ if the underlying state space dynamics is stochastic even if all other assumptions hold. Here we take the linear $Q^\star$ lower bound MDP construction from Wang et al. (2021) and lift it to a POMDP by simply treating the original MDP's state as observation.

Theorem 2 (Deterministic state space dynamics assumption is minimal)**.** *Let d, H be sufficiently large constants and consider any online learning algorithm ALG. Then, there exists state and observation feature vectors $\phi : \mathcal{S} \to R^d$ and $\psi : \mathcal{O} \to R^d$ with $\max_{s,o}\{\|\phi(s)\|, \|\psi(o)\|\} \le 1$, and a POMDP $(H, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathbb{T}, \mathbb{O}, \mathbb{Y}, s_0)$ that satisfies Assumptions 1, 3 and 4 w.r.t features ψ and ϕ such that with probability at least $1/10$ ALG requires at least $\Omega\left(2^{\Omega(\min\{d,H\})}\right)$ many samples to return a $1/20$ suboptimal policy for this POMDP.*

The above two results indicate that neither latent determinism nor gap condition alone can ensure statistically efficient learning, so our assumptions are minimal. In the next section, we show that efficient PAC learning is possible when latent determinism and gap conditions are combined.

4. Algorithm for HSE-POMDPs

In this section, we discuss the case where features are finite-dimensional and propose a new algorithm. Before presenting our algorithm, we review some useful observations.

When latent transition dynamics and initial states are deterministic, given any sequence $a_{0:h-1}$, the latent state that it reaches is fixed. We denote the latent state corresponding to $a_{0:h-1}$ as $s_h(a_{0:h-1})$. Since latent states are not observable, even if we knew $Q_h^\star(s,a), \forall s,a$, we cannot extract the optimal policy easily since during execution we never observe a latent state s_h. To overcome this issue, we leverage the existence of left-invertible linear conditional mean embedding operator in Assumption 1: $Q_h^\star(s,a)$ is equal to

$$\langle w_{a;h}^\star, G_h^\dagger G_h \phi(s)\rangle = \langle \{G_h^\dagger\}^\top w_{a;h}^\star, \mathbb{E}_{o\sim\mathbb{O}_h(s)}[\psi(o)]\rangle.$$

Hence, letting $\theta_{a;h}^\star = \{G_h^\dagger\}^\top w_{a;h}^\star$, the function $Q_h^\star(s,a)$ is linear in a new latent-state feature $\mathbb{E}_{o\sim\mathbb{O}_h(s)}[\psi(o)]$. By leveraging the determinism in the latent transition, given a sequence of actions $a_{0:h-1}$, we can estimate the observable feature $x_h(a_{0:h-1}) := \mathbb{E}_{o\sim\mathbb{O}_h(s_h(a_{0:h-1}))}[\psi(o)]$ by repeatedly executing $a_{0:h-1}$ M times from the beginning, recording the M i.i.d observations $\{o^{(i)}\}_{i=1}^M$ generated from $\mathbb{O}_h(\cdot|s_h(a_{0:h-1}))$, resulting in an estimator defined as: $\hat{x}_h(a_{0:h-1}) = \sum_j \psi(o^{(j)})/M$. Now, if we knew $\theta_{a;h}^\star$, we could consistently estimate $Q_h^\star(s_h(a_{0:h-1}),a)$ by ${\theta_{a;h}^\star}^\top \hat{x}_h(a_{0:h-1})$ using the observable quantity $\hat{x}_h(a_{0:h-1})$.

The remaining challenge is to learn $\theta^\star_{a;h}$. At high-level, if we knew $V^\star_{h+1}(s_{h+1}(a_{0:h-1}, a))$, then we can estimate $\theta^\star_{a;h}$ by regressing target $r_h(s_h(a_{0:h-1}), a) + V^\star_{h+1}(s_{h+1}(a_{0:h-1}, a))$ on the feature $\hat{x}_h(a_{0:h-1})$. Below, we present our recursion based algorithm that recursively estimates $V^\star_h$ and also performs exploration at the same time.

4.1. Algorithm

We present the description of our algorithm. The algorithm is divided into two parts: Algorithm 1, in which we define the main loop, and Algorithm 2, in which we define a recursion-based subroutine. Intuitively, Algorithm 2 takes any sequence of actions $a_{0:h-1}$ as input, and returns a Monte-Carlo estimator of $V^\star_h(s_h(a_{0:h-1}))$ with sufficiently small error. We keep two data sets $\mathcal{D}_h, \mathcal{D}_{a;h}$ in the algorithm. $\mathcal{D}_h$ simply stores features in the format of $\hat{x}_h(a_{0:h-1})$, and $\mathcal{D}_{a;h}$ stores pairs of feature and scalar $(\hat{x}_h(a_{0:h-1}), y)$ where as we will explain later y approximates $Q^\star_h(s(a_{0:h-1}), a)$. The dataset $\mathcal{D}_{a;h}$ will be used for linear regression.

The high-level idea behind our algorithm is that at a latent state reached by $a_{0:h-1}$, we use least squares to predict the optimal action when the data at hand is exploratory enough to cover $\hat{x}_h(a_{0:h-1})$ (intuitively, coverage means $\hat{x}_h(a_{0:h-1})$ lives in the span of the features in $\mathcal{D}_h$). Once we predict the optimal action a, we execute that action a and call Algorithm 2 to estimate the value $V^\star_{h+1}(s_h(a_{0:h-1}, a))$, which together with the reward $r_h(s_h(a_{0:h-1}), a)$, gives us an estimate of $V^\star_h(s_h(a_{0:h-1}))$. On the other hand, if the data $\mathcal{D}_h$ does not cover $\hat{x}_h(a_{0:h-1})$, which means that we cannot rely on least square predictions to confidently pick the optimal action at $s_h(a_{0:h-1})$, we simply try out all possible actions $a \in \mathcal{A}$, each followed by a call to Algorithm 2 to compute the value of Compute-$V^\star(a_{0:h-1}, a)$. Once we estimate $Q^\star_h(s_h(a_{0:h-1}), a), \forall a \in \mathcal{A}$, we can select the optimal action. To avoid making too many recursive calls, we notice that whenever our algorithm encounters the situation where the current data does not cover the test point $\hat{x}_h(a_{0:h-1})$ (i.e., a bad event), we add $\hat{x}_h(a_{0:h-1})$ to the dataset $\mathcal{D}_h$ to expand the coverage of $\mathcal{D}_h$.

We first explain Algorithm 1 assuming Algorithm 2 returns the optimal $V^\star_h(s_h(a_{0:h-1}))$ with small error when the input is $a_{0:h-1}$. In line 6, we recursively estimate $Q^\star_h(s_h(a_{0:h-1}), a)$ by running least squares regression. If at every level the data is exploratory in line 8, then we return the set of actions in line 9 and terminate the algorithm. We later prove that this returned sequence of actions is indeed the globally optimal sequence of actions. If the data is not exploratory at some level h, the estimation based on least squares regression would not be accurate enough, we query recursive calls for all actions and get an estimation of $Q^\star_h(s_h(a_{0:h-1}), a))$ for all $a \in \mathcal{A}$. Whenever line 11 is triggered, it means that we run into a state $s_h(a_{0:h-1})$ whose feature $\hat{x}_h(a_{0:h-1})$ is not covered by the training data $\mathcal{D}_h$. Hence, to keep track of the progress of learning, we add all new data collected at $s(a_{0:h-1})$ into the existing data set in line 14 and line 17.

Next, we explain Algorithm 2 whose goal is to return an estimate of $V^\star_h(s_h(a_{0:h-1}))$ with sufficiently small error for a given sequence $a_{0:h-1}$. This algorithm is recursively defined. In line 7, we judge whether the data is exploratory enough so that least square predictions can be accurate. If it is, we choose the optimal action using estimate of $Q^\star_h(s_h(a_{0:h-1}), a)$ for each a on the data set $\mathcal{D}_{a;h}$, i.e., $\langle \hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1}) \rangle$ by running least squares regression in line 8. While the data set has good coverage, the finite sample error still remains in this estimation step. Thanks to the gap in Assumption 4, even if there is certain estimation error in $\langle \hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1}) \rangle$, as long as that is smaller than half of the gap, the selected action a_h in line 8 is correct (i.e., a_h is the optimal action at latent state $s_h(a_{0:h-1})$). Then, after rolling out this a_h and calling the recursion at $h+1$ in line 10, we get a Monte-Carlo estimate of $V^\star_h(s_h(a_{0:h-1}))$ with sufficiently small error as proved by induction later.

We consider bad events where the data is not exploratory. In this case, for each action $a' \in \mathcal{A}$, we call the recursion in line 14. Since this call gives a Monte-Carlo estimate of $V^\star_{h+1}(s_{h+1}(a_{0:h-1}, a'))$, we can obtain a Monte-Carlo estimator of $Q^\star_h(s_h(a_{0:h-1}), a)$ by adding $\hat{r}_h(s_h(a_{0:h-1}), a)$. In bad events, we record the pair of $\{\hat{x}_h(a_{0:h-1}), y_{a;h}\}$ for each $a \in \mathcal{A}$ in Line 14 and record $\hat{x}_h(a_{0:h-1})$ in line 17. Whenever these bad events happen, by adding new data to the datasets, we have explored. In line 18, we return an estimate of $V^\star_h(s_h(a_{0:h-1}))$ with small error.

4.2. Analysis

The following theorems are our main results. We can ensure our algorithm is both statistically and computationally efficient. Our work is the first work with such a favorable guarantee on POMDPs.

Theorem 3 (Sample Complexity)**.** *Suppose Assumption 1, 2, 3 and 4 hold. Assume $\|\psi(o)\| \le 1$ for any $o \in \mathcal{O}$. Define $\Theta = W/\min_h \eta(G_h)$ where $\eta(G_h)$ is the smallest singular value of G_h. By properly setting λ, M, ε, with probability $1-\delta$, the algorithm outputs the optimal actions $a^\star_{0:H-1}$ after using at most the following number of samples*

$$\tilde{O}\left(H^5\Theta^5 d^2 A^2 (1/\Delta)^5 \ln(1/\delta)\right).$$

Here, we $\tilde{O}$ suppresses $\mathrm{polylog}(H, d, \ln(1/\delta), 1/\Delta, A, \Theta)$ *multiplicative factors.*

Note Theorem 3 is a PAC result, except there is no "approximately" (the "A" of "PAC") because we output the true optimal action sequence with probability $1-\delta$. I.e., we are simply *probably correct*.

Corollary 1 (Computational complexity)**.** *Assume basic arithmetic operations $+, -, \times, \div$, sampling one sample,*

Algorithm 1 Efficient Q-learning for Deterministic POMDPs (EQDP)

Input: parameters M, ε, λ
Initialize datasets $\mathcal{D}_{a;0}, \dots, \mathcal{D}_{a;H-1}$ for any $a \in \mathcal{A}$ and $\mathcal{D}_0, \dots, \mathcal{D}_{H-1}$. Given $\mathcal{D}_{a;h}$ and $\mathcal{D}_h$,

$$\hat{\theta}_{a;h}(\mathcal{D}_{a;h}) := \Sigma_h^{-1}(\mathcal{D}_h) \sum_{j=1}^{|\mathcal{D}_h|} \hat{x}_h(a_{0:h-1}^{(j)}) y_{a;h}^{(j)},$$
$$\Sigma_h(\mathcal{D}_h) := \sum_{j=1}^{|\mathcal{D}_h|} \hat{x}_h(a_{0:h-1}^{(j)}) \hat{x}_h^\top(a_{0:h-1}^{(j)}) + \lambda I$$

where $\mathcal{D}_{a;h} = \{\hat{x}_h(a_{0:h-1}^{(j)}), y_{a;h}^{(j)}\}$ and $\mathcal{D}_h = \{\hat{x}_h(a_{0:h-1}^{(j)})\}$.
while true **do**
  for $h = 0 \to H-1$ **do**
    Collect M i.i.d samples from $\mathbb{O}_h(\cdot \mid s_h(a_{0:h-1}))$ by executing $\{a_{0:h-1}\}$ and construct an estimated feature $\hat{x}_h(a_{0:h-1}) = (1/M) \sum \psi(o^{(i)})$
    Set $a_h = \operatorname{argmax}_a \langle \hat{\theta}_{a;h}(\mathcal{D}_{a;h}), \hat{x}_h(a_{0:h-1}) \rangle$
  end for
  if $\forall h : \|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} \le \varepsilon$ **then**
    Return $\{a_0, \dots, a_{H-1}\}$
  else
    Find the smallest h such that $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} > \varepsilon$
    for $\forall a' \in \mathcal{A}$ **do**
      Collect M i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean
      Compute $y_{a';h} = \hat{r}_h(s_h(a_{0:h-1}), a') +$ Compute-$V^\star(h+1; \{a_{0:h-1}, a'\})$
      Add $\mathcal{D}_{a';h} = \mathcal{D}_{a';h} + \{\hat{x}_h(a_{0:h-1}), y_{a';h}\}$
    end for
    Add $\mathcal{D}_h = \mathcal{D}_h + \{\hat{x}_h(a_{0:h-1})\}$
  end if
end while

Algorithm 2 Compute-$V^\star$

Input: time step h, state $a_{0:h-1}$
if $h = H-1$ **then**
  Collect M i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean for any $a' \in \mathcal{A}$
  Return $\max_a \hat{r}_h(a_{0:H-2}, a)$
else
  Collect M i.i.d samples from $\mathbb{O}_h(\cdot \mid s_h(a_{0:h-1}))$ by executing $\{a_{0:h-1}\}$ and construct an estimated feature $\hat{x}_h(a_{0:h-1}) = 1/M \sum \psi(o^{(i)})$
  if $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} \le \varepsilon$ **then**
    Set $a_h = \operatorname{argmax}_a \langle \hat{\theta}_{a;h}(\mathcal{D}_{a;h}), \hat{x}(a_{0:h-1}) \rangle$
    Collect M i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a_h)$ by executing $\{a_{0:h-1}, a_h\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a_h)$ by taking its mean
    Return $\hat{r}_h(s_h(a_{0:h-1}), a_h)$ + Compute-$V^\star(h+1; \{a_{0:h-1}, a_h\})$
  else
    for $a' \in \mathcal{A}$ **do**
      Collect M i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean
      $y_{a';h} = \hat{r}_h(s_h(a_{0:h-1}), a') +$ Compute-$V^\star(h+1; \{a_{0:h-1}, a'\})$
      $\mathcal{D}_{a';h} := \mathcal{D}_{a';h} + \{\hat{x}_h(a_{0:h-1}), y_{a';h}\}$
    end for
    Add $\mathcal{D}_h := \mathcal{D}_h + \{\hat{x}_h(a_{0:h-1})\}$
    Return $\max_a y_{a;h}$
  end if
end if

comparison of two values, take unit time. The computational complexity [1] *is* $\mathrm{poly}(H, d, \Theta, \ln(1/\delta), 1/\Delta, A)$.

We provide the sketch of the proof. For ease of understanding, suppose the reward is deterministic; thus, $\hat{r}_h = r_h$. The full proof is deferred to Section C. The proof consists of three steps:

1. Show *Compute-$V^\star$* always returns $V_h^\star(s_h(a_{0:h-1}))$ given input $a_{0:h-1}$ in high probability.
2. Show when the algorithm terminates, it returns the optimal policy.
3. Show the number of samples we use is upper-bounded by $\mathrm{poly}(H, d, \Theta, \ln(1/\delta), 1/\Delta, A)$.

[1] We ignore the bit complexity following the convention. We focus on arithmetic complexity.

Hereafter, we always condition on events $\|\hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1})\|$ is small enough every time when we generate $\hat{x}_h(a_{0:h-1})$. Before proceeding, we remark in MDPs with deterministic transitions, a similar strategy is employed (Du et al., 2020; Wen & Van Roy, 2013). Compared to them, we need to handle the unique challenge of uncertainty about $\hat{x}_h$. Recall we cannot use the true x_h.

First step. We use induction regarding $h \in [H]$. The correctness of the base case ($h = H-1$) is immediately verified. Thus, we prove this is true at h assuming *Compute-$V^\star$*$(h+1, a_{0:h})$ returns $V_{h+1}^\star(s_{h+1}(a_{0:h}))$ at $h+1$ for any possible inputs $a_{0:h}$ in the algorithm.

We need to consider two cases. The first case is a good event when $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} \le \varepsilon$. In this case, we first regress $y_{a;h}$ on $\hat{x}_h$ and obtain $\langle \hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1}) \rangle$. As we mentioned, the challenge is that the estimated feature $\hat{x}_h(a_{0:h-1})$ is not equal to the true feature $x_h(a_{0:h-1})$. Here,

we have

$$
\begin{aligned}
&|\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - Q^\star_h(s_h(a_{0:h-1}), a)| \\
&= |\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - \langle\theta^\star_{a;h}, x_h(a_{0:h-1})\rangle| \\
&\le |\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - \langle\theta^\star_{a;h}, \hat{x}_h(a_{0:h-1})\rangle| \\
&+ |\theta^\star_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - \langle\theta^\star_{a;h}, x_h(a_{0:h-1})\rangle| \\
&\le \underbrace{\mathrm{poly}(1/M, d, \Theta, H)\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)}}_{(a)} \\
&+ \underbrace{\Theta\|\hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1})\|}_{(b)}.
\end{aligned}
$$

From the second line to the third line, we use some non-trivial reformulation as explained in Section C. In (a), the term $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)}$ is upper-bounded by ε. By setting ε properly and taking large M, we can ensure the term (a) is less than $\Delta/4$. Similarly, by taking large M, we can ensure the term (b) is upper-bounded by $\Delta/4$. Therefore, we can show $|\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - Q^\star_h(s_h(a_{0:h-1}), a)| < \Delta/2$ for any $a \in \mathcal{A}$. Since $|\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle - Q^\star_h(s_h(a_{0:h-1}), a)| < \Delta/2$ for any $a \in \mathcal{A}$, together with the gap assumption, by setting $M = \mathrm{poly}(H, d, \Theta, \ln(1/\delta), 1/\Delta, A)$, we can ensure $\mathrm{argmax}_a\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle = \mathrm{argmax}_a Q^\star_h(s_h(a_{0:h-1}), a)$ in line 8 in Algorithm 2. Since this selected action a_h is optimal (after $a_{0:h-1}$), by inductive hypothesis, we ensure to return $V^\star_h(s_h(a_{0:h-1}))$ recalling *Compute-$V^\star$*$(h+1; a_{0:h}) = V^\star_{h+1}(s_{h+1}(a_{0:h}))$.

Next, we consider a bad event when $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} > \varepsilon$. In this case, we query the recursion for any $a' \in \mathcal{A}$. By inductive hypothesis, we can ensure $y_{a';h} = Q^\star_h(s_h(a_{0:h-1}), a')$. Hence, in Line 18 in Algorithm 2, $\max_a y_{a;h} = V^\star_h(s_h(a_{0:h-1}))$ is returned.

Second step. When the algorithm terminates, i.e., $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} < \varepsilon$ for all h, following the first-step, we can show $a_h = \mathrm{argmax}_a\langle\hat{\theta}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle$ always returns the optimal action $a^\star_h$.

Third step. The total number of bad events $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}(\mathcal{D}_h)} > \varepsilon$ (line 10 in Algorithm 1 and line 11 in Algorithm 2) for any h can be bounded in the order of $O(d/\varepsilon^2)$ via a standard elliptical potential argument. Once no such bad events happen, the termination criteria in the main algorithm ensures we will terminate. With some additional argument, we can also show the number of times we visit Line 10 and line 14 in Algorithm 2 is upper-bounded by $O(H^2Ad/\varepsilon^2)$. Thus the algorithm must terminate in polynomial number of calls of Compute-$V^\star$. Each procedure Compute-$V^\star$ collects $O(M)$ fresh samples in line 6. Thus the total sample complexity is bounded by $O(H^2MAd/\varepsilon^2)$.

Remark 1 (Comparison to Du et al. (2020))**.** *In deterministic MDPs, Du et al. (2020) uses a gap assumption to tackle agnostic learning, i.e., model misspecification. The reason we use the gap is different from theirs. We use the gap assumption to handle the noise from estimating features using future observations. We additionally deal with the unique challenge arising from uncertainty in features.*

Remark 2 (Practical performance of algorithm)**.** *We consider experiments using grid-world environments where we observe noisy observations of the latent state due to imperfect sensors in Section G. As mentioned in Section 1, this experiment is motivated by practical scenarios in autonomous driving. Similar experimental settings are considered in Du et al. (2019). We demonstrate our proposed method can return the optimal policy with low sample complexity.*

4.3. Examples

We instantiate our results with tabular POMDPs and Gaussian POMDPs.

Example 3 (continues=ex:undercomplete)**.** *Let $|\mathcal{S}| = S, |\mathcal{O}| = O$. In the tabular case, we suppose $S \le O$. Here, $d = O$. Recall we suppose the reward at any step lies in $[0, 1]$. Since $Q^\star_h(s, a)$ belongs to $\{\langle\theta_{a;h}, \phi(s)\rangle; \|\theta_a\| \le \sqrt{S}H\}$ where $\phi(\cdot)$ is a one-hot encoding vector over $\mathcal{S}$, we can set $\Theta = \sqrt{S}H/(\min_h \eta(\mathbb{O}_h))$. The sample complexity is $\tilde{O}(H^{10}S^{5/2}A^2O^2\ln(1/\delta)/\{\min_h \eta(\mathbb{O}_h)^5\Delta^5\})$.*

Jin et al. (2020a) obtain a similar result in the tabular setting without a gap condition to get an ε-near optimal policy. Together with the gap, their algorithm can also output the exact optimal policy with polynomial sample complexity like our guarantee. However, it is unclear whether their algorithm can be extended to HSE-POMDPs where state space or observation space is continuous.

Example 4 (continues=ex:gaussian)**.** *In Gaussian POMDPs, we assume $S \le d$. Recall we suppose the reward at any step lies in $[0, 1]$. Since $Q^\star_h(s, a)$ belongs to $\{\langle\theta_{a;h}, \phi(s)\rangle; \|\theta_{a;h}\| \le \sqrt{S}H\}$ where $\phi(\cdot)$ is a one-hot encoding vector over $\mathcal{S}$, we can set $\Theta = \sqrt{S}H/(\min_h \eta(O_h))$. The sample complexity is $\tilde{O}(H^{10}S^{5/2}A^2d^2\ln(1/\delta)/\{\min_h \eta(\mathbb{O}_h)^5\Delta^5\})$. Notably, this result does not depend on $|\mathcal{O}|$.*

4.4. Infinite-Dimensional Case

We briefly discuss the case when ϕ and ψ are infinite-dimensional. The detail is deferred to Section B. We introduce a kernel $k_{\mathcal{S}}(\cdot,\cdot) : \mathcal{S}\times\mathcal{S} \to \mathbb{R}$ and $k_{\mathcal{O}}(\cdot,\cdot) : \mathcal{O}\times\mathcal{O} \to \mathbb{R}$ and denote the corresponding feature vector $\psi : \mathcal{S} \to \mathbb{R}$ and $\phi : \mathcal{O} \to \mathbb{R}$, respectively. Then, when $Q^\star_h(\cdot, a)$ belongs to $\mathcal{H}_{\mathcal{S}}$ which is an RKHS corresponding to $k_{\mathcal{S}}$, if there exists a left invertible conditional embedding, we can ensure $Q^\star(\cdot, a)$ is linear in $\mathbb{E}_{o\sim\mathbb{O}_h(\cdot)}[\psi(o)]$. After this observation, we can use a similar algorithm as Algorithm 1 and Algorithm 2 by replacing linear regression with kernel ridge regression using $k_{\mathcal{S}}(\cdot,\cdot)$ and $k_{\mathcal{O}}(\cdot,\cdot)$. Finally, the sample

complexity is similarly obtained by replacing d with the maximum information gain over $\psi(\cdot)$ denoted by $\tilde{d}$. The rate of maximum information gain is known in many kernels such as Matérn kernel or Gaussian kernel (Valko et al., 2013; Srinivas et al., 2009; Chowdhury & Gopalan, 2017). In terms of computation, we can still ensure the polynomial complexity with respect to $\tilde{d}$ noting kernel ridge regression just requires $O(n^3)$ computation when we have n data at hand (n depends on $\tilde{d}$).

5. Learning with Multi-step Futures

We have so far considered a one-step future that has some signal of latent states. In this section, we show we can use multi-step futures that can be useful in settings such as overcomplete POMDPs. To build intuition, we first focus on the tabular case.

Tabular overcomplete POMDPs. Consider a distribution $\mathcal{S} \to \Delta(\mathcal{O}^K)$: $\mathbb{P}(o_{h:h+K-1} \mid s_h; a_{h:h+K-2})$ which means the conditional distribution of $o_{h:h+K-1}$ given s_h when we execute actions $a_{h:h+K-2}$. Let $\mathbb{P}_h^{[K]}(a_{h:h+K-2}) \in \mathbb{R}^{O^K \times S}$ be the corresponding matrix where each entry is $\mathbb{P}(o_{h:h+K-1} \mid s_h; a_{h:h+K-2})$. For undercomplete POMDPs, we have $\mathbb{P}_h^{[1]} = \mathbb{O}_h$ and $\mathbb{P}_h^{[1]}$ being full column rank. Note there is no dependence of actions when $K = 1$.

Assumption 5. *Given $K \in \mathbb{N}^+$, there exists an (unknown) sequence $a^\diamond_{h:h+K-2} \in \mathcal{A}^{K-1}$ such that $\mathbb{P}_h^{[K]}(a^\diamond_{h:h+K-2})$ is full-column rank, i.e., $\mathrm{rank}(\mathbb{P}_h^{[K]}(a^\diamond_{h:h+K-2})) = S$.*

This assumption says a multi-step future after executing some (unknown) action sequence with length $K-1$ has some signal of latent states. Executing such a sequence of actions can be considered as performing the procedure of information gathering (i.e., a robot hand with touch sensors can always execute the sequential actions of touching an object from multiple angles to localize the object before grasping it). This assumption is weaker than $\mathrm{rank}(\mathbb{P}_h^1) = S$ and extensively used in the literature on PSRs (Boots et al., 2011; Littman & Sutton, 2001; Singh et al., 2004). This assumption permits learning in the overcomplete case $S > O$. Under Assumption 5, we can show $Q^\star_h$ is still linear in some estimable feature.

Lemma 1. *For an overcomplete tabular POMDP, suppose Assumption 5 holds. Define a mapping $z_h^{[K]} : \mathcal{S} \to \mathbb{R}^{\mathcal{O}^K \times \mathcal{A}^{K-1}}$ as $z_h^{[K]}(s_h) = \{\mathbb{P}(o_{h:h+K-1} \mid s_h; a_{h:h+K-2})\}$. For $\forall a \in \mathcal{A}$, there exists $\theta^\star_{a;h}$ such that $Q^\star_h(s,a) = \langle \theta^\star_{a;h}, z_h^{[K]}(s) \rangle$.*

Non-tabular setting. We return to the non-tabular setting. We define a feature $\psi : \mathcal{O}^K \to \mathbb{R}^d$. We need the following assumption, which is a generalization of Assumption 5.

Assumption 6. *Given $K \in \mathbb{N}^+$, there exists an (unknown) sequence $a^\diamond_{h:h+K-2} \in \mathcal{A}^{K-1}$ and a left-invertible matrix G_h such that $\mathbb{E}[\psi(o_{h:h+K-1}) \mid s_h; a^\diamond_{h:h+K-2}] = G_h \phi(s_h)$.*

Then, we can ensure $Q^\star_h(s,a)$ is linear in some estimable feature. This is a generalization of Lemma 1.

Lemma 2. *Suppose Assumption 6. We define a feature $z_h^{[K]} : \mathcal{S} \to \mathbb{R}^{d\mathcal{A}^{K-1}}$ where $z_h^{[K]}(s_h)$ is defined as a $d\mathcal{A}^{K-1}$-dimensional vector stacking $\mathbb{E}[\psi(o_{h:h+K-1}) \mid s_h; a_{h:h+K-2}]$ for each $a_{h:h+K-2} \in \mathcal{A}^{K-1}$. Then, for each $a \in \mathcal{A}$, there exists $\theta^\star_{a;h} \in \mathbb{R}^{d\mathcal{A}^{K-1}}$ such that $Q^\star_h(s,a) = \langle \theta^\star_{a;h}, z_h^{[K]}(s) \rangle$*

The above lemma suggests that $Q^\star_h(s,a)$ is linear in $z_h^{[K]}(s)$ for each $a \in \mathcal{A}$. However, since we cannot exactly know $z_h^{[K]}(s)$, we need to estimate this new feature. Compared to the case with $K = 1$, we need to execute multiple $(K-1)$ actions. Given a sequence $a_{0:h-1}$, we want to estimate $x^{[K]}(a_{0:h-1}) := z_h^{[K]}(s_h(a_{0:h-1}))$ since our aim is to estimate $Q^\star_h(s_h(a_{0:h-1}), a)$ for any $a \in \mathcal{A}$ at time step h. The feature $x^{[K]}(a_{0:h-1}) \in \mathbb{R}^{d\mathcal{A}^{K-1}}$ is estimated by taking an empirical approximation of $\mathbb{E}[\psi(o_{h:h+K-1}) \mid s_h(a_{0:h-1}); a_{h:h+K-2}]$ by rolling out every possible actions of $a_{h:h+K-2}$ after $a_{0:h-1}$. We denote this estimate by $\hat{x}_h^{[K]}(a_{0:h-1})$. Therefore, we can run the same algorithm as Algorithm 1 and Algorithm 2 by just replacing $\hat{x}_h(a_{0:h-1})$ with $\hat{x}_h^{[K]}(a_{0:h-1})$. Compared to the case with $K = 1$, when $K > 1$, we need to pay an additional multiplicative A^{K-1} factor to try every possible action with length $K-1$. We have the following guarantee.

Theorem 4 (Sample complexity)**.** *Suppose Assumptions 2, 3,4, 6 hold. Assume $\|\psi(o)\| \le 1$ for any $o \in \mathcal{O}$. Let $\Theta = W/\min_h \eta(G_h)$. By properly setting λ, M and ε, with probability $1-\delta$, the algorithm outputs the optimal actions $a^\star_{0:H-1}$ after using the following number of samples*

$$\tilde{O}\left(H^5 \Theta^5 A^{3K-1} d^2 (1/\Delta)^5 \ln(1/\delta)\right).$$

Comparing to Theorem 3, we would incur additional $O(A^K)$. In the tabular case, noting $d = O^K$, we would additionally incur $O(O^K)$. Computationally, we also need to pay $O(A^K)$. Hence, there is some tradeoff between the weakness of the assumption and the sample/computational complexity.

6. Summary

We propose a computationally and statistically efficient algorithm on large-scale POMDPs where transitions are deterministic and emission have conditional mean embeddings.

Acknowledgement

WS acknowledges funding support from NSF IIS-2154711. NK acknowledge funding support from NSF IIS-1846210.

References

Agarwal, A., Jiang, N., Kakade, S. M., and Sun, W. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 2019.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. Reinforcement learning of pomdps using spectral methods. In *Conference on Learning Theory*, pp. 193–256. PMLR, 2016.

Besse, C. and Chaib-Draa, B. Quasi-deterministic partially observable markov decision processes. In *International Conference on Neural Information Processing*, pp. 237–246. Springer, 2009.

Bonet, B. Deterministic pomdps revisited. *arXiv preprint arXiv:1205.2659*, 2012.

Boots, B., Siddiqi, S. M., and Gordon, G. J. Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research*, 30(7):954–966, 2011.

Boots, B., Gordon, G., and Gretton, A. Hilbert space embeddings of predictive state representations. *arXiv preprint arXiv:1309.6819*, 2013.

Burago, D., De Rougemont, M., and Slissenko, A. On the complexity of partially observed markov decision processes. *Theoretical Computer Science*, 157(2):161–183, 1996.

Cai, Q., Yang, Z., and Wang, Z. Sample-efficient reinforcement learning for pomdps with linear function approximations. *arXiv preprint arXiv:2204.09787*, 2022.

Chen, F., Bai, Y., and Mei, S. Partially observable rl with b-stability: Unified structural condition and sharp sample-efficient algorithms. *arXiv preprint arXiv:2209.14990*, 2022.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pp. 844–853. PMLR, 2017.

Chowdhury, S. R. and Oliveira, R. No-regret reinforcement learning with value function approximation: a kernel embedding approach. *arXiv preprint arXiv:2011.07881*, 2020.

Dani, V., Hayes, T. P., and Kakade, S. M. Stochastic linear optimization under bandit feedback. 2008.

Dann, C., Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. On oracle-efficient pac rl with rich observations. *Advances in neural information processing systems*, 31, 2018.

Dikkala, N., Lewis, G., Mackey, L., and Syrgkanis, V. Minimax estimation of conditional moment models. *Advances in Neural Information Processing Systems*, 33:12248–12262, 2020.

Du, S., Kakade, S., Lee, J., Lovett, S., Mahajan, G., Sun, W., and Wang, R. Bilinear classes: A structural framework for provable generalization in rl. In *International Conference on Machine Learning*, pp. 2826–2836. PMLR, 2021.

Du, S. S., Luo, Y., Wang, R., and Zhang, H. Provably efficient q-learning with function approximation via distribution shift error checking oracle. *Advances in Neural Information Processing Systems*, 32, 2019.

Du, S. S., Lee, J. D., Mahajan, G., and Wang, R. Agnostic q-learning with function approximation in deterministic systems: Near-optimal bounds on approximation error and sample complexity. *Advances in Neural Information Processing Systems*, 33:22327–22337, 2020.

Even-Dar, E., Kakade, S. M., and Mansour, Y. Reinforcement learning in pomdps without resets. 2005.

Even-Dar, E., Kakade, S. M., and Mansour, Y. The value of observation for monitoring dynamic systems. In *IJCAI*, pp. 2474–2479, 2007.

Golowich, N., Moitra, A., and Rohatgi, D. Planning in observable pomdps in quasipolynomial time. *arXiv preprint arXiv:2201.04735*, 2022.

Guo, Z. D., Doroudi, S., and Brunskill, E. A pac rl algorithm for episodic pomdps. In *Artificial Intelligence and Statistics*, pp. 510–518. PMLR, 2016.

He, J., Zhou, D., and Gu, Q. Logarithmic regret for reinforcement learning with linear function approximation. In *International Conference on Machine Learning*, pp. 4171–4180. PMLR, 2021.

Hsu, D., Kakade, S. M., and Zhang, T. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Hu, Y., Kallus, N., and Uehara, M. Fast rates for the regret of offline reinforcement learning. *arXiv preprint arXiv:2102.00479*, 2021.

Javdani, S., Srinivasa, S. S., and Bagnell, J. A. Shared autonomy via hindsight optimization. *Robotics science and systems: online proceedings*, 2015, 2015.

Ji, S., Parr, R., and Carin, L. Nonmyopic multiaspect sensing with partially observable markov decision processes. *IEEE Transactions on Signal Processing*, 55(6):2720–2730, 2007.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2017.

Jin, C., Kakade, S., Krishnamurthy, A., and Liu, Q. Sample-efficient reinforcement learning of undercomplete pomdps. *Advances in Neural Information Processing Systems*, 33:18530–18539, 2020a.

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pp. 2137–2143. PMLR, 2020b.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Kearns, M., Mansour, Y., and Ng, A. Approximate planning in large pomdps via reusable trajectories. *Advances in Neural Information Processing Systems*, 12, 1999.

Krishnamurthy, A., Agarwal, A., and Langford, J. Pac reinforcement learning with rich observations. *Advances in Neural Information Processing Systems*, 29, 2016.

Kwon, J., Efroni, Y., Caramanis, C., and Mannor, S. Rl for latent mdps: Regret guarantees and a lower bound. *Advances in Neural Information Processing Systems*, 34, 2021.

Lale, S., Azizzadenesheli, K., Hassibi, B., and Anandkumar, A. Adaptive control and regret minimization in linear quadratic gaussian (lqg) setting. In *2021 American Control Conference (ACC)*, pp. 2517–2522. IEEE, 2021.

Li, G., Chen, Y., Chi, Y., Gu, Y., and Wei, Y. Sample-efficient reinforcement learning is feasible for linearly realizable mdps with limited revisiting. *Advances in Neural Information Processing Systems*, 34, 2021.

Littman, M. and Sutton, R. S. Predictive representations of state. *Advances in neural information processing systems*, 14, 2001.

Littman, M. L. Memoryless policies: Theoretical limitations and practical results. In *From Animals to Animats 3: Proceedings of the third international conference on simulation of adaptive behavior*, volume 3, pp. 238. Cambridge, MA, 1994.

Littman, M. L. *Algorithms for sequential decision-making*. Brown University, 1996.

Liu, B., Hsu, D., Ravikumar, P., and Risteski, A. Masked prediction tasks: a parameter identifiability view. *arXiv preprint arXiv:2202.09305*, 2022a.

Liu, Q., Chung, A., Szepesvári, C., and Jin, C. When is partially observable reinforcement learning not scary? *arXiv preprint arXiv:2204.08967*, 2022b.

Lykouris, T., Simchowitz, M., Slivkins, A., and Sun, W. Corruption-robust exploration in episodic reinforcement learning. In *Conference on Learning Theory*, pp. 3242–3245. PMLR, 2021.

Papadimitriou, C. H. and Tsitsiklis, J. N. The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

Pattipati, K. R. and Alexandridis, M. G. Application of heuristic search and information theory to sequential fault diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(4):872–887, 1990.

Platt, R., Kaelbling, L., Lozano-Perez, T., and Tedrake, R. Efficient planning in non-gaussian belief spaces and its application to robot grasping. In *Robotics Research*, pp. 253–269. Springer, 2017.

Platt Jr, R., Tedrake, R., Kaelbling, L., and Lozano-Perez, T. Belief space planning assuming maximum likelihood observations. 2010.

Simchowitz, M. and Jamieson, K. G. Non-asymptotic gap-dependent regret bounds for tabular mdps. *Advances in Neural Information Processing Systems*, 32, 2019.

Simchowitz, M., Singh, K., and Hazan, E. Improper learning for non-stochastic control. In *Conference on Learning Theory*, pp. 3320–3436. PMLR, 2020.

Singh, S., James, M. R., and Rudary, M. R. Predictive state representations: a new theory for modeling dynamical systems. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 512–519, 2004.

Song, L., Boots, B., Siddiqi, S., Gordon, G. J., and Smola, A. Hilbert space embeddings of hidden markov models. 2010.

Song, L., Fukumizu, K., and Gretton, A. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111, 2013.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Uehara, M., Sekhari, A., Lee, J. D., Kallus, N., and Sun, W. Provably efficient reinforcement learning in partially observable dynamical systems. *arXiv preprint arXiv:2206.12020*, 2022.

Valko, M., Korda, N., Munos, R., Flaounas, I., and Cristianini, N. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Wang, Y., Wang, R., and Kakade, S. An exponential lower bound for linearly realizable mdp with constant suboptimality gap. *Advances in Neural Information Processing Systems*, 34, 2021.

Weisz, G., Szepesvári, C., and György, A. Tensorplan and the few actions lower bound for planning in mdps under linear realizability of optimal value functions. *arXiv preprint arXiv:2110.02195*, 2021.

Wen, Z. and Van Roy, B. Efficient exploration and value function generalization in deterministic systems. *Advances in Neural Information Processing Systems*, 26, 2013.

Wu, J., Braverman, V., and Yang, L. Gap-dependent unsupervised exploration for reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 4109–4131. PMLR, 2022.

Xiong, Y., Chen, N., Gao, X., and Zhou, X. Sublinear regret for learning pomdps. *arXiv preprint arXiv:2107.03635*, 2021.

Zhan, W., Uehara, M., Sun, W., and Lee, J. D. Pac reinforcement learning for predictive state representations. *arXiv preprint arXiv:2207.05738*, 2022.

Zhong, H., Xiong, W., Zheng, S., Wang, L., Wang, Z., Yang, Z., and Zhang, T. A posterior sampling framework for interactive decision making. *arXiv preprint arXiv:2211.01962*, 2022.

A. Proof of Lower bounds

A.1. Proof of Theorem 1

The proof of Theorem 1 is based on the lower bounds in (Weisz et al., 2021), and consists of an underlying deterministic dynamics (on the state space) with stochastic rewards. We recall the following result from (Weisz et al., 2021):

Theorem 5 (Theorem 1.1 (Weisz et al., 2021) rephrased, Lower bound for the MDP setting)**.** *Suppose the learner has access to the features $\varphi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ such that $\max_s\{\|\psi(s)\| \leq 1\}$. Furthermore, let W, d, H be large enough constants There exists a class $\mathcal{M}$ of MDPs with deterministic transitions, stochastic rewards, action space $\mathcal{A}$ with $|\mathcal{A}| = d^{1/4} \wedge H^{1/2}$, and linearly realizable Q^* w.r.t. feature φ (i.e. $Q^*(s,a) = (w^*)^\top \phi(s,a)$ with $\|w^*\| \leq W$), such that any online planner that even has the ability to query a simulator at any state and action of its choice, must query at least $\Omega\big(2^{\Omega(d^{1/4} \wedge H^{1/2})}\big)$ many samples (in expectation) to find an $1/10$-optimal policy for some MDP in this class.*

Since learning is harder than planning, the lower bound also extends to the online learning setting.

An important thing to note about the above construction is that the suboptimality-gap is exponentially small in d, i.e. $\Delta = O(A^{-d})$. The above lower bound can be immediately extended to the POMDP setting. The key idea is to encode the stochastic rewards as "stochastic observations" while still preserving the linear structure. However, one needs to be careful of the fact that in our setting the features only depend on the states whereas in the above lower bound, the features depend on both state and actions. This can be easily fixed for finite action setting as shown in the following.

Proof. The proof follows by lifting the class of MDPs $\mathcal{M}$ in Theorem 5 to POMDPs. Consider any MDP $M = (\mathcal{S}, \mathcal{A}, \mathbb{T}, H, d', s_0, \varphi) \in \mathcal{M}$. Note that the construction of $\mathcal{M}$ guarantees that for $M \in \mathcal{M}$,

(a) There exists a w^* with $\|w^*\| \leq W$ such that for any s, $Q^*(s,a) = (w^*)^\top \varphi(s,a)$.

(b) There exists a stochastic reward function $\mathbb{Y}(s,a)$ for any (s,a).

(c) $|\mathcal{A}| = d'^{1/4} \wedge H^{1/2}$.

In the following, for each MDP $M \in \mathcal{M}$, we define a corresponding POMDP P_M. The underlying dynamics of the state space remains the same. The feature vector for any $s \in \mathcal{S}$ is defined as

$$\phi(s) = ((\varphi(s,a), r(s,a))_{a \in \mathcal{A}}).$$

where the dimensionality of ϕ is given by $d = (d'+1)|A| = (d'+1)(d'^{1/4} \wedge H^{1/2}) \leq 2(d'^{5/4} \wedge d'H^{1/2})$. Furthermore, we define $w_a^* = ((w^* \mathbb{1}\{a' = a\}, 0)_{a' \in \mathcal{A}}) \in \mathbb{R}^d$ and note that

$$\begin{aligned} Q^*(s,a) = w^\top \varphi(s,a) &= \langle ((w \mathbb{1}\{a' = a\}, 0)_{a' \in \mathcal{A}}), ((\varphi(s,a), r(s,a))_{a \in \mathcal{A}}) \rangle \\ &= w_a^\top \phi(s), \end{aligned} \tag{1}$$

and thus the above feature maps satisfies the linear Q^* property w.r.t. the features ϕ. By Theorem 5, $\|w_a\| \leq W$ (Assumption 3 satisfied).

We next define the emission distribution $\mathbb{O}$ and the feature maps $\psi : \mathcal{O} \to \mathbb{R}^d$. At any state s, we have stochastic observations o of the form

$$\psi(o) = ((\varphi(s,a), \mathbb{Y}(s,a))_{a \in \mathcal{A}}),$$

Since the rewards are stochastic, the observations above are also stochastic and clearly the emission distribution $\mathbb{O}$ is partitioned into $|\mathcal{S}|$ many components since each $o \in \mathcal{O}$ is associated with only one state $s \in \mathcal{S}$. Furthermore, the above definition satisfies the relation s

$$\mathbb{E}_{o \sim \mathbb{O}(\cdot|s)}[\psi(o)] = \phi(s). \tag{2}$$

Clearly, the above shows that Assumption 1 holds. Finally, Assumption 2 is satisfied by the construction in Theorem 5. Thus, the POMDP P_M constructed above satisfies Assumption 1, 2 and 3. We can similarly lift every MDP $M \in \mathcal{M}$ to

construct the POMDP class $\mathcal{P} = (P_M)_{M \in \mathcal{M}}$. Clearly, the observations in the POMDP (and the corresponding feature vectors) do not reveal any new information to the learner that can not be accessed by making $\mathcal{A}$ many calls in the underlying MDP at the same state (which due to deterministic state space dynamics can be simulated by taking all the other actions same till the last step, and then trying all other actions at the last step). Thus, from the query complexity lower bound in Theorem 5, we immediately get that there must exist some POMDP in the class $\mathcal{P}$ for which we need to collect

$$\Omega\Big(\frac{1}{|\mathcal{A}|} 2^{\Omega(d'^{1/4} \wedge H^{1/2})}\Big)$$

many samples (in expectation) in order to find an $1/10$-optimal policy, where the $\Omega(\cdot)$ notation hides polynomial dependence on W, d and H. Plugging in the relation $d = d'^{5/4} \wedge H^{1/2}$ in the above, we get the lower bound

$$\Omega\Big(\frac{1}{d^{1/5} \wedge H^{1/2}} 2^{\Omega\left(d^{1/5} \wedge H^{1/2}\right)}\Big).$$

□

A.2. Proof of Theorem 2

The proof of Theorem 2 is based on the lower bounds in (Wang et al., 2021), and consists of an underlying MDP with stochastic transitions (on the state space) and deterministic rewards. We recall the following result from (Wang et al., 2021).

Theorem 6 (Theorem 1 (Wang et al., 2021) rephrased, Lower bound for the MDP setting)**.** *Fix any $\Delta > 0$, and consider any online RL algorithm ALG that takes the state feature mapping $\varphi : \mathcal{S} \to \mathbb{R}^d$ and action feature mapping $\chi : \mathcal{A} \to \mathbb{R}^d$ as input. There exists a pair of state and action feature mappings (φ, χ) with $\max_{s,a} \{\|\varphi(s), \chi(a)\|\} \leq 1$, and an MDP $(\mathcal{S}, \mathcal{A}, \mathbb{T}, H, r)$ such that:*

(a) (Linear Q^* property) There exists an $M \in \mathbb{R}^{d \times d}$ such that $Q^*(s,a) = \varphi(s)^T M \chi(a)$ for any (s,a). Furthermore, $\|M\| \leq B$ for some universal constant B.

(b) (Suboptimality gap) There exists a $\Delta > 0$ such that $\min_{h \in [H], s, a} \{gap_h(s,a) \mid gap_h(s,a) > 0\} = \Delta$ where $gap_h(s,a)$ is defined as $V_h^*(s) - Q_h^*(s,a)$.

(c) The state space dynamics $\mathbb{T}$ is not deterministic.

Furthermore, ALG requires at least $\Omega\left(2^{\Omega(\min\{d,H\})}\right)$ samples to find an $1/20$-suboptimal policy for this MDP with probability at least $1/10$.

Proof. The proof is almost identical to the proof of Theorem 1 in (Wang et al., 2021). However, there is a subtle difference in the feature mapping considered. (Wang et al., 2021) consider feature mappings that take both s and a as inputs, however for our result we need separate state features and actions features. A closer analysis of (Wang et al., 2021) reveals that one can in-fact replicate their lower bounds with separate state features and action features. In particular, note that the result in (Wang et al., 2021) follows by associating a vector $v_s \in \mathbb{R}^{d'}$ with each state s and a vector $u_a \in \mathbb{R}^{d'}$ with each action a such that:

$$Q^*(s,a) = (\langle v_s, u_a \rangle + 2\alpha)\langle v_s, v_{a^*} \rangle,$$

where α is a universal constant and a^* is a fixed special action. Clearly, we can define the feature $\varphi(s) = \text{vec}([1, v_s] \otimes [1, v_s]) \in \mathbb{R}^d$, the feature $\chi(a) = \text{vec}([1, v_a] \otimes [1, v_a]) \in \mathbb{R}^d$ and the matrix $M \in \mathbb{R}^{d \times d}$ with $d = 2d' + 2$ such that

$$Q^*(s,a) = \varphi(s)^\top M \chi(a).$$

The minimum suboptimality-gap assumption and the lower bound now follow immediately from their result. We refer the reader to (Wang et al., 2021) for complete details of the construction. □

Note that the above construction has stochastic state space dynamics. The above lower bound can be immediately extended to our POMDP settings as shown below.

Proof. The POMDP that we construct is essentially the MDP given in Theorem 6. We define the features $\phi(s) = \varphi(s)$ (where φ are the features defined in Theorem 6)

We set the observations to exactly contain the underlying state, i.e. $\mathcal{O} = \mathcal{S}$ and $\mathbb{O}[o, s] = \mathbb{1}(s = o)$. Further, for any o, we define the features $\psi(o) = \phi(s)$ where s is the corresponding state for o. Clearly, Assumption 1 is satisfied.

We next note that $Q^*(s, a) = \phi(s)^\top M\chi(a) = w_a^\top \phi(s)$, where $w_a = M\chi(a)$ and satisfies $\|w_a\| \le \|M\|\|\chi(a)\| \le B$. Thus, Assumption 3 is satisfied. Finally, Assumption 4 is satisfied by the statement of Theorem 6. Finally, note that learning in this POMDP is exactly equivalent to learning in the corresponding MDP and thus the lower bound extends naturally.

□

B. Learning in Infinite Dimensional HSE-POMDPs

We consider the extension to infinite-dimensional RKHS. We introduce several definitions, provide an algorithm and show the guarantee. To simplify the notation, we assume $\mathbb{O}_h = \mathbb{O}$ for any $h \in [H]$.

Let $k_{\mathcal{S}}(\cdot, \cdot)$ be a (positive-definite) kernel over a state space. We denote the corresponding RKHS and feature vector as $\mathcal{H}_{\mathcal{S}}$ and $\phi(\cdot)$, respectively. We list several key properties in RKHS (**?**)Chapter 12]wainwright2019high. First, for any $f \in \mathcal{F}$, there exists $\{a_i\}$ such that $f = \sum_i a_i\phi_i$ and the following holds $\mathbb{E}_{s\sim u_S(s)}[\phi_i(s)\phi_i(s)] = \mathrm{I}(i = j)\mu_i$ where $u_S(s)$ is some distribution over $\mathcal{S}$. Besides, we have $k(\cdot, \cdot) = \sum_i \phi_i(\cdot)\phi_i(\cdot)$ and the inner product of f, g in $\mathcal{H}_{\mathcal{S}}$ satisfies $\langle f, g\rangle_{\mathcal{H}_{\mathcal{S}}} = \langle \sum_i a_i\phi_i, \sum_i b_i\phi_i\rangle_{\mathcal{H}_{\mathcal{S}}} = \sum_i a_ib_i$. Similarly, let $k_{\mathcal{O}}(\cdot, \cdot)$ be a (positive-definite) kernel over the observation space with feature $\psi(o)$ such that $\mathbb{E}_{o\sim u_{\mathcal{O}}(o)}[\psi_i(o)\psi_j(o)] = \mathrm{I}(i = j)\nu_i$ where $u_{\mathcal{O}}(\cdot)$ is some distribution over $\mathcal{O}$.

Then, the new kernel $\mathbb{E}_{o\sim z(s),o'\sim z(s')}[k_{\mathcal{O}}(o, o')]$ over $\mathcal{S}\times\mathcal{S}$ is induced. We denote this kernel by $\bar{k}(\cdot, \cdot)$ and the corresponding RKHS by $\mathcal{H}_{\bar{\mathcal{S}}}$.

Now, we introduce the following assumption which corresponds to Assumption 1.

Assumption 7 (Existence of linear mean embedding and its well-posedness)**.** *Suppose $\mathcal{H}_{\bar{\mathcal{S}}} = \mathcal{H}_{\mathcal{S}}$ and*

$$\sup_{\|p\|\le 1} \frac{p^\top \mathbb{E}_{u\sim u_S(s)}[\phi(s)\phi(s)^\top]p}{p^\top \mathbb{E}_{u\sim u_S(s)}[\mathbb{E}_{o\sim\mathbb{O}}[\psi(o) \mid s]\mathbb{E}_{o\sim\mathbb{O}}[\psi(o) \mid s]^\top]p} < \iota^2. \tag{3}$$

The first assumption states that for any $a^\top\phi(s)$ in $\mathcal{H}_{\mathcal{S}}$, there exists $b^\top\mathbb{E}_{s\sim\mathbb{O}}[\psi(o)]$ and the vice versa holds. This is a common assumption to ensure the existence of linear mean embedding operators (Song et al., 2010; Chowdhury & Oliveira, 2020). Equation 3 is a technical condition to impose constraints on the norms. For example, when ψ and ϕ are finite-dimensional, we can obtain this condition by setting $\iota = 1/(\min_h \eta(K_h))$. We remark a similar assumption is often imposed in the literature on instrumental variables (Dikkala et al., 2020). Under the above assumption, we can obtain the following lemma.

Lemma 3. *Given $a^\top\phi(s) \in \mathcal{H}_{\mathcal{S}}$ s.t. $\|a\| \le 1$, there exists b s.t. $a^\top\phi(s) = b^\top\mathbb{E}_{o\sim O(s)}[\psi(o)]$ and $\|b\| \le c$.*

When linear $Q^\star$-assumption holds as $Q_h^\star(\cdot, a) \in \mathcal{H}_{\mathcal{S}}(\forall a \in \mathcal{A})$, since $Q_h^\star(\cdot, a) \in \mathcal{H}_{\bar{\mathcal{S}}}(\forall a \in \mathcal{A})$ from the assumption, we can run a kernel regression corresponding to $\bar{k}(\cdot, \cdot)$ to estimate $Q_h^\star(\cdot, a)$. The challenge here is we cannot directly use $\bar{k}(\cdot, \cdot)$ in $\mathcal{H}_{\bar{\mathcal{S}}}$. We can only obtain an estimate of $\bar{k}(\cdot, \cdot)$. More concretely, given $a_{0:h-1}$, an estimate of $\bar{k}(s_h(a_{0:h-1}), s_h(a_{0:h-1}))$ is given by

$$\hat{k}\left(Z(a_{0:h-1}), Z(a'_{0:h-1})\right) = 1/M^2 \sum_{z_i\in Z(a_{0:h-1}), z'_j\in Z(a'_{0:h-1})} k_{\mathcal{O}}(z_i, z'_j)$$

where $Z(a_{0:h-1})$ is a set of i.i.d M samples following $\mathbb{O}(\cdot \mid s_h(a_{0:h-1}))$ and $Z(a'_{0:h-1})$ is a set of i.i.d M samples following $\mathbb{O}(\cdot \mid s_h(a'_{0:h-1}))$.

B.1. Algorithm

With slight modification, we can use the same algorithm as Algorithm 3 and Algorithm 4. The only modification is changing the forms of $\mu_{a;h}$ and σ_h^2 using (nonparametric) kernel regression. Here, we define

$$\mu_{a;h}(Z(a_{0:h-1}), \mathcal{D}_{a;h}) = \hat{\mathbf{k}}(Z(a_{0:h-1}), \mathcal{D}_{a;h})^\top(\hat{\mathbf{K}}(\mathcal{D}_h) + \lambda I)^{-1}\mathbf{Y}(\mathcal{D}_{a;h}),$$
$$\sigma_h^2(\{a_{0:h-1}\}, \mathcal{D}_h) = \hat{k}(Z(a_{0:h-1}), Z(a_{0:h-1})) - \|\hat{\mathbf{k}}(Z(a_{0:h-1}), \mathcal{D}_h)\|^2_{(\hat{\mathbf{K}}(\mathcal{D}_h)+\lambda I)^{-1}}.$$

Algorithm 3 Deterministic POMDP

Initialize datasets $\mathcal{D}_{a;0}, \dots, \mathcal{D}_{a;H-1}$ for any $a \in \mathcal{A}$ and $\mathcal{D}_0, \dots, \mathcal{D}_{H-1}$
while true **do**
  for $h = 0 \to H-1$ **do**
    Collect M i.i.d samples $Z(\{a_{0:h-1}\}) \sim \mathbb{O}_h(\cdot \mid s_h(a_{0:h-1}))$ by executing $\{a_{0:h-1}\}$
    Set $a_h = \operatorname{argmax}_a \mu_{a;h}(Z(a_{0:h-1}), \mathcal{D}_{a;h})$
  end for
  if $\forall h : \sigma_h(Z(a_{0:h-1}), \mathcal{D}_h) \le \varepsilon$ **then**
    Return $\{a_0, \dots, a_{H-1}\}$
  else
    Find the smallest h such that $\sigma_h(Z(a_{0:h-1}), \mathcal{D}_h) > \varepsilon$,
    for $\forall a' \in \mathcal{A}$ **do**
      Collect M' i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean
      Compute $y_{a';h} = \hat{r}_h(s_h(a_{0:h-1}), a') +$ Compute-$V^\star(h+1; \{a_{0:h-1}, a'\})$
      Add $\mathcal{D}_{a';h} = \mathcal{D}_{a';h} + \{Z(a_{0:h-1}), y_{a';h}\}$
    end for
    Add $\mathcal{D}_h = \mathcal{D}_h + \{Z(a_{0:h-1})\}$
  end if
end while

where

$$\hat{\mathbf{k}}(Z(x), \mathcal{D}_h) = \{\hat{k}(Z(x), Z(x^i))\}_{i=1}^{|\mathcal{D}_h|}, \quad \hat{\mathbf{K}}(\mathcal{D}_h) = \{\hat{k}\left(Z(x^i), Z(x^j)\right)\}_{i=1,j=1}^{|\mathcal{D}_h|,|\mathcal{D}_h|}, \quad \mathbf{Y}(\mathcal{D}_{a;h}) = \{y_a^i\}_{i=1}^{|\mathcal{D}_{a;h}|}.$$

Note when features are finite-dimensional, they are reduced to Algorithm 1 and Algorithm 2.

B.2. Analysis

Let $\gamma(N; k_{\mathcal{O}})$ be a maximum information gain corresponding to a kernel $k_{\mathcal{O}}(\cdot, \cdot)$ defined by $\max_{C \subset \mathcal{O}:|C|=N} \ln(\det(I + \mathbf{K}_C))$ where $\mathbf{K}_C$ is a kernel matrix whose (i,j)-th entry is $k_{\mathcal{O}}(x_i, x_j)$ when $C = \{x_i\}$. This corresponds to d in the finite-dimensional setting. Maximum information gain can be computed in many kernel such as Gaussian kernels or Matérn kernels (Srinivas et al., 2009; Valko et al., 2013).

Theorem 7. *Suppose for any $a \in \mathcal{A}, h \in [H]$, $Q_h^\star(\cdot, a) \in \mathcal{H}_{\mathcal{S}}$ such that $\|Q_h^\star(\cdot, a)\|_{\mathcal{H}_{\mathcal{S}}} \le W$, Assumption 2, 3, 4 and 7. Then, when $\gamma(N; k_{\mathcal{O}}) = \Gamma N^\alpha (0 < \alpha < 1)$ and $k_{\mathcal{O}}(\cdot, \cdot) \le 1$, with probability $1 - \delta$, the algorithm outputs the optimal sequence of actions $a_{0:H-1}^\star$ using at most the following number of samples:*

$$\operatorname{poly}(W, \iota, \log(1/\delta), H, \Gamma, 1/\Delta, A).$$

The computational complexity is $\operatorname{poly}(W, \iota, \log(1/\delta), H, \Gamma, 1/\Delta, A)$ as well.

C. Proof of Section 4

The proof consists of three steps. We flip the order of the first and third step comparing to the main body to formalize the proof. To make the proof clear, we write the number of samples we use to construct $\hat{r}_h$ by M'. In the end, we set $M = M'$. Besides, we set $\lambda = 1$. In the proof, $c_1, c_2, \cdots$ are universal constants.

C.1. First Step

We start with the following lemma to show the algorithm terminates and the sample complexity is $\operatorname{poly}(M, M', H, d, 1/\varepsilon)$. We will later set appropriate M, M', ε.

Lemma 4 (Sample complexity)**.** *Algorithm 1 terminates after using $O((M + M')H^3 Ad/\varepsilon^2 \ln(1/\varepsilon))$ samples.*

Proof. The proof consists of two steps.

Algorithm 4 Compute-$V^\star$

Input: time step h, state $a_{0:h-1}$
if $h = H-1$ **then**
  Collect M' i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean for any $a' \in \mathcal{A}$
  Return $\max_a \hat{r}_h(a_{0:H-2}, a)$
else
  Collect M i.i.d samples $Z(\{a_{0:h-1}\}) \sim \mathbb{O}_h(\cdot \mid s_h(a_{0:h-1}))$ by executing $\{a_{0:h-1}\}$
  if $\sigma_h(Z(a_{0:h-1}), \mathcal{D}_h) \leq \varepsilon$ **then**
    Set $a_h = \mathrm{argmax}_a \, \mu_{a;h}(Z(a_{0:h-1}), \mathcal{D}_{a;h})$
    Collect M' i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a_h)$ by executing $\{a_{0:h}\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a_h)$ by taking its mean
    Return $\hat{r}_h(s_h(a_{0:h-1}), a_h) +$ Compute-$V^\star(h+1; \{a_{0:h-1}, a_h\})$
  else
    for $a' \in \mathcal{A}$ **do**
      Collect M' i.i.d samples from $\mathbb{Y}_h(\cdot \mid s_h(a_{0:h-1}), a')$ by executing $\{a_{0:h-1}, a'\}$ and compute $\hat{r}_h(s_h(a_{0:h-1}), a')$ by taking its mean
      $y_{a';h} = \hat{r}_h(s_h(a_{0:h-1}), a') +$ Compute-$V^\star(h+1; \{a_{0:h-1}, a'\})$
      $\mathcal{D}_{a';h} := \mathcal{D}_{a';h} + \{Z(a_{0:h-1}), y_{a';h}\}$
    end for
    Add $\mathcal{D}_h = \mathcal{D}_h + \{Z(a_{0:h-1})\}$
    Return $\max_a y_{a;h}$
  end if
end if

The number of times we call Line 11 in Algorithm 1 ($I_{\max}$) is upper-bounded by $O(Hd/\varepsilon^2 \ln(dHA/\varepsilon))$ At horizon h, when the new data $\hat{x}_h(a_{0:h-1})$ is added, we always have $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} > \varepsilon$ (Line 11 in Algorithm 1 or Line 11 in Algorithm 2). Let the total number of times we call Line 11 in Algorithm 1 and Line 11 in Algorithm 2 be N'. Then, we have

$$\sum_{i=1}^{N'} \|\hat{x}_h^{(i)}\|_{\Sigma_h^{-1}} \leq \sqrt{dN' \ln(1 + N'/d)}. \tag{4}$$

Thus, the following holds

$$\varepsilon N' \leq c_4 \sqrt{dN' \ln(1 + N'/d)}.$$

This implies N' is upper-bounded by

$$O(d/\varepsilon^2 \ln(1/\varepsilon)).$$

Thus, the number of we call Line 11 in Algorithm 1 is upper-bounded by $O(d/\varepsilon^2 \ln(1/\varepsilon))$ for any layer h. Considering the whole layer, $I_{\max}$ is upper-bounded by $O(Hd/\varepsilon^2 \ln(1/\varepsilon))$.

Calculation of total sample complexity When we call Line 11 in Algorithm 1, we consider the running time from Line 11 to 11. Let $m-1$ be the number of times we already visit Line 11 in Algorithm 1. Recall the maximum of m is at most $O(Hd/\varepsilon^2 \ln(1/\varepsilon))$.

Hereafter, we consider the case at iteration m. When we visit Line 14 in Algorithm 1, we need to start the recursion step in Algorithm 2. This recursion is repeated in a DFS manner from h to $H-1$ as in Figure **??**. When the algorithm moves from some layer to another layer, the algorithm calls Line 10 or Line 11, i.e., Line 14 $|A|$ times in Algorithm 2. Let the number of total times the algorithm calls Line 10 in Algorithm 2 (g in Figure **??**) be α_m. Let the number of times the algorithm visits Line 14 in Algorithm 2 and Line 14 in Algorithm 1 (b_a in Figure **??**) be β_m, respectively.

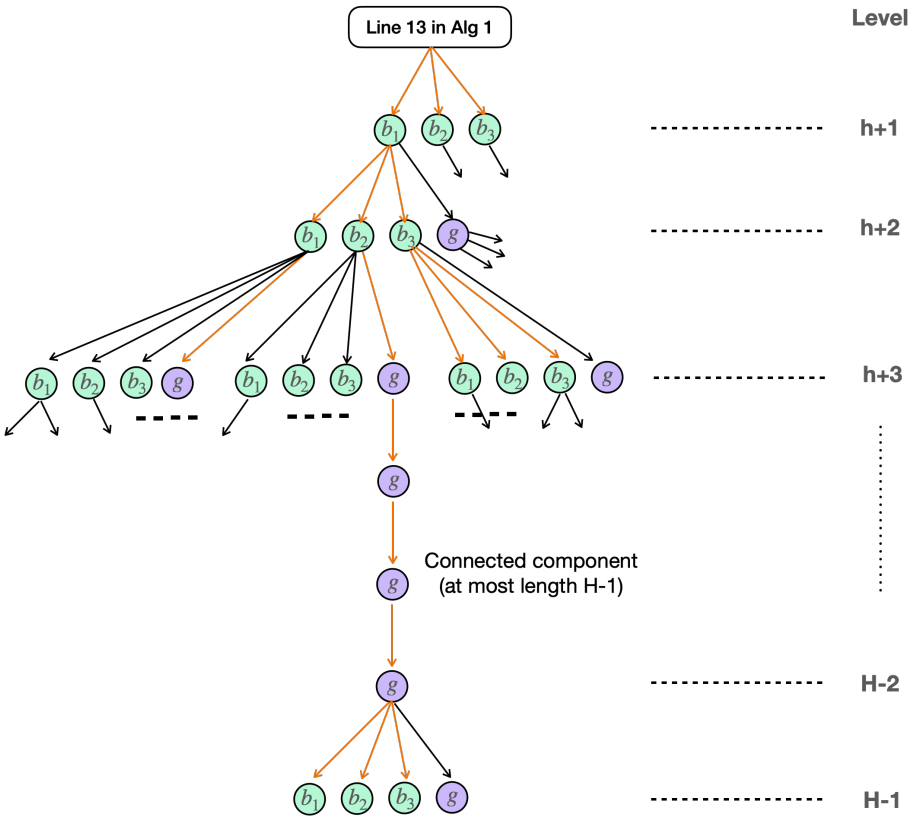


Figure 1. The root node corresponds to Line 14 in Algorithm 1. We denote Line 10 in Algorithm 2 by g. We denote line 14 in Algorithm 2 and Line 14 in Algorithm 1 corresponding to $a \in \mathcal{A}$ by b_a. In the illustration, we set $A = 3$. The example of paths the algorithm traverse is marked in orange. This corresponds to a graph $\tilde{\Omega}_m$. The number α_m is the total number of times the algorithm visits g at iteration m. The number β_m is the total number of times the algorithm visits $\{b_a\}$ at iteration m.

Then, the total sample complexity is upper-bounded by

$$\underbrace{HMI_{\max}}_{(a)} + \underbrace{\sum_{m=1}^{I_{\max}} (M + M'(A+1))H\alpha_m + (M/A + M'(A+1))H\beta_m\}}_{(b)}.$$

The term (a) comes from samples we use line 4 to line 7 in Algorithm 1. Note H is the number of samples we need to reset, M is the number of samples in $\hat{x}_h(a_{0:h-1})$ and N' upper-bounds the number of iterations in the main loop. Next, we see the term (b). Here, M' is the number of samples in $\hat{r}_h$. More specifically, we need MH samples in line 6 in Algorithm 2, which we traverse in both good and bad events (per bad event, we just use MH/A samples). Additionally, we need $M'H$ samples in line 10 in Algorithm 2 in good events and $M'H$ samples in line 14 in Algorithm 2 in bad events. When we call $H-1$, we use additionally use $AM'H$ samples. For each visit, we use at most $(A+1)M'H$ samples in line 3.

Next, we show $\alpha_m \leq H\beta_m$. First, we denote sets of all g and b_a nodes the algorithm traverse at iteration m in the tree by G_m and B_m, respectively. We denote a subgraph on the tree consisting of nodes and edges which the algorithm traverses by $\tilde{\Omega}_m$. We denote a subgraph in $\tilde{\Omega}_m$ consisting of nodes G_m and edges whose both sides belong to G_m by $\tilde{G}_m$. We divide $\tilde{G}_m$ into connected components on $\tilde{\Omega}_m$. Here, each component has at most H nodes. The most upstream node in a component is adjacent to some node in B_m on $\tilde{\Omega}_m$. Besides, this node in B_m is not shared by other connected components in $\tilde{G}_m$. This ensures that $\alpha_m \leq H\beta_m$.

Finally, we use $\sum_m \beta_m \leq O(HAd/\varepsilon^2 \ln(1/\varepsilon))$ as we see the number of times we call Line 11 in Algorithm 1 and Line 11

in Algorithm 2 at $h \in [H]$ is upper-bounded by $O(d/\varepsilon^2 \ln(1/\varepsilon))$ and we multiply it by HA. Thus,

$$HMI_{\max} + \sum_{m=1}^{I_{\max}} (M + M'(A+1))H\alpha_m + (M/A + M'(A+1))H\beta_m\}$$
$$= O((M+M')H^3A^2d/\varepsilon^2 \ln(1/\varepsilon)).$$

□

In this lemma, as a corollary, the following statement holds:

- The number of times we visit line 6 in Algorithm 2 is upper-bounded by $\sum_m (\beta_m/A + \alpha_m)$.
- The number of times we visit line 3, line 9 and line 13 in Algorithm 2 is upper-bounded by $\sum_m (\beta_m + \alpha_m)(A+1)$.

Here, we have

$$\sum_{m=1}^{I_{\max}} \beta_m = O(HAd/\varepsilon^2 \ln(1/\varepsilon)), \quad \sum_{m=1}^{I_{\max}} \alpha_m = O(H^2Ad/\varepsilon^2 \ln(1/\varepsilon)).$$

C.2. Second Step

We prove some lemma which implies that Algorithm 2 always returns a good estimate of $V_h^\star(s_h(a_{0:h-1}))$ in the algorithm in high probability. Before providing the statement, we explain several events we need to condition on.

C.2.1. Preparation

We first note for in the data $\mathcal{D}_{a;h}$, a value $y_{a:h}$ corresponding to $\hat{x}_h(a_{0:h-1})$ is always in the form of

$$y_{a:h} = \mathbb{E}[\sum_{k=h}^{H} r_k \mid s_h(a_{1:h-1}); a_h = a, a_{h+1:H-1} = a'_{h+1:H-1}] + \sum_{k=h}^{H} \nu_k, \quad \nu_k = 1/M' \sum_{i=1}^{M'} \tau_k^{[i]}$$

for some (random) action sequence $a'_{h:H-1}$. Note this is not a high probability statement. Here, $\tau_{a;h}$ is an i.i.d noise in rewards which come from line 3 in Algorithm 2 (when $h = H-1$), line 9 in Algorithm 2 (good events in when $h < H-1$), and in line 13 in Algorithm 2 (bad events in when $h < H-1$). We denote the whole noise part in $y_{a:h}$ by

$$z_{a;h} = \sum_{k=h}^{H} \nu_k.$$

C.2.2. Events we need to condition on

In the lemma, we need to condition on two types of events.

First event Firstly, we condition on the event

$$\forall a \in \mathcal{A}; |\langle \theta^\star_{a;h}, x_h(a_{0:h-1}) - \hat{x}_h(a_{0:h-1})\rangle| \le \min\left(\frac{\Delta}{6}, \frac{\Delta}{12\sqrt{N'\varepsilon}}\right) \tag{5}$$

every time we visit line 6 in Algorithm 2 and line 5 in Algorithm 1. The concentration is obtained noting $\theta_{a;h}^\top\{\hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1})\}$ is a Θ^2/M sub-Gaussian random variable with mean zero conditional on $x_h(a_{0:h-1})$. Formally, by properly setting M, we use the following lemma (simple application of Hoeffeding's inequality).

Lemma 5 (Concentration of feature estimators)**.** *With probability* $1-\delta'$,

$$\forall a; |\langle \theta^\star_{a;h}, x_h(a_{0:h-1}) - \hat{x}_h(a_{0:h-1})\rangle| \le \Theta\sqrt{\ln(A/\delta')/M}.$$

We later choose M so that (5) holds. Note the number of visitation is

$$I_{\max} H + \sum_m (\alpha_m + \beta_m / A) = O(H^2 A d / \varepsilon^2 \ln(1/\varepsilon)).$$

We take the union bound later.

Second event We condition on the event

$$|\nu_k| \le \min\left\{ \Delta/(6H), \frac{\Delta}{12\sqrt{N'\varepsilon}} \right\} \tag{6}$$

every time we visit line 3 in Algorithm 2 (when $h = H-1$), line 13 in Algorithm 2 (good events in $h \le H-1$), and line 17 in Algorithm 2 (bad events in $h \le H-1$). By properly setting M', the concentration is obtained noting ν_k is a $1/M^2$-sub-Gaussian variable as follows. This is derived as a simple application of Hoeffeding's inequality.

Lemma 6 (Concentration of reward estimators)**.** *With probability* $1-\delta'$*,*

$$|\nu_k| \le 2\sqrt{\ln(1/\delta')/M'}.$$

Later, we choose M' so that (6) is satisfied. Note the number of times we visit is

$$\sum_{m=1}^{I_{\max}} (\alpha_m + \beta_m)(A+1) = O((M+M')H^2 A^2 d/\epsilon^2 \ln(1/\epsilon)).$$

We take the union bound later.

Accuracy of Compute-$V^\star$ When the above events hold, we can ensure Algorithm 2 always returns $V_h^\star(s_h(a_{0:h-1}))$ with some small deviation error.

Lemma 7 (The accuracy of Compute-$V^\star$)**.** *We set* ε *such that* $\varepsilon = \Delta/(6\Theta)$*. Let* $a^\star_{h:H}(a_{0:h-1}) \in \mathcal{A}^{H-h}$ *be the optimal action sequence from* h *to* $H-1$ *after* $a_{0:h-1}$*. We condition on the events we have mentioned above. Then, in the algorithm, we always have*

$$\textit{Compute-}V^\star(h; a_{0:h-1}) = V_h^\star(s_h(a_{0:h-1})) + \sum_{k=h}^{H-1} \nu_k.$$

Proof. We prove by induction. We want to prove this statement for any query we have in the algorithm. Suppose we already visit Line 11 in Algorithm 1 $m-1$ times. In other words, we are now at the episode at m. Thus, we use induction in the sense that assuming the statement holds in all queries in the previous episodes before m and all queries from level $h+1$ to level $H-1$ in episode m, we want to prove the statement holds for all queries at level h in episode m.

We first start with the base case level $H-1$ at episode m. When $h = H-1$, our procedure simply returns $\max_a \hat{r}_{H-1}(s_{H-1}(a_{0:H-2}), a)$. From the gap assumption, we have $\max_a \hat{r}_{H-1}(s_{H-1}(a_{0:H-2}), a) = \max_{a \in \mathcal{A}} r_{H-1}(s_{H-1}(a_{0:H-2}), a)$ noting we condition on the event the difference is upper-bounded by $\Delta/(6H)$ from (6). Thus, $V^\star_{H-1}(s_{H-1}(a_{0:H-2})) + \nu_{H-1}$ by definition.

Now assume that the conclusion holds for all queries at level $h+1$ in episode m and all queries in the previous episodes before episode m. We prove the statement also holds for all queries at level h in episode m when $h < H-1$.

We divide into two cases.

Case 1: $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} > \varepsilon$ The first case is $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} > \varepsilon$. In this case, we aim to calculate $Q_h^\star(s_h(a_{0:h-1}), a')$ for all $a' \in \mathcal{A}$ by calling Compute-$V^\star$ at layer $h+1$ with input $\{a_{0:h-1}, a'\}$. Note that by inductive hypothesis, we have

$$\text{Compute-}V^\star(h+1; \{a_{0:h-1}, a'\}) = V_{h+1}^\star(s_h(a_{0:h-1}, a')) + \sum_{k=h+1}^{H-1} \nu_k.$$

Hence, from the definition of $y_{a';h}$,

$$y_{a';h} = \hat{r}_h(s_h(a_{0:h-1}, a') + V^\star_{h+1}(s_h(a_{0:h-1}, a')) + \sum_{k=h+1}^{H-1} \nu_k$$
$$= Q^\star_h(s_h(a_{0:h-1}), a') + \sum_{k=h}^{H-1} \nu_k.$$

Thus, noting we condition on the event ν_k are upper-bounded by $\Delta/(6H)$ in the algorithm, we have

$$|y_{a';h} - Q^\star_h(s_h(a_{0:h-1}), a')| \le (H-h)\Delta/(6H).$$

From the gap assumption (Assumption 4), thus $\operatorname{argmax}_{a'} y_{a';h} = \operatorname{argmax}_{a'} Q^\star_h(s_h(a_{0:h-1}), a') = a^\star_h(a_{0:h-1})$. Thus, after choosing the optimal action we return

$$V^\star_h(s_h(a_{0:h-1})) + \sum_{k=h}^{H-1} \nu_k.$$

This implies that the conclusion holds for queries at level h in the first case.

Case 2: $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} \le \varepsilon$ The second case is $\|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} \le \varepsilon$. We first note from the inductive hypothesis, in the data $\mathcal{D}_{a;h}$, for any $\hat{x}_h(a_{0:h-1})$, the corresponding $y_{a;h}$ is

$$y_{a;h} = Q^\star_h(s_h(a_{0:h-1}), a) + \sum_{k=h}^{H} \nu_k$$

Recall that $Q^\star_h(s_h(a_{0:h-1}), a) = (\theta^\star_{a;h})^\top x_h(a_{0:h-1})$. Then, for any $a \in \mathcal{A}$,

$$\begin{aligned}
\hat{\theta}_{a;h} &= \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} \hat{x}_h^{(i)}(a_{0:h-1})\{\langle x_h^{(i)}(a_{0:h-1}), \theta^\star_{a;h}\rangle + z^{(i)}_{a;h}\} \\
&= \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} \hat{x}_h(a_{0:h-1})\{\langle x_h^{(i)}(a_{0:h-1}) - \hat{x}_h^{(i)}(a_{0:h-1}), \theta^\star_{a;h}\rangle + \langle \hat{x}_h^{(i)}(a_{0:h-1}), \theta^\star_{a;h}\rangle + z^{(i)}_{a;h}\} \\
&= \theta^\star_{a;h} - \lambda\Sigma_h^{-1}\theta^\star_{a;h} + \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} \hat{x}_h^{(i)}(a_{0:h-1})\{\langle x_h^{(i)}(a_{0:h-1}) - \hat{x}_h^{(i)}(a_{0:h-1}), \theta^\star_{a;h}\rangle + z^{(i)}_{a;h}\} \\
&= \theta^\star_{a;h} - \lambda\Sigma_h^{-1}\theta^\star_{a;h} + \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} \hat{x}_h^{(i)}(a_{0:h-1}) w^{(i)}_{a;h}
\end{aligned}$$

where $w^{(i)}_{a;h} = \langle \hat{x}_h^{(i)}(a_{0:h-1}) - x_h^{(i)}(a_{0:h-1}), \theta^\star_{a;h}\rangle + z^{(i)}_{a;h}$. On the events we condition ((5) and (6)), we have $|w^{(i)}_{a;h}| \le \text{Error}$ where

$$\text{Error} := \Delta/(6\sqrt{N'\varepsilon}).$$

Using the above, at level h in episode m,

$$\begin{aligned}
&\forall a : \left|\hat{\theta}_{a;h}^\top \hat{x}_h(a_{0:h-1}) - (\theta^\star_{a;h})^\top x_h(a_{0:h-1})\right| \\
&\le \left|(\hat{\theta}_{a;h} - \theta^\star_{a;h})^\top \hat{x}_h(a_{0:h-1})\right| + \left|(\theta^\star_{a;h})^\top(\hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1}))\right| \\
&\le \underbrace{\left|\langle \lambda\Sigma_h^{-1}\theta^\star_{a;h}, \hat{x}_h(a_{0:h-1})\rangle\right|}_{(a)} + \underbrace{\left|\langle \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} \hat{x}_h^{(i)}(a_{0:h-1}) w^{(i)}_{a;h}, \hat{x}_h(a_{0:h-1})\rangle\right|}_{(b)} + \Delta/6. \qquad \text{(Use (5))}
\end{aligned}$$

The first term (a) is upper-bounded by

$$\begin{aligned}
\left|\langle \lambda \Sigma_h^{-1} \theta^\star_{a;h}, \hat{x}_h(a_{0:h-1})\rangle\right| &\le \lambda \|\Sigma_h^{-1}\theta^\star_{a;h}\|_{\Sigma_h} \|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} && \text{(CS inequality)} \\
&\le \sqrt{\lambda}\Theta\varepsilon && (\theta^\star_{a;h} \le \Theta \text{ and } \|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} \le \varepsilon) \\
&\le \Delta/6. && \text{(We set a parameter } \varepsilon \text{ to satisfy this condition)}
\end{aligned}$$

The second term (b) is upper-bounded by

$$\begin{aligned}
&\text{Error} \times \sum_{i=1}^{|\mathcal{D}_{a;h}|} |\hat{x}_h^\top(a_{0:h-1})\Sigma_h^{-1} x_h^{(i)}(a_{0:h-1})| \\
&\le \text{Error} \times \sqrt{|\mathcal{D}_{a;h}| \hat{x}_h^\top \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} x_h^{(i)}(a_{0:h-1}) x_h^{(i)}(a_{0:h-1})^\top \Sigma_h^{-1} \hat{x}_h} && \text{(From L1 norm to L2 norm)} \\
&\le \text{Error} \times \sqrt{N' \hat{x}_h^\top \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} x_h^{(i)}(a_{0:h-1}) x_h^{(i)}(a_{0:h-1})^\top \Sigma_h^{-1} \hat{x}_h} && (N' \text{ upper-bounds } |\mathcal{D}_{a;h}|) \\
&\le \text{Error} \times \sqrt{N'\varepsilon} \\
&\le \Delta/6. && \text{(We set } M, M', \varepsilon \text{ to satisfy this condition)}
\end{aligned}$$

From the third line to the fourth line, we use a general fact when $x^\top (C + \lambda I)^{-1} x \le \varepsilon$ is satisfied, we have $x^\top (C + \lambda I)^{-1} C (C + \lambda I)^{-1} x \le \varepsilon$ for any matrix C and vector x.

Thus, we have that:

$$\forall a : \left| \hat{\theta}_{a;h}^\top \hat{x}_h(a_{0:h-1}) - (\theta^\star_{a;h})^\top x_h(a_{0:h-1}) \right| \le \Delta/2.$$

Together with the gap assumption, this means

$$\operatorname*{argmax}_a \hat{\theta}_{a;h}^\top \hat{x}_h(a_{0:h-1}) = \operatorname*{argmax}_a (\theta^\star_{a;h})^\top x_h(a_{0:h-1}) = a^\star_h(a_{0:h-1}).$$

Thus, we select $a^\star_h(a_{0:h-1})$ at h. Then, when we query Compute-$V^\star(h+1; \{a_{0:h-1}, a^\star_h(a_{0:h-1})\})$, which by inductive hypothesis we return

$$V^\star_{h+1}(s_{h+1}(\{a_{0:h-1}, a^\star_h(a_{0:h-1})\})) + \sum_{k=h+1}^{H-1} \nu_k.$$

Finally, adding the reward, we return

$$V^\star_h(s_h(a_{0:h-1})) + \sum_{k=h}^{H-1} \nu_k.$$

This implies that the conclusion holds for any queries at level h in the second case. □

C.3. Third Step

The next lemma shows that when the algorithm terminates, we must find an exact optimal policy.

Lemma 8 (Optimality upon termination)**.** *Algorithm 1 returns an optimal policy on termination.*

Proof. Recall we denote the optimal trajectory by $\{s^\star_h, a^\star_h\}_{h=0}^{H-1}$ where $s^\star_h = s_h(a^\star_{0:h-1})$. Upon termination of Algorithm 1, we have $\forall h : \|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}} \le \varepsilon$. We prove the theorem by induction. At $h = 0$, we know that $s_0 = s^\star_0$. By our linear regression guarantee as we see in the second step of the proof, we can ensure that for all $a \in \mathcal{A}$,

$$\left|\theta_{a;0}^\top \hat{x}_0 - (\theta^\star_{a;0})^\top x_0\right| \le \Delta/2,$$

which means that $a_0 = \mathrm{argmax}_a \, \theta_{a;0}^\top x_0 = a_0^\star$. This completes the base case.

Now we assume it holds a step 0 to h. We prove the statement for step $h+1$. Thus, we can again use the linear regression guarantee to show that the prediction error for all a must be less than $\Delta/2$, i.e.,

$$\left|\theta_{a;h}^\top \hat{x}_h(a_{0:h-1}^\star) - (\theta_{a;h}^\star)^\top x_h(a_{0:h-1}^\star)\right| \le \Delta/2.$$

This indicates that at $h+1$, we will pick the correct action $a_{h+1}^\star$. This completes the proof. □

Finally, combining lemmas so far, we derive the final sample complexity.

Theorem 8. *With probability $1-\delta$, the algorithms output the optimal actions after using at most the following number of samples*

$$\tilde{O}\left(\frac{H^5 A \Theta^5 d^2 \ln(1/\delta)}{\Delta^5}\right).$$

Here, we ignore $\mathrm{Polylog}(H, d, \ln(1/\delta), 1/\Delta, |\mathcal{A}|, \Theta)$.

Proof. Recall we use the following number of samples:

$$O((M+M')H^3 A d/\varepsilon^2 \ln(d/\varepsilon))$$

Here, the rest of the task is to properly set M, M', ε.

Number of times we use concentration inequalities We use high probability statements to bound three-types of terms:

$$\forall a \in \mathcal{A}; |\langle \theta_{a;h}^\star, \hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1})\rangle| \le 2\Theta\sqrt{\ln(1/\delta')/M}, \tag{7}$$

$$|\nu_k| \le 2\sqrt{\ln(1/\delta')/M'}. \tag{8}$$

Let $N' = O(Hd/\varepsilon^2 \ln(d/\varepsilon))$. We also set $\varepsilon = O(\Delta/\Theta)$. Recall we need to set M and M' such that

$$2\Theta\sqrt{\ln(1/\delta')/M} \le \min\left(\Delta/6, \Delta/(12\sqrt{N'\varepsilon})\right), \quad 2\sqrt{\ln(1/\delta')/M'} \le \min\left(\Delta/(6H), \Delta/(12\sqrt{N'\varepsilon})\right).$$

Thus, we set

$$M = O\left(\frac{\ln(1/\delta')\Theta^3 H d \ln(d\Theta/\Delta)}{\Delta^3}\right), \quad M' = O\left(\frac{\ln(1/\delta')\Theta H^2 d \ln(d\Theta/\Delta)}{\Delta^3}\right).$$

Here, events (7) and (8) are called $O(\sum(\alpha_m + \beta_m))$ times. Thus, we set $\delta' = \delta/(\sum(\alpha_m + \beta_m))$.

Collect all events Recall we need the following number of samples:

$$O\left((M+M')H^3 A^2 d/\varepsilon^2 \ln(d/\varepsilon)\right).$$

Thus, the total sample complexity is

$$N = \tilde{O}\left(\frac{H^2 d \Theta^3 \ln(1/\delta')}{\Delta^3} \times \frac{H^3 A^2 d \times \Theta^2}{\Delta^2}\right)$$

where $1/\delta' = 1/\delta \times O(\sum(\alpha_m + \beta_m))$. Hence,

$$N = \tilde{O}\left(\frac{H^5 A^2 \Theta^5 d^2 \ln(1/\delta)}{\Delta^5}\right).$$

□

D. Proof of Section B

D.1. Proof of Lemma 3

Since $a^\top \phi(s) \in \mathcal{H}_{\bar{\mathcal{S}}}$, it can be written in the form of

$$a^\top \phi(\cdot) = \sum_{i=1}^{\infty} \alpha_i \bar{k}(\cdot, s^{[i]}).$$

where $s^{[i]} \in \mathcal{S}$. Then, it is equal to

$$\begin{aligned}\sum_{i=1}^{\infty} \alpha_i \bar{k}(\cdot, s^{(i)}) &= \sum_{i=1}^{\infty} \alpha_i \mathbb{E}_{o' \sim \mathbb{O}(s^{(i)}), o \sim \mathbb{O}(\cdot)}[\psi^\top(o')\psi(o)] \\ &= \langle \sum_{i=1}^{\infty} \alpha_i \mathbb{E}_{o' \sim \mathbb{O}(s^{(i)})}[\psi^\top(o')], \mathbb{E}_{o \sim \mathbb{O}(\cdot)}[\psi(o)] \rangle\end{aligned}$$

Thus, there exists b s.t. $a^\top \phi(\cdot) = b^\top \mathbb{E}_{o \sim \mathbb{O}(\cdot)}[\psi(o)]$. Finally,

$$\begin{aligned}1 &\geq a^\top \mathbb{E}_{s \sim u_S(s)}[\phi(s)\phi^\top(s)]a = b^\top \mathbb{E}_{s \sim u_S(s)}[\mathbb{E}_{o \sim \mathbb{O}(s)}[\psi(o)]\mathbb{E}_{o \sim \mathbb{O}(s)}[\psi^\top(o')]]b \\ &\geq b^\top \mathbb{E}_{s \sim u_S(s)}[\phi(s)\phi^\top(s)]b(1/\iota)^2 = \|b\|_2^2 (1/\iota)^2.\end{aligned}$$

Hence, $\|b\|_2^2 \leq \iota^2$.

D.2. Proof of Theorem 7

We first introduce several notations. We set $\lambda = 1$.

We define feature vectors $x_h(a_{0:h-1}) = \mathbb{E}_{o \sim \mathbb{O}(s_h(a_{0:h-1}))}[\psi(o)], \hat{x}_h(a_{0:h-1}) = \hat{\mathbb{E}}_{o \sim \mathbb{O}(s_h(a_{0:h-1}))}[\psi(o)]$ where $\hat{\cdot}$ means empirical approximation using M samples. Then,

$$\hat{k}(a_{0:h-1}, a'_{0:h-1}) = \hat{x}_h(a_{0:h-1})^\top \hat{x}_h(a'_{0:h-1}), \quad k(a_{0:h-1}, a'_{0:h-1}) = x_h(a_{0:h-1})^\top x_h(a'_{0:h-1}).$$

Primal representation We mainly use a primal representation in the analysis. Let $Q_h^\star(\cdot, a) = \langle \theta_{a;h}^\star, \phi(s) \rangle$. Then,

$$\begin{aligned}\mu_{a;h}(a_{0:h-1}, \mathcal{D}_{a;h}) &= \hat{x}_h^\top(a_{0:h-1})\hat{\theta}_{a;h}, \quad \hat{\theta}_{a;h} = \Sigma_h^{-1} \sum_{i=1}^{|\mathcal{D}_{a;h}|} y_{a;h}^{(i)} \hat{x}_h(a_{0:h-1}^{(i)}), \\ \sigma_h(a_{0:h-1}, \mathcal{D}_h) &= \|\hat{x}_h(a_{0:h-1})\|_{\Sigma_h^{-1}}, \quad \Sigma_h = \sum_{j=1}^{|\mathcal{D}_h|} \hat{x}_h(a_{0:h-1}^{(j)}) \hat{x}_h(a_{0:h-1}^{(j)})^\top + \lambda I,\end{aligned}$$

Regarding the derivation, for example, refer to (Chowdhury & Gopalan, 2017). [2]

Most of the proof in Section C similarly goes through. We list parts where we need to change as follows:

1. We need to modify (4) in the first step of Section C to upper-bound the number of bad events we encounter.
2. We need to modify Lemma 5.

Then, we can similarly conclude that the sample complexity is $\mathrm{poly}(W, \iota, \ln(1/\delta), H, \Gamma, 1/\Delta, A)$,

[2] Formally, we should use notation based on operators But following the convention on these literature, we use a matrix representation. Every argument is still valid.

First modification Recall $\gamma(N; k_{\mathcal{O}})$ is defined by $\max_{|C|=N} \ln\det(I + \mathbf{K}_C)$ where $\mathbf{K}_C$ is a $N \times N$ matrix where (i,j)-th entry is $k_{\mathcal{O}}(x_i, x_j)$ when $C = \{x_i\}$.

Lemma 9 (Information gain on the estimated feature in RKHS)**.** *Let* $|\mathcal{D}_h| = N$.

$$\ln\det(I + \hat{\mathbf{K}}(\mathcal{D}_h)) \le M\gamma(N; k_{\mathcal{O}}).$$

Proof. Recall $\hat{\mathbf{K}}(\mathcal{D}_h) = 1/M \sum_{j=1}^{M} \mathbf{K}_j$ where $\mathbf{K}_j$ is a $N \times N$ matrix with an entry $\{k_{\mathcal{O}}(\cdot, \diamond)\}_{\cdot \in \mathcal{D}_h, \diamond \in \mathcal{D}_h}$. Then,

$$\begin{aligned}\ln\det(I + 1/M \sum_{j=1}^{M} \mathbf{K}_j) &\le \ln \prod_{i=1}^{M} \det(I + 1/M\mathbf{K}_j)) \\ &\le \sum_{j=1}^{M} \ln(\det(I + 1/M\mathbf{K}_j)) \le M\gamma(N; k_{\mathcal{O}}).\end{aligned}$$

□

Then, in (4) in the first step of Section C, we can use the following inequality

$$\varepsilon N' \le \sum_{i=1}^{N'} \|\hat{x}_h^{(i)}\|_{\Sigma_h^{-1}} \le c\sqrt{N' \ln\det(I + \hat{\mathbf{K}}(D_h))} \le c\sqrt{N' M \gamma(N'; k_{\mathcal{O}})}.$$

Letting $O(\Gamma N'^{\alpha}) = \gamma(N'; k_{\mathcal{O}})$,

$$N' = \left(\frac{\Gamma^{1/2} M^{1/2}}{\varepsilon}\right)^{2/(1-\alpha)}.$$

Second modification We first check the concentration on the estimated feature.

Lemma 10 (Concentration of the estimated feature)**.** *Suppose* $k_{\mathcal{O}}(\cdot,\cdot) \le 1$. *Then,* $(\hat{x}_h(a_{0:h-1}) - x_h(a_{0:h-1}))^\top \theta^\star_{a;h}$ *is a* Θ^2/M *sub-Gaussian random variable.*

Proof. Here,

$$\|\psi(o)^\top \theta^\star_{a;h}\| \le \|\psi(o)\| \|\theta^\star_{a;h}\| \le k_{\mathcal{O}}(o,o)^{1/2}\Theta \le \Theta.$$

Then, we use Hoeffeding's inequality.

□

E. Proof of Section 5

E.1. Proof of Lemma 1

Recall our assumption is

$$\begin{aligned}Q^\star_h(s_h, a) &= \langle w^\star_{a;h}, \phi(s_h)\rangle = \langle w^\star_{a;h}, G_h^\dagger G_h \phi(s_h)\rangle \\ &= \langle \{G_h^\dagger\}^\top w^\star_{a;h}, G_h\phi(s_h)\rangle = \langle \{G_h^\dagger\}^\top w^\star_{a;h}, \mathbb{E}[\psi(o_{h:h+K-1}) \mid s_h; a^\diamond_{h:h+K-2}]\rangle.\end{aligned}$$

Thus, the above is written in the form of $\langle \theta^\star_{a;h}, z_h^{[K]}(s_h)\rangle$ noting $\mathbb{E}[\psi(o_{h:h+K-1}) \mid s_h; a^\diamond_{h:h+K-2}]$ is a sub-vector of $z_h^{[K]}(s_h)$.

E.2. Proof of Lemma 2

Let $\psi(\cdot)$ be a one-hot encoding vector over $\mathcal{S}$. We have

$$Q^\star_h(s,a) = \langle w^\star_{a;h}, \psi(s)\rangle = \langle \{\mathbb{P}_h^{[K]}(a^\diamond_{h:h+K-2})\}^\dagger w^\star_{a;h}, \mathbb{P}_h^{[K]}(a^\diamond_{h:h+K-2})\psi(s)\rangle$$

Since $z_h^{[K]}(s)$ includes $\mathbb{P}_h^{[K]}(a^\diamond_{h:h+K-2})\psi(s)$, the statement is concluded.

E.3. Proof of Theorem 4

Most part of the proof is similarly completed as the proof of Theorem 3. We need to take the following differences into account:

- M needs to be multiplied by A^{K-1},
- d needs to be multiplied by A^{K-1}.

Then, the sample complexity is

$$\tilde{O}\left(\frac{H^5 A^{3K-1}\Theta^5 d^2 \ln(1/\delta)}{\Delta^5}\right).$$

F. Auxiliary lemmas

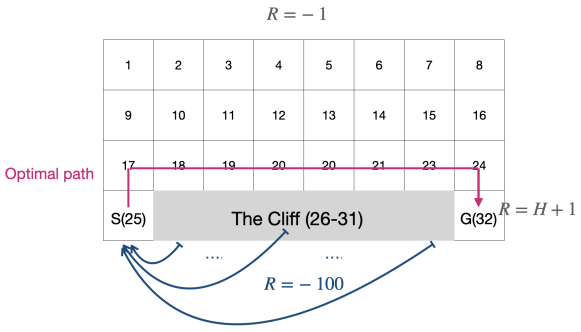
Refer to Agarwal et al. (2019, Chapter 6) for the lemma below.

Lemma 11 (Potential function lemma)**.** *Suppose* $\|X_i\| \leq B$ *and* $\Sigma_i = \sum_{k=1}^{i} X_k X_k^\top$. *Then,*

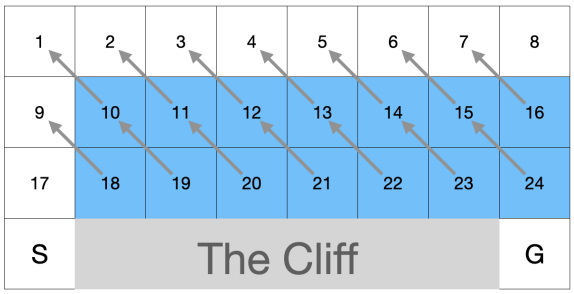
$$\sum_{i=1}^{N} X_i^\top \Sigma_{i-1}^{-1} X_i \leq \ln(\det(\Sigma_N)/\det(\lambda I)) \leq d \ln\left(1 + \frac{NB^2}{d\lambda}\right).$$

G. Experiment

We consider experiments using grid-world environments where we observe noisy observations of the latent state due to imperfect sensors. As mentioned in Section 1, this experiment is motivated by possible practical scenarios in autonomous driving. Similar experimental settings are considered in Du et al. (2019). We demonstrate our proposed method can return the optimal policy with low sample complexity.



(a) Cliff walking with $H = 8$. A state $s = 25$ is an initial state and $s = 32$ is a goal. We obtain reward -1 at $s \in (1, \cdots, 25)$, reward -100 at $s \in (26, \cdots, 31)$ and a reward $H + 1$ at the goal.



(b) Observation process. For example, when $s = 10$, with probability α, we observe 1 and with probability $1 - \alpha$, we observe 10.

Figure 2. Our environment

The environments. We consider the environment, "cliff walking" as illustrated in Figure 2a. The implementation of the environment is given in `https://github.com/openai/gym/blob/master/gym/envs/toy_text/cliffwalking.py`. [3]. The number of actions is 4 and the number of state space is $4 \times H$ where H is the number of columns. There are four actions "up, down, right, and left". We get a reward -100 at the cliff, a reward H at the goal, and

[3] In the original implementation, the reward at the goal is 0 and $H = 12$. In our setting, we vary H and set the reward at the goal to be $H + 1$ so that the cumulative reward of the optimal trajectory is 0.

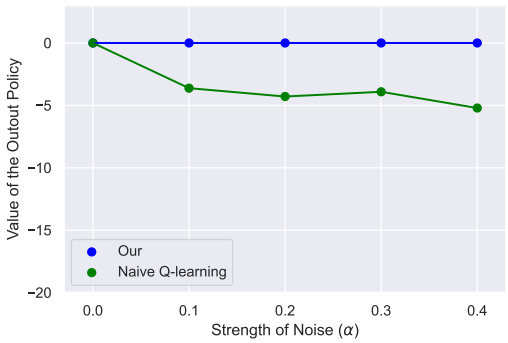
get a reward -1 at other states. The optimal path, which has a cumulative reward 0, is depicted in pink. We want to identify this optimal path in a sample efficient manner.

In our experiment, we consider a scenario where we can only get noise observations but not latent states. The observation process is illustrated in Figure 2b. When agents are in the blue states, with probability α, observations are given as $s - H - 1$ and with probability $1 - \alpha$, we observe s. Note these environments do not belong to block MDPs since we cannot uniquely decode latent states from the whole historical observations. For example, when we observe a sequence $(s_0, s_1, s_2) = (25, 17, 18)$, the possible latent state at $h = 2$ is 9 or 18.

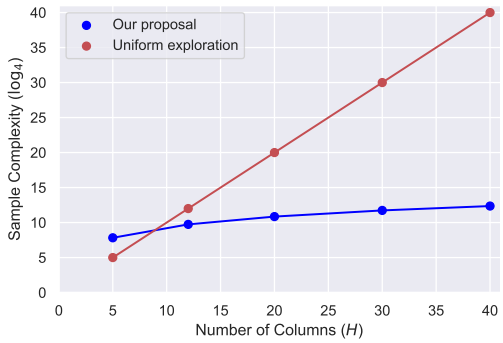
Baseline, hyperparameters. The most naive baseline is exploring all four directions at every time step and selecting the action sequence with the highest value. This approach requires at least $4^{(H+1)}$ samples, which is prohibitively large. Additionally, we consider the following two methods.

- Our proposal. We set $M = 50, \epsilon = 0.6, \lambda = 0.1$ unless otherwise noted. We confirm our proposal is robust to hyperparameters as will be discussed later.
- Naive Q-learning with $\epsilon\,(= 0.1)$ greedy action selection that uses current observations as inputs. This is a standard method for MDPs. It is expected to fail since Markovinity breaks down.

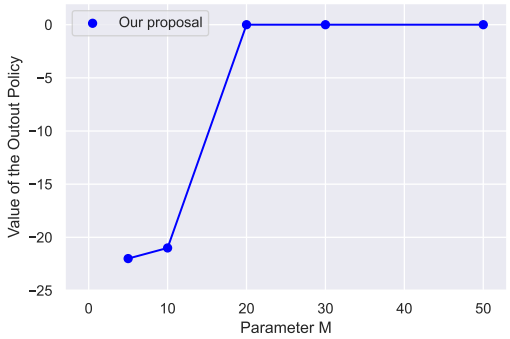
Note methods for block MDPs (Du et al., 2019) are expected to fail since our environment is not a block MDP.



(a) Comparison between our proposal and the naive Q-learning in terms of the quality of outputs



(b) Comparison between our proposal and uniform exploration in terms of sample complexities



(c) Sensitivity analysis

Figure 3. Results of experiments

Results. We first confirm our methods can return the optimal policy in high probability. The result is illustrated Figure 3a. We run our proposal and the naive Q-learning 10 times and take each average of the values of output policies. We set

$H = 20$ and vary $\alpha \in (0, 0.1, 0.2, 0.3, 0.4)$. Recall the value of the optimal policy is 0. It is seen that our proposal always returns the optimal policy. In contrast, the naive Q-learning fails to identify the optimal policy when $\alpha \neq 0$.

Next, we investigate the sample complexity, which is the number of samples to get the optimal policy. We set $\alpha = 0.3$ and vary $H \in (5, 12, 20, 30, 40)$. The result is illustrated in Figure 3b. We calculate the sample complexity of our proposal by running algorithms 10 times and taking the average. Since the naive Q-learning method cannot return the optimal policy , it is not included in the graph. It is seen that compared to uniform exploration, our proposal is much more efficient and the sample complexity does not grow exponentially in horizon.

Finally, we confirm the robustness of our proposal in Figure 3c. We set $H = 20, \alpha = 0.3$. We vary $M \in (20, 50, 100, 200)$. Figure 3c demonstrates the proposal can identify the optimal policy as long as M is larger than 20. Practically, we recommend we repeat running the algorithm by gradually increasing M every time till the value of the output policy is stable. Besides, we confirm our proposal is robust to ϵ and λ in that the proposal can identify the optimal policy when ϵ and λ are less than 1 if M is large enough.