Markov Decision Processes with Continuous Side Information

Aditya Modi ADMODI@UMICH.EDU
Computer Science and Engineering, University of Michigan Ann Arbor

Nan Jiang NANJIANG@UMICH.EDU
Microsoft Research, New York

Satinder Singh BAVEJA@UMICH.EDU
Computer Science and Engineering, University of Michigan Ann Arbor

Ambuj Tewari TEWARIA@UMICH.EDU
Department of Statistics, University of Michigan Ann Arbor

Editors: Mehryar Mohri and Karthik Sridharan

Abstract

We consider a reinforcement learning (RL) setting in which the agent interacts with a sequence of episodic MDPs. At the start of each episode the agent has access to some side-information or context that determines the dynamics of the MDP for that episode. Our setting is motivated by applications in healthcare where baseline measurements of a patient at the start of a treatment episode form the context that may provide information about how the patient might respond to treatment decisions.

We propose algorithms for learning in such Contextual Markov Decision Processes (CMDPs) under an assumption that the unobserved MDP parameters vary smoothly with the observed context. We give lower and upper PAC bounds under the smoothness assumption. Because our lower bound has an exponential dependence on the dimension, we also consider a tractable linear setting where the context creates linear combinations of a finite set of MDPs. For the linear setting, we give a PAC learning algorithm based on KWIK learning techniques.

Keywords: Reinforcement Learning, PAC bounds, KWIK Learning.

1. Introduction

Consider a basic sequential decision making problem in healthcare, namely that of learning a treatment policy for patients to optimize some health outcome of interest. One could model the interaction with every patient as a Markov Decision Process (MDP). In *precision or personalized medicine*, we want the treatment to be personalized to every patient. At the same time, the amount of data available on any given patient may not be enough to personalize well. This means that modeling each patient via a different MDP will result in severely suboptimal treatment policies. The other extreme of pooling all patients' data results in more data but most of it will perhaps not be relevant to the patient we currently want to treat. We therefore face a trade-off between having a large amount of shared data to learn a single policy, and, finding the most relevant policy for each patient. A similar trade-off occurs in other applications in which the agent's environment involves humans, such as in online tutoring and web advertising.

A key observation is that in many personalized decision making scenarios, some side information is available about individuals which might help in designing personalized policies and also help pool the interaction data across the right subsets of individuals. Examples of such data include laboratory data or medical history of patients in healthcare, user profiles or history logs in web advertising, and student profiles or historical scores in online tutoring. Access to such side information should allow learning of better policies even with a limited amount of interaction with individual users. We refer to this side-information as *context* and adopt an augmented model called *Contextual Markov Decision Process* (CMDP) proposed by Hallak et al. (2015). We assume that contexts are fully observed and available before the interaction starts for each new MDP.[1]

In this paper we study the sample complexity of learning in CMDPs when contexts can potentially be generated adversarially. We consider two concrete settings of learning in a CMDP with continuous contexts. In the first setting, the individual MDPs vary in an arbitrary but smooth manner with the contexts, and we propose our Cover-Rmax algorithm in Section 3 with PAC (Probably Approximately Correct) bounds. The innate hardness of learning in this general case is captured by our lower bound construction in Section 3.1. To show that it is possible to achieve significantly better sample complexity in more structured CMDPs, we consider another setting where contexts create linear combinations of a finite set of fixed but unknown MDPs. We use the KWIK (Knows What It Knows) framework to devise the KWIK_LR-Rmax algorithm in Section 4 and also provide a PAC upper bound for the algorithm.

2. Contextual Markov Decision Process

In this section, we describe the problem formulation, followed by the interaction protocol between the environment and the learner. We also introduce the criterion used to judge learning algorithms in our setting.

2.1. Problem setup and notation

We start with basic definitions and notations, and then introduce the contextual case.

Definition 1 (Markov Decision Processes) *A Markov Decision Process (MDP) is defined as a tuple $(\mathcal{S}, \mathcal{A}, p(\cdot|\cdot,\cdot), r(\cdot,\cdot), \mu)$ where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space; $p(s'|s,a)$ defines the transition probability function for a tuple (s,a,s') where $s,s' \in \mathcal{S}, a \in \mathcal{A}$; and μ defines the initial state distribution for the MDP.*

We consider episodic MDPs with fixed horizon H. For each episode, an initial state s_0 is observed according to the distribution μ and afterwards, for $0 \leq h < H$, the agent chooses an action $a_h = \pi_h(s_h)$ according to a (non-stationary) policy π. There is a reward r_h and then a next state s_{h+1} according to the reward and the transition functions. The *value* for

[1] Hallak et al. (2015) assumes *latent* contexts, which results in significant differences from our work in application scenarios, required assumptions, and results. See detailed discussion in Section 5.

policy π is defined as follows:

$$V_M^\pi = \mathbb{E}_{s_0 \sim \mu, M, \pi}\Big[\frac{1}{H}\sum_{h=0}^{H-1} r(s_h, \pi_h(s_h))\Big]. \tag{1}$$

An optimal policy π^* is one that achieves the largest possible value (called optimal value and denoted V_M^*). Next we introduce the contextual model, inspired by Hallak et al. (2015):

Definition 2 (Contextual MDP) *A contextual Markov Decision Process (CMDP) is defined as a tuple $(\mathcal{C}, \mathcal{S}, \mathcal{A}, \mathcal{M})$ where $\mathcal{C}$ is the context space (assumed to lie in some Euclidean space), $\mathcal{S}$ is the state space, and $\mathcal{A}$ is the action space. $\mathcal{M}$ is a function which maps a context $c \in \mathcal{C}$ to MDP parameters $\mathcal{M}(c) = \{p^c(\cdot|\cdot,\cdot), r^c(\cdot,\cdot), \mu^c(\cdot)\}$.*

The MDP for a context c is denoted by M^c. For simplification, the initial state distribution is assumed to be the same irrespective of the context and rewards are assumed to be bounded between 0 and 1. We denote $|\mathcal{S}|, |\mathcal{A}|$ by S, A respectively. We also assume that the context space is bounded, and for any $c \in \mathcal{C}$ the ℓ_2 norm of c is upper bounded by some constant.

2.2. Protocol and Efficiency Criterion

We consider the online learning scenario with the following protocol: For $t = 1, 2, \ldots$:

1. Observe context $c_t \in \mathcal{C}$.
2. Choose a policy π_t (based on c_t and previous episodes).
3. Experience an episode in M^{c_t} using π_t.

The protocol does not make any distributional assumptions over the context sequence. Instead, the context sequence can be chosen in an arbitrary and potentially *adversarial* manner. A natural criteria for judging the efficiency of the algorithm is to look at the number of episodes where it performs sub-optimally. The main aim of the PAC analysis is to bound the number of episodes where $V_{M^{c_t}}^{\pi_t} < V_{M^{c_t}}^* - \epsilon$, i.e., the value of the algorithm's policy is not ϵ-optimal (Dann and Brunskill, 2015). Although, we do give PAC bounds for the Cover-Rmax algorithm given below, the reader should make note that, we have not made explicit attempts to achieve the tightest possible result. We use the Rmax (Brafman and Tennenholtz, 2002) algorithm as the base of our construction to handle exploration-exploitation because of its simplicity. Our approach can also be combined with the other PAC algorithms (Strehl and Littman, 2008; Dann and Brunskill, 2015) for improved dependence on S, A and H.

3. Cover-Rmax

The key motivation for our contextual setting is that sharing information/data among different contexts might be helpful. A natural way to capture this is to assume that the MDPs corresponding to similar contexts will themselves be similar. This can be formalized by the following smoothness assumption:

Definition 3 (Smoothness) *Given a CMDP $(\mathcal{C}, \mathcal{S}, \mathcal{A}, \mathcal{M})$ and a distance metric over the context space $\phi(\cdot,\cdot)$, if for any two contexts c_1, $c_2 \in \mathcal{C}$, we have the following constraints:*

$$\begin{aligned}\|p^{c_1}(\cdot|s,a) - p^{c_2}(\cdot|s,a)\|_1 &\le L_p \phi(c_1, c_2)\\ |r^{c_1}(s,a) - r^{c_2}(s,a)| &\le L_r \phi(c_1, c_2)\end{aligned}$$

then, the CMDP is referred as a smooth CMDP with smoothness parameters L_p and L_r.

Throughout the paper, the distance metric and the constants L_p and L_r are assumed to be known. This smoothness assumption allows us to use a modified version of Rmax (Brafman and Tennenholtz, 2002) and provide an analysis for smooth CMDPs similar to existing literature on PAC bounds in MDPs (Kearns and Singh, 2002; Strehl et al., 2009; Strehl and Littman, 2008). Given the transition dynamics and the expected reward functions for each state-action pair in a finite MDP, computing the optimal policy is straightforward. The idea of Rmax is to distinguish the state-action pairs as *known* or *unknown*: a state-action pair is known if it has been visited enough number of times, so that the empirical estimates of reward and transition probabilities are near-accurate due to sufficient data. A state s becomes *known* when (s, a) is *known* for all actions a. Rmax then constructs an auxiliary MDP which encourages optimistic behaviour by assigning maximum reward (hence the name Rmax) to the remaining *unknown* states. The optimal policy in the auxiliary MDP ensures that one of the following must happen: 1) it achieves near-optimal value, or, 2) it visits unknown states and accumulates more information efficiently.

Formally, for a set of known states K, we define an (approximate) *induced MDP* $\hat{M}_K$ in the following manner. Let $n(s, a)$ and $n(s, a, s')$ denote the number of observations of state-action pair (s, a) and transitions (s, a, s') respectively. Also, let $R(s, a)$ denote the total reward obtained from state-action pair (s, a). For each $s \in K$, define the values

$$\begin{aligned}p_{\hat{M}_K}(s'|s,a) &= \frac{n(s,a,s')}{n(s,a)},\\ r_{\hat{M}_K}(s,a) &= R(s,a)/n(s,a).\end{aligned} \tag{2}$$

For each $s \notin K$, define the values as $p_{\hat{M}_K}(s'|s,a) = \mathbb{I}\{s' = s\}$ and $r_{\hat{M}_K}(s,a) = 1$.

Rmax uses the certainty equivalent policy computed for this induced MDP and performs balanced wandering (Kearns and Singh, 2002) for unknown states. Balanced wandering ensures that all actions are tried equally and fairly for *unknown* states. Assigning maximum reward to the *unknown* states pushes the agent to visit these states and provides the necessary exploration impetus. The generic template of Rmax is given in Algorithm 1.

For the contextual case, there would be an infinite number of such MDPs. The idea behind our algorithm is that, close enough contexts can be grouped together and be considered as a single MDP. Utilizing the boundedness of the context space $\mathcal{C}$, we can create a *cover* of $\mathcal{C}$ with finitely many balls $B_r(o_i)$ of radius r centered at $o_i \in \mathbb{R}^d$. The bias introduced by ignoring the differences among the MDPs in the same ball can be controlled by tuning the radius r. Doing so allows us to pool together the data from all MDPs in a ball, so that we avoid the difficulty of infinite MDPs and instead only deal with finitely many of them. The size of the cover, i.e. the number of balls, can be measured by the notion of *covering numbers* (see Section 5.1 in Wainwright (2017)), defined as

$$\mathcal{N}(\mathcal{C}, r) = \min\{|\mathcal{Y}| : \mathcal{C} \subseteq \cup_{y\in\mathcal{Y}} B_r(y)\}.$$

Algorithm 1: Rmax Template for CMDP

```
$Initialize(S, A, \mathcal{C}, \epsilon, \delta)$
for each episode $t = 1, 2, \cdots$ do
    Receive context $c_t \in \mathcal{C}$
    Set $K$, $\hat{M}_K$ using $Predict(c_t, s, a)$ for all $(s, a)$. $\pi \leftarrow \pi^*_{\hat{M}_K}$
    for $h = 0, 1, \cdots H - 1$ do
        if $s_h \in K$ then
            Choose $a_h := \pi_h(s_h)$
        else
            Choose $a_h : (s_h, a_h)$ is unknown
            $Update(c_t, s_h, a_h, (s_{h+1}, r_h))$
        end
    end
end
```

The resulting algorithm, Cover-Rmax, is obtained by using the subroutines in Algorithm 2, and we state its sample complexity guarantee in Theorem 4.

Algorithm 2: Cover-Rmax

```
Function Initialize($\mathcal{S}, \mathcal{A}, \mathcal{C}, \epsilon, \delta$)
    Create an $r_0$-cover of $\mathcal{C}$ with $r_0 = \min(\frac{\epsilon}{8HL_p}, \frac{\epsilon}{8L_r})$
    Initialize counts for all balls $\mathcal{B}(o_i)$
Function Predict($c, s, a$)
    Find $j$ such that $c \in \mathcal{B}(o_j)$
    if $n_j(s, a) < m$ then
        return $\hat{p}^c(\cdot|s, a)$ and $\hat{r}^c(s, a)$ using (2)
    else
        return unknown
    end
Function Update($c, s, a, (s', r)$)
    Find $j$ such that $c \in \mathcal{B}(o_j)$
    if $n_j(s, a) < m$ then
        Increment counts and rewards in $\mathcal{B}(o_j)$
    end
```

Theorem 4 (PAC bound for Cover-Rmax) *For any input values $0 < \epsilon, \delta \leq 1$ and a CMDP with smoothness parameters L_p and L_r, with probability at least $1 - \delta$, the Cover-Rmax algorithm produces a sequence of policies $\{\pi_t\}$ which yield at most*

$$\mathcal{O}\Big(\frac{NH^2SA}{\epsilon^3}\big(S + \ln\frac{NSA}{\delta}\ln\frac{N}{\delta}\big)\Big)$$

non-ϵ-optimal episodes, where $N = \mathcal{N}(\mathcal{C}, r_0)$ and $r_0 = \min(\frac{\epsilon}{8HL_p}, \frac{\epsilon}{8L_r})$.

Proof We first of all carefully adapt the analysis of Rmax by Kakade (2003) to get the PAC bound for an episodic MDP. Let m be the number of visits to a state-action pair after which the model's estimate $\hat{p}(\cdot|s,a)$ for $p(\cdot|s,a)$ has an ℓ_1 error of at most $\epsilon/4H$ and reward estimate $\hat{r}(s,a)$ has an absolute error of at most $\epsilon/4$. We can show that:

Lemma 5 *Let M be an MDP with the fixed horizon H. If $\hat{\pi}$ is the optimal policy for $\hat{M}_K$ as computed by Rmax, then for any starting state s_0, with probability at least $1-2\delta$, we have $V_M^{\hat{\pi}} \geq V_M^* - 2\epsilon$ for all but $\mathcal{O}(\frac{mSA}{\epsilon}\ln\frac{1}{\delta})$ episodes.*

Now instead of learning the model for each contextual MDP separately, the algorithm combines the data within each ball. To control the bias induced by sharing data, the radius r for the cover is set to be $r \leq r_0 = \min\left(\frac{\epsilon}{8HL_p}, \frac{\epsilon}{8L_r}\right)$. Further, the value of m, which is the number of visits after which a state becomes *known* for a ball, is set as $m = \frac{128(S\ln 2 + \ln\frac{SA}{\delta})H^2}{\epsilon^2}$. This satisfies the assumptions in Lemma 5, whereby, we obtain an upper bound on number of non-ϵ episodes in a single ball (generated by Cover-Rmax) as $\mathcal{O}\left(\frac{H^2SA}{\epsilon^3}\left(S + \ln\frac{SA}{\delta}\ln\frac{1}{\delta}\right)\right)$ with probability at least $1-\delta$.

Setting the individual failure probability to be $\delta/N(\mathcal{C}, r_0)$ and using the union bound, we get the stated PAC bound. A detailed proof can be found in Appendix A. ■

The obtained PAC bound has linear dependence on covering number of the context space. In case of a d-dimensional Euclidean metric space, the covering number is of the order $O(\frac{1}{r^d})$. However, we show in Section 3.1, that, the dependence on the covering number is at least linear in the worst case and indicate the difficulty of optimally learning in such cases.

3.1. Lower Bound

We prove a lower bound on the number of sub-optimal episodes for any learning algorithm in a smooth CMDP which shows that a linear dependence on the covering number of the context space is unavoidable. As far as we know, there is no existing way of constructing PAC lower bounds for continuous state spaces with smoothness, so we cannot simply augment the state representation to include context information. Instead, we prove our own lower bound in Theorem 6 which builds upon the work of Dann and Brunskill (2015) on lower bounds for episodic finite MDPs and of Slivkins (2014) on lower bounds for contextual bandits.

Theorem 6 (Lower bound for smooth CMDP) *There exists constants δ_0, ϵ_0, such that for every $\delta \in (0, \delta_0)$ and $\epsilon \in (0, \epsilon_0)$, any algorithm that satisfies a PAC guarantee for (ϵ, δ) and computes a sequence of deterministic policies for each context, there is a hard CMDP $(\mathcal{C}, \mathcal{S}, \mathcal{A}, \mathcal{M})$ with smoothness constant $L_p = 1$, such that*

$$\mathbb{E}[B] = \Omega\left(\frac{\mathcal{N}(\mathcal{C}, \epsilon_1)SA}{\epsilon^2}\right) \tag{3}$$

where B is the number of sub-optimal episodes and $\epsilon_1 = \frac{1280H\epsilon e^4}{(H-2)}$.

The overall idea is to embed multiple MDP learning problems in a CMDP, such that the agent has to learn the optimal policy in each MDP separately and cannot generalize across them. We show that the maximum number of problems that can be embedded scales with the covering number, and the result follows by incorporating known PAC lower bound for episodic MDPs. We refer the reader to Appendix B for the proof.

4. Contextual Linear Combination of MDPs

From the previous section, it is clear that for a contextual MDP with just smoothness assumptions, exponential dependence on context dimension is unavoidable. Further, the computational requirements of our Cover-Rmax algorithm scales with the covering number of the context space. As such, in this section, we focus on a more structured assumption about the mapping from context space to MDPs and show that we can achieve substantially improved sample and computational efficiency.

The specific assumption we make in this section is that the model parameters of an individual MDP M^c is the linear combination of the parameters of d base MDPs, i.e.,

$$\begin{aligned} p^c(s'|s,a) &= c^\top \begin{bmatrix} p_1(s'|s,a) \\ \vdots \\ p_d(s'|s,a) \end{bmatrix} := c^\top P(s,a,s'), \\ r^c(s,a) &= c^\top \begin{bmatrix} r_1(s,a) \\ \vdots \\ r_d(s,a) \end{bmatrix} := c^\top R(s,a). \end{aligned} \tag{4}$$

We use $P(s,a,s')$ and $R(s,a)$ as shorthand for the $d \times 1$ vectors that concatenate the parameters from different base MDPs for the same s,a (and s'). The parameters of the base MDPs (p_i and r_i) are unknown and need to be recovered from data by the learning agent, and the combination coefficients are directly available which is the context vector c itself. This assumption can be motivated in an application scenario where the user/patient responds according to her characteristic distribution over d possible behavioural patterns.

A mathematical difficulty here is that for an arbitrary context vector $c \in \mathbb{R}^d$, $p^c(\cdot|\cdot,\cdot)$ is not always a valid transition function and may violate non-negativity and normalization constraints. Therefore, we require that $c \in \Delta_{d-1}$, that is, c stays in the probability simplex so that $p^c(\cdot|\cdot,\cdot)$ is always valid.[2]

4.1. KWIK_LR-Rmax

We first explain how to estimate the model parameters in this linear setting, and then discuss how to perform exploration properly.

Model estimation Recall that in Section 3, the Cover-Rmax algorithm treats the MDPs whose contexts fall in a small ball as a single MDP, and estimates its parameters using data from the *local* context ball. In this section, however, we have a *global* structure due to our parametric assumption (d base MDPs that are shared across all contexts). This implies that data obtained at a context may be useful for learning the MDP parameters at another context that is far away, and to avoid the exponential dependence on d we need to leverage this structure and generalize globally across the entire context space.

Due to the linear combination setup, we use linear regression to replace the estimation procedure in Equation 2: in an episode with context c, when we observe the state-action pair (s,a), a next-state s_{next} will be drawn from $p^c(\cdot|s,a)$.[3] Therefore, the indicator of whether

[2] Δ_n is the n-simplex $\{x \in \mathbb{R}^{n+1} : \sum_{i=1}^{n+1} x_i = 1, x_i \geq 0\}$.

[3] Here we use s_{next} to denote the random variable, and s' to denote a possible realization.

s_{next} is equal to s' forms an unbiased estimate of $p^c(s'|s,a)$, i.e., $\mathbb{E}_{s_{\text{next}} \sim p^c(\cdot|s,a)}\left[\mathbb{I}[s_{\text{next}} = s']\right] = p^c(s'|s,a) = c^\top P(s,a,s')$. Based on this observation, we can construct a feature-label pair

$$(c,\, \mathbb{I}[s_{\text{next}} = s']) \tag{5}$$

whenever we observe a transition tuple (s, a, s_{next}) under context c, and their relationship is governed by a linear prediction rule with $P(s,a,s')$ being the coefficients. Hence, to estimate $P(s,a,s')$ from data, we can simply collect the feature-label pairs that correspond to this particular (s,a,s') tuple, and run linear regression to recover the coefficients. The case for reward function is similar, hence, not discussed.

If the data is abundant (i.e., (s,a) is observed many times) and exploratory (i.e., the design matrix that consists of the c vectors for (s,a) is well-conditioned), we can expect to recover $P(s,a,s')$ accurately. But how to guarantee these conditions? Since the context is chosen adversarially, the design matrix can indeed be ill-conditioned.

Observe, however, when the matrix is ill-conditioned and new contexts lie in the subspace spanned by previously observed contexts, we can make accurate predictions despite the inability to recover the model parameters. An *online* linear regression (LR) procedure will take care of this issue, and we choose KWIK_LR (Walsh et al., 2009) as such a procedure.

The original KWIK_LR deals with scalar labels, which can be used to decide whether the estimate of $p^c(s'|s,a)$ is sufficiently accurate (*known*). A (s,a) pair then becomes known if (s,a,s') is known for all s'. This approach, however, generally leads to a loose analysis, because there is no need to predict $p^c(s'|s,a)$ for each individual s' accurately: if the estimate of $p^c(\cdot|s,a)$ is close to the true distribution under L_1 error, the (s,a) pair can already be considered as known. We extend the KWIK_LR analysis to handle vector-valued outputs, and provide tighter error bounds by treating $p^c(\cdot|s,a)$ as a whole. Below we introduce our extended version of KWIK_LR, and explain how to incorporate the knownness information in Rmax skeleton to perform efficient exploration.

Identifying known (s,a) with KWIK_LR

The KWIK_LR-Rmax algorithm we propose for the linear setting still uses Rmax template (Algorithm 1) for exploration: in every episode, it builds the induced MDP $\hat{M}_K$, and acts greedily according to its optimal policy with balanced wandering. The major difference from Cover-Rmax lies in how the set of known states K are identified and how $\hat{M}_K$ is constructed, which we explain below (see pseudocode in Algorithm 3).

At a high level, the algorithm works in the following way: when constructing $\hat{M}_K$, the algorithm queries the KWIK procedure for estimates $\hat{p}^c(\cdot|s,a)$ and $\hat{r}^c(s,a)$ for every pair (s,a) using *Predict*(c,s,a). The KWIK procedure either returns $\perp$ (don't know), or returns estimates that are guaranteed to be accurate. If $\perp$ is returned, then the pair (s,a) is considered as unknown and s is associated with $R_{\max}$ reward for exploration. Such optimistic exploration ensures significant probability of observing (s,a) pairs on which the method predicts $\perp$. If such pairs are observed in an episode, KWIK_LR-Rmax calls *Update* with feature-label pairs formed via Equation 5 to make progress on estimating parameters for unknown state-action pairs.

Next we walk through the pseudocode and explain how *Predict* and *Update* work in detail. Then we prove an upper bound on the number of updates that can happen (i.e., the **if** condition holds on Line 11), which forms the basis of our analysis of KWIK_LR-Rmax.

In Algorithm 3, matrices Q and W are initialized for each (s,a) using $Initialize(\cdot)$ and are updated over time. Let $C_t(s,a)$ be the design matrix at episode t, where each row is a context c_τ such that (s,a) was observed in episode $\tau < t$. By matrix inverse rules, we can verify that the update rule on Line 10 essentially yields $Q_t(s,a) = (I + C_t^\top C_t)^{-1}$, where $Q_t(s,a)$ is the value of $Q(s,a)$ in episode t. This is the inverse of the (unnormalized and regularized) empirical covariance matrix, which plays a central role in linear regression analysis. The matrix W accumulates the outer product between the feature vector (context) c and the one-hot vector label $y = (\{\mathbb{I}[s_{\text{next}} = s']\}_{\forall s' \in \mathcal{S}})^\top$. It is then obvious that $Q_t(s,a)W_t(s,a)$ is the linear regression estimate of $P(s,a)$ using the data up to episode t. When a new input vector c_t comes, the algorithm checks whether $\|Q(s,a)c_t\|_2$ is below a predetermined threshold α_S (Line 5). Recall that $Q(s,a)$ is the inverse covariance matrix, so a small $\|Q(s,a)c_t\|_2$ implies that the estimate $Q_t(s,a)W_t(s,a)$ is close to $P(s,a)$ along the direction of c_t, so it predicts $p^{c_t}(\cdot|s,a) = c_t^\top P(s,a) \approx c_t^\top Q(s,a)W(s,a)$; otherwise returns $\perp$. The KWIK subroutine for rewards is similar hence omitted. To ensure that the estimated transition probability is valid, the estimated vector is projected onto Δ_{S-1}, which can be done efficiently using existing techniques (Duchi et al., 2008).

Below we state the KWIK bound for learning the transition function; the KWIK bound for learning rewards is much smaller hence omitted here. We use the KWIK bound for scalar linear regression from Walsh et al. (2009) and the property of multinomial samples to get our KWIK bound.

Theorem 7 (KWIK_LR bound for learning multinomial vectors) *For any $\epsilon > 0$ and $\delta > 0$, if the KWIK_LR algorithm is executed for probability vectors $p_t(\cdot|s,a)$, with $\alpha_S = \min\{b_1 \frac{\epsilon^2}{d^{3/2}}, b_2 \frac{\epsilon^2}{\sqrt{d}\log(d2^S/\delta)}, \frac{\epsilon}{2\sqrt{d}}\}$ with suitable constants b_1 and b_2, then the number of $\perp$'s where updates take place (see Line 11) will be bounded by $\mathcal{O}(\frac{d^2}{\epsilon^4}\max\{d^2, S^2\log^2(d/\delta')\})$, and, with probability at least $1-\delta$, $\forall c_t$ where a non-"$\perp$" prediction is returned, $\|\hat{p}_t^{c_t}(\cdot|s,a) - p_t^{c_t}(\cdot|s,a)\|_1 \leq \epsilon$.*

Proof (See full proof in Appendix C.) We provide a direct reduction to KWIK bound for learning scalar values. The key idea is to notice that for any vector $v \in \mathbb{R}^S$:

$$\|v\|_1 = \sup_{f \in \{-1,1\}^S} v^\top f.$$

So conceptually we can view Algorithm 3 as running 2^S scalar linear regression simultaneously, each of which projects the vector label to a scalar by a fixed linear transformation f. We require every scalar regressor to have $(\epsilon, \delta/2^S)$ KWIK guarantee, and the ℓ_1 error guarantee for the vector label follows from union bound. ■

With this result, we are ready to prove the formal PAC guarantee for KWIK_LR-Rmax.

Theorem 8 (PAC bound for KWIK_LR-Rmax) *For any input values $0 < \epsilon, \delta \leq 1$ and a linear CMDP model with d number of base MDPs, with probability $1-\delta$, the KWIK_LR-Rmax algorithm, produces a sequence of policies $\{\pi_t\}$ which yield at most*

$$\mathcal{O}\Big(\frac{d^2 H^4 SA}{\epsilon^5} \log\tfrac{1}{\delta} \max\{d^2, S^2 \log^2(\tfrac{dSA}{\delta})\}\Big)$$

non-ϵ-optimal episodes.

Algorithm 3: KWIK learning of $p^c(\cdot|s,a)$

```
1  Function Initialize($S, d, \alpha_S$)
2      $Q(s,a) \leftarrow I_d$ for all $(s,a)$
3      $W(s,a) \leftarrow \{0\}^{d\times S}$ for all $(s,a)$
4  Function Predict($c, s, a$)
5      if $\|Q(s,a)c\|_1 \leq \alpha_S$ then
6          return $\hat{p}^c(\cdot|s,a) = c^\top Q(s,a)W(s,a)$
7      else
8          return $\hat{p}^c(\cdot|s,a) = \perp$
9      end
10 Function Update($c, s, a, s_{next}$)
11     if $\|Q(s,a)c\|_1 > \alpha_S$ ("$\perp$" prediction) then
12         $Q(s,a) \leftarrow Q(s,a) - \frac{(Q(s,a)c)(Q(s,a)c)^\top}{1+c^\top Q(s,a)c}$
13         $y \leftarrow (\{\mathbb{I}[s_{\text{next}} = s']\}_{\forall s' \in \mathcal{S}})^\top$
14         $W(s,a) \leftarrow W(s,a) + cy$
15     end
```

Proof When the KWIK subroutine (Algorithm 3) makes non-"$\perp$" predictions $\hat{p}^c(s,a,s')$, we require that

$$\|\hat{p}^c(\cdot|s,a) - p^c(\cdot|s,a)\|_1 \leq \epsilon/8H.$$

After projection onto Δ_{S-1}, we have:

$$\|\Pi_{\Delta_{S-1}}(\hat{p}^c(s,a)) - p^c(\cdot|s,a)\|_1 \leq 2\|\hat{p}^c(\cdot|s,a) - p^c(\cdot|s,a)\|_1 \leq \epsilon/4H.$$

Further, the update to the matrices Q and W happen only when an unknown state action pair (s,a) is visited and the KWIK subroutine still predicts $\perp$ (Line 10). The KWIK bound states that after a fixed number of updates to an unknown (s,a) pair, the parameters will always be known with desired accuracy. The number of updates m can be obtained by setting the desired accuracy in transitions to $\epsilon/8H$ and failure probability as δ/SA in Theorem 7:

$$m = \mathcal{O}\big(\tfrac{d^2H^4}{\epsilon^4}\max\{d^2, S^2\log^2(\tfrac{dSA}{\delta})\}\big)$$

We now use Lemma 5 where instead of updating counts for number of visits, we look at the number of updates for unknown (s,a) pairs. On applying a union bound over all state action pairs and using Lemma 5, it is easy to see that the sub-optimal episodes are bounded by $\mathcal{O}\left(\frac{mSA}{\epsilon}\ln\frac{1}{\delta}\right)$ with probability at least $1-\delta$. The bound in Theorem 8 is obtained by substituting the value of m. ■

We see that for this contextual MDP, the linear structure helps us in avoiding the exponential dependence in context dimension d. The combined dependence on S and d is now $O(\max\{d^4S, d^2S^3\})$.

The running time of KWIK_LR-Rmax algorithm is $\mathcal{O}(SA\max\{d^2, S\})$ per episode. The detailed discussion for running time analysis can be found in Appendix D.

5. Related work

Transfer in RL with latent contexts The general definition of CMDPs captures the problem of transfer in RL and multi-task RL. See Taylor and Stone (2009) and Lazaric (2011) for surveys of empirical results. Recent papers have also advanced the theoretical understanding of transfer in RL. For instance, Brunskill and Li (2013) and Hallak et al. (2015) analyzed the sample complexity of CMDPs where each MDP is an element of a finite and small set of MDPs, and the MDP label is treated as the *latent* (i.e., unseen) context. Mahmud et al. (2013) consider the problem of transferring the optimal policies of a large set of known MDPs to a new MDP. All of these recent papers assume that the MDP label (i.e., the context) is *not* observed. Hence, their methods have to initially explore in every new MDP to identify its label, which requires the episode length to be substantially longer than the planning horizon. This can be a problematic assumption in our motivating scenarios, where we interact with a patient / user / student for a limited period of time and the data in a single episode (whose length H *is* the planning horizon) is not enough for identifying the underlying MDP. In contrast to prior work, we propose to leverage observable context information to perform more direct transfer from previous MDPs, and our algorithm works with arbitrary episode length H.

RL with side information Our work leverages the available side-information for each MDP, which is inspired by the use of contexts in contextual bandits (Langford and Zhang, 2008; Li et al., 2010). The use of such side information can also be found in RL literature: Ammar et al. (2014) developed a multi-task policy gradient method where the context is used for transferring knowledge between tasks; Killian et al. (2016) used parametric forms of MDPs to develop models for personalized medicine policies for HIV treatment.

RL in metric space For smooth CMDPs (Section 3), we pool observations across similar contexts and reduce the problem to learning policies for finitely many MDPs. An alternative approach is to consider an infinite MDP whose state representation is augmented by the context, and apply PAC-MDP methods for metric state spaces (e.g., C-PACE proposed by Pazis and Parr (2013)). However, doing so might increase the sample and computational complexity unnecessarily, because we no longer leverage the structure that a particular component of the (augmented) state, namely the context, remains the same in an episode. Concretely, the augmenting approach needs to perform planning in the augmented MDP over states and contexts, which makes its computational/storage requirement worse than our solution: we only perform planning in MDPs defined on $\mathcal{S}$, whose computational characteristics have no dependence on the context space. In addition, we allow the context sequence to be chosen in an adversarial manner. This corresponds to adversarially chosen initial states in MDPs, which is usually not handled by PAC-MDP methods.

KWIK learning of linear hypothesis classes Our linear combination setting (Section 4) provides an instance where parametric assumptions can lead to substantially improved PAC bounds. We build upon the KWIK-Rmax learning framework developed in previous work (Li et al., 2008; Szita and Szepesvári, 2011) and use KWIK linear regression as a sub-routine. For the resulting KWIK_LR-Rmax algorithm, its sample complexity bound inherently depends on the KWIK bound for linear regression. It is well known that even for linear hypothesis classes, the KWIK bound is exponential in input dimension in the agnostic case (Szita and Szepesvári, 2011). Therefore, the success of the algorithm relies on the validity of the modelling assumption.

Abbasi-Yadkori and Neu (2014) studied a problem similar to our linear combination setting, and proposed a no-regret algorithm by combining UCRL2 (Jaksch et al., 2010) with confidence set techniques from stochastic linear optimization literature (Filippi et al., 2010). Our work takes an independent and very different approach, and we provide a PAC guarantee which is not directly comparable to regret bound. Still, we observe that our dependence on A is optimal for PAC whereas theirs is not ($\sqrt{A}$ is optimal for bandit regret analysis and they have A); on the other hand, their dependence on T (the number of rounds) is optimal, and our dependence on $1/\epsilon$, its counterpart in PAC analysis, is suboptimal. It is an interesting future direction to combine the algorithmic ideas from both papers to improve the guarantees.

6. Conclusion

We presented a general setting involving the use of side information, or context, for learning near-optimal policies in a large and potentially infinite number of MDPs. Our Cover-Rmax algorithm is a model-based PAC-exploration algorithm for the case where MDPs vary smoothly with respect to the observed side information. Our lower bound construction indicates the necessary exponential dependence of any PAC algorithm on the context dimension in a smooth CMDP. We also consider another instance with a parametric assumption, and using a KWIK linear regression procedure, present the KWIK_LR-Rmax algorithm for efficient exploration in linear combination of MDPs. Our PAC analysis shows a significant improvement with this structural assumption.

The use of context based modelling of multiple tasks has rich application possibilities in personalized recommendations, healthcare treatment policies, and tutoring systems. We believe that our setting can possibly be extended to cover the large space of multi-task RL quite well with finite/infinite number of MDPs, observed/latent contexts, and deterministic/noisy mapping between context and environment.

Acknowledgments

This work was supported in part by a grant from the Open Philanthropy Project to the Center for Human-Compatible AI, and in part by NSF Grant IIS 1319365. Ambuj Tewari acknowledges the support from NSF grant CAREER IIS-1452099 and Sloan Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the views of the sponsors.

References

Yasin Abbasi-Yadkori and Gergely Neu. Online learning in MDPs with side information. *arXiv preprint arXiv:1406.6812*, 2014.

Haitham B Ammar, Eric Eaton, Paul Ruvolo, and Matthew Taylor. Online multi-task learning for policy gradient methods. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1206–1214, 2014.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct): 213–231, 2002.

Emma Brunskill and Lihong Li. Sample complexity of multi-task reinforcement learning. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 122–131. AUAI Press, 2013.

Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279. ACM, 2008.

Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*, pages 586–594, 2010.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual Markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Sham Machandranath Kakade. *On the sample complexity of reinforcement learning.* PhD thesis, University College London, 2003.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Taylor Killian, George Konidaris, and Finale Doshi-Velez. Transfer learning across patient variations with hidden parameter Markov decision processes. *arXiv preprint arXiv:1612.00475*, 2016.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824, 2008.

A. Lazaric. Transfer in reinforcement learning: a framework and a survey. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning: State of the Art*. Springer, 2011.

Lihong Li, Michael L Littman, and Thomas J Walsh. Knows what it knows: a framework for self-aware learning. In *Proceedings of the 25th international conference on Machine learning*, pages 568–575. ACM, 2008.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

MM Mahmud, Majd Hawasly, Benjamin Rosman, and Subramanian Ramamoorthy. Clustering Markov decision processes for continual transfer. *arXiv preprint arXiv:1311.3959*, 2013.

Jason Pazis and Ronald Parr. PAC optimal exploration in continuous space Markov decision processes. In *AAAI*, 2013.

Aleksandrs Slivkins. Contextual bandits with similarity information. *Journal of Machine Learning Research*, 15(1):2533–2568, 2014.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

István Szita and Csaba Szepesvári. Agnostic KWIK learning and efficient approximate reinforcement learning. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 739–772, 2011.

Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

M. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. In preparation, 2017.

Thomas J Walsh, István Szita, Carlos Diuk, and Michael L Littman. Exploring compact reinforcement-learning representations with linear regression. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 591–598. AUAI Press, 2009. A corrected version is available as Technical Report DCS-tr-660, Department of Computer Science, Rutgers University, December, 2009.

Appendix A. Proofs from Section 3

A.1. Proof of Lemma 5

We adapt the analysis in Kakade (2003) for the episodic case which results in the removal of a factor of H, since complete episodes are counted as mistakes and we do not count every sub-optimal action in an episode. The detailed analysis is reproduced here for completeness. For completing the proof of Lemma 5, firstly, we will look at a version of simulation lemma from Kearns and Singh (2002). Also, for the complete analysis we will assume that the rewards lie between 0 and 1.

Definition 9 (Induced MDP) *Let M be an MDP with $K \subseteq \mathcal{S}$ being a subset of states. Given, such a set K, we define an induced MDP M_K in the following manner. For each*

$s \in K$, define the values

$$\begin{aligned} p_{M_K}(s'|s,a) &= p_M(s'|s,a) \\ r_{M_K}(s,a) &= r_M(s,a) \end{aligned}$$

For all $s \notin K$, define $p_{M_K}(s'|s,a) = \mathbb{I}\{s' = s\}$ and $r_{M_K}(s,a) = 1$.

Lemma 10 (Simulation lemma for episodic MDPs) *Let M and M' be two MDPs with the same state-action space. If the transition dynamics and the reward functions of the two MDPs are such that*

$$\|p_M(\cdot|s,a) - p_{M'}(\cdot|s,a)\|_1 \leq \epsilon_1 \ \forall s \in \mathcal{S}, a \in \mathcal{A}$$

$$|r_M(s,a) - r_{M'}(s,a)| \leq \epsilon_2 \ \forall s \in \mathcal{S}, a \in \mathcal{A}$$

then, for every (non-stationary) policy π the two MDPs satisfy this property:

$$|V_M^\pi - V_{M'}^\pi| \leq \epsilon_2 + H\epsilon_1$$

Proof Consider $\mathcal{T}_h$ to be the set of all trajectories of length h and let $P_M^\pi(\tau)$ denote the probability of observing trajectory τ in M with the behaviour policy π. Further, let $U_M(\tau)$ the expected average reward obtained for trajectory τ in MDP M.

$$\begin{aligned} |V_M^\pi - V_{M'}^\pi| &= |\sum_{\tau \in \mathcal{T}_H} [P_M^\pi(\tau) U_M(\tau) - P_{M'}^\pi(\tau) U_{M'}(\tau)]| \\ &\leq |\sum_{\tau \in \mathcal{T}_H} [P_M^\pi(\tau) U_M(\tau) - P_M^\pi(\tau) U_{M'}(\tau) + P_M^\pi(\tau) U_{M'}(\tau) - P_{M'}^\pi(\tau) U_{M'}(\tau)]| \\ &\leq |\sum_{\tau \in \mathcal{T}_H} [P_M^\pi(\tau)(U_M(\tau) - U_{M'}(\tau))]| + |\sum_{\tau \in \mathcal{T}_H} [U_{M'}(\tau)(P_M^\pi(\tau) - P_{M'}^\pi(\tau))]| \\ &\leq |\sum_{\tau \in \mathcal{T}_H} P_M^\pi(\tau)|\epsilon_2 + |\sum_{\tau \in \mathcal{T}_H} [P_M^\pi(\tau) - P_{M'}^\pi(\tau)]| \\ &\leq \epsilon_2 + |\sum_{\tau \in \mathcal{T}_H} [P_M^\pi(\tau) - P_{M'}^\pi(\tau)]| \end{aligned}$$

The bound for the second term follows from the proof of Lemma 8.5.4 in Kakade (2003). Combining the two expressions, we get the desired result. ■

Lemma 11 (Induced inequalities) *Let M be an MDP with K being the set of known states. Let M_K be the induced MDP as defined in 9 with respect to K and M. We will show that for any (non-stationary) policy π, all states $s \in \mathcal{S}$,*

$$V_{M_K}^\pi(s) \geq V_M^\pi(s)$$

and

$$V_M^\pi(s) \geq V_{M_K}^\pi(s) - P_M^\pi[\textit{Escape to an unknown state}|s_0 = s]$$

where $V_M^\pi(s)$ denotes the value of policy π in MDP M when starting from state s.

Proof See Lemma 8.4.4 from Kakade (2003). ■

Corollary 12 (Implicit Explore and Exploit) *Let M be an MDP with K as the set of known states and M_K be the induced MDP. If $\pi^*_{M_K}$ and π^*_M be the optimal policies for M_K and M respectively, we have for all states s:*

$$V_M^{\pi^*_{M_K}}(s) \geq V^*_M(s) - P^\pi_M[\textit{Escape to an unknown state}|s_0 = s]$$

Proof Follows from Lemma 8.4.5 from Kakade (2003). ■

Proof of Lemma 5 Let π^*_M be the optimal policy for M. Also, using the assumption about m, we have an $\epsilon/2$-approximation of M_K as the MDP $\hat{M}_K$. Rmax computes the optimal policy for $\hat{M}_K$ which is denoted by $\hat{\pi}$. Then, by Lemma 10,

$$\begin{aligned} V^{\hat{\pi}}_{M_K}(s) &\geq V^{\hat{\pi}}_{\hat{M}_K}(s) - \epsilon/2 \\ &\geq V^{\pi^*_M}_{\hat{M}_K}(s) - \epsilon/2 \\ &\geq V^{\pi^*_M}_{M_K}(s) - \epsilon \end{aligned}$$

Combining this with Lemma 11, we get

$$\begin{aligned} V^{\hat{\pi}}_M(s) &\geq V^{\hat{\pi}}_{M_K}(s) - \mathrm{P}^\pi_M[\text{Escape to an unknown state}|s_0 = s] \\ &\geq V^{\pi^*_M}_{M_K}(s) - \epsilon - \mathrm{P}^\pi_M[\text{Escape to an unknown state}|s_0 = s] \\ &\geq V^*_M(s) - \epsilon - \mathrm{P}^\pi_M[\text{Escape to an unknown state}|s_0 = s] \end{aligned}$$

If this escape probability is less than ϵ, then the desired relation is true. Therefore, we need to bound the number of episodes where this expected number is greater than ϵ. Note that, due to balanced wandering, there can be at most mSA visits to unknown states for the Rmax algorithm. In the execution, the agent may encounter an extra $H-1$ visits as the estimates are updated only after the termination of an episode.

Whenever this quantity is more than ϵ, the expected number of exploration steps in mSA/ϵ such episodes is at least mSA. By the Hoeffding's inequality, for N episodes, with probability, at least $1-\delta$, the number of successful exploration steps is greater than

$$N\epsilon - \sqrt{\frac{N}{2}\ln\frac{1}{\delta}}$$

Therefore, if $N = \mathcal{O}(\frac{mSA}{\epsilon}\ln\frac{1}{\delta})$, with probability at least $1-\delta$, the total number of visits to an unknown state is more than mSA. Using the upper bound on such visits, we conclude that these many episodes suffice. ■

A.2. Proof of Theorem 4

We now need to compute the required resolution of the cover and the number of transitions m which will guarantee the approximation for the value functions as required in the previous lemma. The following is a key result:

Lemma 13 (Cover approximation) *For a given CMDP and a finite cover, i.e.,* $\mathcal{C} = \cup_{i=1}^{N(\mathcal{C},r)} \mathcal{B}_i$ *such that* $\forall i, \forall c_1, c_2 \in \mathcal{B}_i$ *:*

$$\|p^{c_1}(\cdot|s,a) - p^{c_2}(\cdot|s,a)\|_1 \leq \epsilon/8H$$

and

$$|r^{c_1}(s,a) - r^{c_2}(s,a)| \leq \epsilon/8$$

if the agent visit every state-action pair $m = \frac{128(S\ln 2 + \ln\frac{SA}{\delta})H^2}{\epsilon^2}$ *times in a ball* $\mathcal{B}_i$ *summing observations over all* $c \in \mathcal{B}_i$*, then, for any policy* π *and with probability at least* $1-2\delta$*, the approximate MDP* $\hat{M}_i$ *corresponding to* $\mathcal{B}_i$ *computed using empirical averages will satisfy*

$$|V^\pi_{M_c} - V^\pi_{\hat{M}_i}| \leq \epsilon/2$$

for all $c \in \mathcal{B}_i$*.*

Proof

With each visit to a state action pair (s,a), a transition to some $s' \in \mathcal{S}$ is observed for context $c_t \in \mathcal{B}_i$ in t_{th} visit with probability $p_{c_t}(s,a)$. Let us encode this by an S-dimensional vector I_t with 0 at all indices except s'. After observing m such transitions, the next state distribution for any $c \in \mathcal{B}_i$ is esitmated as $p_{\hat{M}_i}(\cdot|s,a) = \frac{1}{m}\sum_{t=1}^m I_t$. Now for all $c \in \mathcal{B}_i$,

$$\|p_{\hat{M}_i}(\cdot|s,a) - p^c(\cdot|s,a)\|_1 \leq \|p_{\hat{M}_i}(\cdot|s,a) - \frac{1}{m}\sum_{t=1}^m p^{c_t}(\cdot|s,a)\|_1 + \epsilon/8H$$

For bounding the first term, we use the Hoeffding's bound:

$$\begin{aligned}
P\Big[\|p_{\hat{M}_i}(\cdot|s,a) - \frac{1}{m}\sum_{t=1}^m p^{c_t}(\cdot|s,a)\|_1 \geq \epsilon\Big] &= P\Big[\max_{s'\in A\subseteq\mathcal{S}}(p_{\hat{M}_i}(s'\in A|s,a) - \frac{1}{m}\sum_{t=1}^m p^{c_t}(s'\in A|s,a)) \geq \epsilon/2\Big] \\
&\leq \sum_{s'\in A\subseteq\mathcal{S}} P\Big[(p_{\hat{M}_i}(s'\in A|s,a) - \frac{1}{m}\sum_{t=1}^m p^{c_t}(s'\in A|s,a)) \geq \epsilon/2\Big] \\
&\leq (2^S - 2)\exp(-m\epsilon^2/2)
\end{aligned}$$

Therefore, with probability at least $1-\delta/2$, for all $s \in \mathcal{S}, a \in \mathcal{A}$, we have:

$$\|p_{\hat{M}_i}(\cdot|s,a) - p^c(\cdot|s,a)\|_1 \leq \sqrt{\frac{2(S\ln 2 + \ln 2SA/\delta)}{m}} + \epsilon/8H$$

If $m = \frac{128(S\ln 2 + \ln\frac{SA}{\delta})H^2}{\epsilon^2}$, the error becomes $\epsilon/4H$. One can easily verify using similar arguments that, the error in rewards for any context $c \in \mathcal{B}_i$ is less than $\epsilon/4$.

By using the simulation lemma 10, we get the desired result. ■

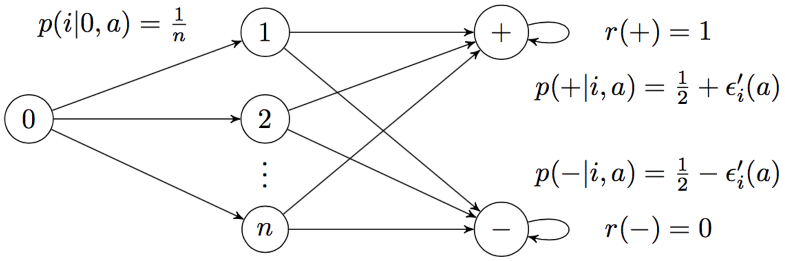
$p(i|0,a) = \frac{1}{n}$
0
1
2
⋮
n
+
−
$r(+) = 1$
$p(+|i,a) = \frac{1}{2} + \epsilon'_i(a)$
$p(-|i,a) = \frac{1}{2} - \epsilon'_i(a)$
$r(-) = 0$

Figure 1: Hard instances for episodic MDP (Dann and Brunskill, 2015). The initial state 0 moves to a uniform distribution over states 1 to n regardless of the action, and states $+/-$ are absorbing with 1 and 0 rewards respectively. States 0 to n have 0 reward for all actions. Each state $i \in [n]$ essentially acts as a hard bandit instance, whose A actions move to $+$ and $-$ randomly. Action a_0 satisfies $p(+|i, a_0) = \frac{1}{2} + \frac{\epsilon'}{2}$ and there is at most one other action a_i with $p(+|i, a_i) = \frac{1}{2} + \epsilon'$. Any other action a_j satisfies $p(+|i, a_j) = \frac{1}{2}$.

Appendix B. Lower bound analysis

The key construction in the proof is to embed multiple MDP learning problems in a CMDP, such that the agent has to learn the optimal policy in each MDP separately. We start with the lower bound for learning in episodic MDPs. See Figure 1 and its caption for details. The construction is due to Dann and Brunskill (2015) and we adapt their lower bound statement to our setting in Theorem 14.

Theorem 14 (Lower bound for episodic MDP (Dann and Brunskill, 2015)) *There exists constants δ_0, ϵ_0, such that for every $\delta \in (0, \delta_0)$ and $\epsilon \in (0, \epsilon_0)$, any algorithm that satisfies a PAC guarantee for (ϵ, δ) and computes a sequence of deterministic policies, there is a hard instance M_{hard} so that $\mathbb{E}[B] = \Omega\left(\frac{SA}{\epsilon^2}\right)$, where B is the number of sub-optimal episodes. The constants can be chosen as $\delta_0 = \frac{e^{-4}}{80}$, $\epsilon_0 = \frac{H-2}{640He^4}$.*[4]

Now we discuss how to populate the context space with these hard MDPs. Note in Figure 1 that, the agent does not know which action is the most rewarding (a_i), and the adversary can choose i to be any element of $[A]$ (which is essentially choosing an instance from a family of MDPs). In our scenario, we would like to allow the adversary to choose the MDP *independently* for each individual packing point to yield a lower bound linear in the packing number. However, this is not always possible due to the smoothness assumption, as committing to an MDP at one point may restrict the adversary's choices at another point.

To deal with this difficulty, we note that any pair of hard MDPs differ from each other by $O(\epsilon')$ in transition distributions. Therefore, we construct a packing of $\mathcal{C}$ with radius $r = 8\epsilon'$, defined as a set of points Z such that any two points in Z are at least r away from

[4] The lower bound here differs from that in the original paper by H^2, because our value is normalized (see Eq.(1)), whereas they allow the magnitude of value to grow with H.

each other. The maximum size of such Z is known as the *packing number*:

$$\mathcal{D}(\mathcal{C}, r) = \max\{|Z| : Z \text{ is an } r\text{-packing of } \mathcal{C}\},$$

which is related to the covering number as $N(\mathcal{C}, r) \leq \mathcal{D}(\mathcal{C}, r)$. The radius r is chosen to be $O(\epsilon')$ so that arbitrary choices of hard MDP instances at different packing points always satisfy the smoothness assumption (recall that $L_p = 1$). Once we fix the MDPs for all $c \in Z$, the MDP for $c \in \mathcal{C} \setminus Z$ is specified as follows: for state i and action a,

$$p_c(+|i,a) = \max_{c' \in Z} \max(1/2, p_{c'}(+|i,a) - \phi(c,c')/2).$$

Essentially, as we move away from a packing point, the transition to $+/-$ become more uniform. We can show that:

Claim 15 *The CMDP defined above is satisfies Definition 3 with constant $L_p = 1$.*[5]

Proof We need to prove that the defined contextual MDP, satisfies the constraints in Definition 3. Let us assume that the smoothness assumption is violated for a context pair (c_1, c_2). The smoothness constraints for rewards are satisfied trivially for any value of L_r as they are constant. This implies that there exists state $i \in [n]$ and action a such that

$$\begin{aligned} \|p_{c_1}(\cdot|i,a) - p_{c_2}(\cdot|i,a)\|_1 &> \phi(c_1, c_2) \\ \Rightarrow 2|p_{c_1}(+|i,a) - p_{c_2}(+|i,a)| &> \phi(c_1, c_2) \end{aligned}$$

We know that, $p_c(+|i,a) \in [1/2, 1/2 + \epsilon']$, which shows that $\phi(c_1, c_2) < 2\epsilon'$. Without loss of generality, assume $p_{c_1}(+|i,a) > p_{c_2}(+|i,a)$ which also leads to

$$\begin{aligned} p_{c_1}(+|i,a) &> 1/2 \\ \Rightarrow \exists c_0 \in Z \text{ such that } \phi(c_1, c_0) &< 2\epsilon' \end{aligned}$$

By triangle inequality, we have

$$\phi(c_2, c_0) < 4\epsilon'$$

Now, $\forall c_0' \in Z$, such that $\phi(c_0', c_0) \geq 8\epsilon'$, by triangle inequality, we have

$$\begin{aligned} \phi(c_0', c_1) &> 6\epsilon' \\ \phi(c_0', c_2) &> 4\epsilon' \end{aligned}$$

This simplifies the definition of $p_c(+|i,a)$ for $c = c_1, c_2$ to

$$p_c(+|i,a) = \max(1/2, p_{c_0}(+|i,a) - \phi(c_0, c)/2)$$

Now,

$$\begin{aligned} |p_{c_1}(+|i,a) - p_{c_2}(+|i,a)| &= p_{c_1}(+|i,a) - p_{c_2}(+|i,a) \\ &= p_{c_0}(+|i,a) - \phi(c_0, c_1)/2 - \max(1/2, p_{c_0}(+|i,a) - \phi(c_0, c_2)/2) \\ &\leq p_{c_0}(+|i,a) - \phi(c_0, c_1)/2 - (p_{c_0}(+|i,a) - \phi(c_0, c_2)/2) \\ &= \tfrac{1}{2}(\phi(c_0, c_2) - \phi(c_0, c_1)) \leq \phi(c_1, c_2)/2 \end{aligned}$$

[5] The reward function does not vary with context, hence, reward smoothness is satisfied for all $L_r \geq 0$.

which leads to a contradiction. ■

We choose the context sequence given as input to be repetitions of an arbitrary permutation of Z. Therefore, our construction populates a set of packing points in the context space with hard MDPs. We claim that these instances are independent of each other from the algorithm's perspective. To formalize this statement, let Z be the $8\epsilon'$-packing as before. The adversary makes the choices of the instances at each context $c \in Z$, as follows: Select an MDP from the family of hard instances described in Figure 1 where the optimal action from each state in $\{1, \dots, n\}$ is chosen randomly and independently from the other assignments. The parameter ϵ' deciding the difference in optimality of actions in Figure 1 is taken as $\frac{160H\epsilon\epsilon^4}{(H-2)}$. The expression is obtained by using the construction of Theorem 14.

We denote these instances by the set $\mathcal{I}$ and an individual instance by $\mathcal{I}_z$. Let $z = \{z_1, z_2, \dots, z_{|Z|}\}$ be the random vector denoting the optimal actions chosen for the MDPs corresponding to the packing points. By construction, we have a uniform distribution $\Gamma \equiv \Gamma_1 \times \Gamma_2 \times \dots \Gamma_{|Z|}$ over these possible set of instances $\mathcal{I}_z$. From Claim 15, any assignment of optimal actions to these packing points would define a valid smooth contextual MDP. Further, the independent choice of the optimal actions makes MDPs at each packing point at least as difficult as learning a single MDP. Formally, let the sequence of transitions and rewards observed by the learning agent for all packing points be $T \equiv \{\tau_1, \tau_2, \dots, \tau_{|Z|}\}$. Due to the independence between individual instances, we can see that:

$$P_\Gamma[\,\tau_1 | \tau_2, \tau_3, \dots, \tau_{|Z|}\,] \equiv P_{\Gamma_1}[\,\tau_1\,]$$

where $P_\Gamma[\tau_i]$ denotes the distribution of trajectories τ_i. Thus, we cannot deduce anything about the optimal actions for one point by observing trajectories from MDP instances at other packing points. With respect to this distribution, learning in the contextual MDP is equivalent to or worse than simulating a PAC algorithm for a single MDP at each of these packing points. For any given contextual MDP algorithm `Alg`, we have:

$$\mathbb{E}_\Gamma[B_i] = \mathbb{E}_{\Gamma_{-i}}[\mathbb{E}_{\Gamma_i,\texttt{Alg}}[B_i | T_{-i}]] \geq \mathbb{E}_{\Gamma_{-i}}[\mathbb{E}_{\Gamma_1,\texttt{Alg*}}[B_i]]$$

where `Alg*` is an optimal single MDP learning algorithm. The expectation is with respect to the distribution over the instances $\mathcal{I}_z$ and the algorithm's randomness. From Theorem 14, we can lower bound the expectation on the right hand side of the inequality by $\Omega\left(\frac{SA}{\epsilon^2}\right)$.

The total number of mistakes is lower bounded as:

$$\mathbb{E}_\Gamma[\sum_{i=1}^{|Z|} B_i] = \sum_{i=1}^{|Z|} \mathbb{E}_\Gamma[B_i] \geq \Omega\left(\frac{|Z|SA}{\epsilon^2}\right)$$

Setting $|Z| = \mathcal{D}(\mathcal{C}, \epsilon_1) \leq \mathcal{N}(\mathcal{C}, \epsilon_1)$ gives the stated lower bound with $\epsilon_1 = 8\epsilon'$.

Appendix C. Proof of the Theorem 7

In this section, we present the proof of our KWIK bound for learning transition probabilities. Our proof uses a reduction technique that reduces the vector-valued label setting to the scalar setting, and combines the KWIK bound for scalar labels given by Walsh et al. (2009).

Proof Fix a state action pair (s, a). Consider a sequence of contexts $c_1, c_2, ...$ for which the transitions were observed for pair (s, a). Given a new context c, we want to estimate:

$$p^c(\cdot|s, a) = c^\top P(s, a)$$

In our KWIK_LR algorithm, this is estimated as:

$$\hat{p}^c(\cdot|s, a) = c^\top Q(s, a)W(s, a)$$

where $Q(s, a)$ and $W(s, a)$ are as described in Section 4.1.

We wish to bound the ℓ_1 error between $\hat{p}^c(\cdot|s, a)$ and $p^c(\cdot|s, a)$ for all c for which a prediction is made. We know that

$$\|p^c(\cdot|s, a) - \hat{p}^c(\cdot|s, a)\|_1 = \sup_{f \in \{-1,1\}^S} (p^c(\cdot|s, a) - \hat{p}^c(\cdot|s, a))f. \tag{6}$$

This representation of ℓ_1-norm can be used to prove a tighter KWIK bound for learning transition probabilities. For every fixed $f \in \{-1, 1\}^S$, we formulate a new linear regression problem with feature-label pair:

$$(c, yf).$$

Recall that $y = (\{\mathbb{I}[s_{\text{next}} = s']\}_{\forall s' \in \mathcal{S}})^\top$ is the vector label of real interest, and f projects y to a scalar value. Algorithm 3 can be viewed as implicitly running this regression thanks to linearity: since Q only depends on input contexts and W is linear in y, $\hat{p}^c(\cdot|s, a)f$ is simply equal to the linear regression prediction for the problem (c, yf). As a result, the KWIK bound for the problem (c, yf) (which we establish below) automatically applies as a property of $\hat{p}^c(\cdot|s, a)f$. Taking union bound over all $f \in \{-1, 1\}^S$ yields the desired ℓ_1 error guarantee for $\hat{p}^c(\cdot|s, a)$ thanks to Equation 6.

Now we establish the KWIK guarantee for the new regression problem. The groundtruth (expected) label is

$$p^c(\cdot|s, a)f = c^\top (P(s, a)f) := c^\top \theta^f. \tag{7}$$

The noise in the label is then

$$\eta^f := (y - p^c(\cdot|s, a))f. \tag{8}$$

This noise has zero-mean and constant magnitude: $|\eta^f| \leq \|y - p^c(\cdot|s, a)\|_1 \|f\|_\infty \leq 2$.

With the above conditions, we can invoke the KWIK bound for scalar linear regression from Walsh et al. (2009):

Theorem 16 (KWIK bound for linear regression (Walsh et al., 2009)) *Suppose the observation noise in a noisy linear regression problem has zero-mean and its absolute value is bounded by β. Let M be an upper bound on the ℓ_2 norm of the true linear coefficients. For any $\delta' > 0$ and $\epsilon > 0$, if the KWIK linear regression algorithm is executed with $\alpha_0 = \min\{b_1 \frac{\epsilon^2}{dM}, b_2 \frac{\epsilon^2}{M \log(d/\delta')}, \frac{\epsilon}{2M}\}$, with suitable constants b_1 and b_2, then the number of $\perp$'s will be $O(M^2 \max\{\frac{d^3}{\epsilon^4}, \frac{d \log^2(d/\delta')}{\epsilon^4}\})$, and with probability at least $1 - \delta'$, for each sample x_t for which a prediction is made, the prediction is ϵ-accurate.*

For our purpose, $\beta = 2$ as $|\eta^f| \leq 2$ and $M = \sqrt{d}$ as $\|\theta^f\|_2 = \|P(s,a)f\|_2 \leq \sqrt{d}$.

Now set $\delta' = \frac{\delta}{2^S}$ in Theorem 16. In the KWIK linear regression algorithm, the *known* status for a context c is checked in the same manner as done in Line 5 in Algorithm 3. Therefore

$$\begin{aligned}
&Pr\Big[\|p^c(\cdot|s,a) - \hat{p}^c(\cdot|s,a)\|_1 \geq \epsilon\Big] \\
&= Pr\Big[\sup_{f\in\{-1,1\}^S} (p^c(\cdot|s,a) - \hat{p}^c(\cdot|s,a))f \geq \epsilon\Big] && \text{(Equation 6)} \\
&\leq \sum_{f\in\{-1,1\}^S} Pr\Big[(p^c(\cdot|s,a) - \hat{p}^c(\cdot|s,a))f \geq \epsilon\Big] && \text{(union bound)} \\
&= \sum_{f\in\{-1,1\}^S} Pr\Big[c^\top\theta^f - c^\top Q(s,a)(W(s,a)f) \geq \epsilon\Big] && \text{(regression w.r.t. } f \text{ implicitly run)} \\
&\leq \sum_{f\in\{-1,1\}^S} \delta/2^S = \delta.
\end{aligned}$$

Substituting the values of M and δ' in Theorem 16, we get:

$$\alpha_S = \min\{b_1 \tfrac{\epsilon^2}{d^{3/2}}, b_2 \tfrac{\epsilon^2}{\sqrt{d}\log(d/2^S\delta')}, \tfrac{\epsilon}{2\sqrt{d}}\}$$

and number of $\perp$'s is bounded as

$$O(\max\{\tfrac{d^4}{\epsilon^4}, \tfrac{d^2S^2\log^2(d/\delta')}{\epsilon^4}\}).$$

■

Appendix D. Computational complexity of KWIK_LR-Rmax

The KWIK_LR-Rmax algorithm maintains SA-many matrices of size $\mathcal{O}(d^2)$. At the start of each episode, the algorithm requires a vector-matrix multiplication for computing the *knownness* of each state-action pair, which takes a total of $\mathcal{O}(d^2SA)$ operations (Line 5). For known state-action pairs, computing $\hat{p}^c(\cdot|s,a)$ requires a projection step along with vector-matrix multiplications and takes $\mathcal{O}(d^2SA + S^2A)$ operations (Line 6)[6]. At the end of every episode, it performs rank-1 updates to the matrices Q and W for every *unknown* state-action pair, each taking $\mathcal{O}(d^2)$ operations (Line 12 and Line 14). We, therefore, observe that the KWIK_LR-Rmax algorithm takes $\mathcal{O}(SA\max\{d^2, S\})$ running time per episode.

[6] The projection onto Δ_{S-1} takes $\mathcal{O}(S)$ operations per state-action pair.