arXiv:1901.09018v1 [cs.LG] 25 Jan 2019

Provably efficient RL with Rich Observations via Latent State Decoding

Simon S. Du [1] **Akshay Krishnamurthy** [2] **Nan Jiang** [3] **Alekh Agarwal** [4] **Miroslav Dudík** [2] **John Langford** [2]

Abstract

We study the exploration problem in episodic MDPs with rich observations generated from a small number of latent states. Under certain identifiability assumptions, we demonstrate how to estimate a mapping from the observations to latent states inductively through a sequence of regression and clustering steps—where previously decoded latent states provide labels for later regression problems—and use it to construct good exploration policies. We provide finite-sample guarantees on the quality of the learned state decoding function and exploration policies, and complement our theory with an empirical evaluation on a class of hard exploration problems. Our method exponentially improves over Q-learning with naïve exploration, even when Q-learning has cheating access to latent states.

1. Introduction

We study reinforcement learning (RL) in episodic environments with rich observations, such as images and texts. While many modern empirical RL algorithms are designed to handle such settings (see, e.g., Mnih et al., 2015), relatively few works focus on the question of strategic exploration in this literature (Ostrovski et al., 2017; Osband et al., 2016) and the sample efficiency of these techniques is not theoretically understood.

From a theoretical perspective, strategic exploration algorithms for provably sample-efficient RL have long existed in the classical tabular setting (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002). However, these methods are difficult to adapt to rich observation spaces, because they all require a number of interactions polynomial in the number of *observed states*, and, without additional structural assumptions, such a dependency is unavoidable (see, e.g., Jaksch et al., 2010; Lattimore & Hutter, 2012). Consequently, treating the observations directly as unique states makes this class of methods unsuitable for most settings of practical interest.

In order to avoid the dependency on the observation space, one must exploit some inherent structure in the problem. The recent line of work on contextual decision processes (Krishnamurthy et al., 2016; Jiang et al., 2017; Dann et al., 2018) identified certain low-rank structures that enable exploration algorithms with sample complexity polynomial in the rank parameter. Such low-rank structure is crucial to circumventing information-theoretic hardness, and is typically found in problems where complex observations are emitted from a small number of *latent states*. Unlike tabular approaches, which require the number of states to be small and observed, these works are able to handle settings where the observation spaces are uncountably large or continuous and the underlying states never observed during learning. They achieve this by exploiting the low-rank structure implicitly, operating only in the observation space. The resulting algorithms are sample-efficient, but either provably computationally intractable, or practically quite cumbersome even under strong assumptions (Dann et al., 2018).

In this work, we take an alternative route: we recover the latent-state structure explicitly by learning a *decoding function* (from a large set of candidates) that maps a rich observation to the corresponding latent state; note that if such a function is learned perfectly, the rich-observation problem is reduced to a tabular problem where exploration is tractable. We show that our algorithms are:

Provably sample-efficient: Under certain identifiability assumptions, we recover a mapping from the observations to underlying latent states as well as a good exploration policy using a number of samples which is polynomial in the number of latent states, horizon and the complexity of the decoding function class with no explicit dependence on the observation space size. Thus we significantly generalize beyond the works of Dann et al. (2018) who require deterministic dynamics and Azizzadenesheli et al. (2016a) whose guarantees scale with the observation space size.

Computationally practical: Unlike many prior works in this vein, our algorithm is easy to implement and substantially outperforms naïve exploration in experiments, even when the baselines have cheating access to the latent states.

In the process, we introduce a formalism called *block Mar-*

[1]Carnegie Mellon University [2]Microsoft Research, New York [3]University of Illinois at Urbana-Champaign [4]Microsoft Research, Redmond.

kov decision process (also implicit in some prior works), and a new solution concept for exploration called ϵ–policy cover.

The main challenge in learning the decoding function is that the hidden states are never directly observed. Our key novelty is the use of a backward conditional probability vector (Equation 1) as a representation for latent state, and learning the decoding function via conditional probability estimation, which can be solved using least squares regression. While learning a low-dimensional representations of rich observations has been explored in recent empirical works (e.g., Silver et al., 2017; Oh et al., 2017; Pathak et al., 2017), our work provides a precise mathematical characterization of the structures needed for such approaches to succeed and comes with rigorous sample-complexity guarantees.

2. Setting and Task Definition

We begin by introducing some basic notation. We write $[h]$ to denote the set $\{1, \dots, h\}$. For any finite set S, we write $U(S)$ to denote the uniform distribution over S. We write $\triangle_d$ for the simplex in $\mathbb{R}^d$. Finally, we write $\|\cdot\|$ and $\|\cdot\|_1$, respectively, for the Euclidean and the ℓ_1 norms of a vector.

2.1. Block Markov Decision Process

In this paper we introduce and analyze a *block Markov decision process* or *BMDP*. It refers to an environment described by a finite, but unobservable *latent state space* $\mathcal{S}$, a finite *action space* $\mathcal{A}$, with $|\mathcal{A}| = K$, and a possibly infinite, but observable *context space* $\mathcal{X}$. The dynamics of a BMDP is described by the *initial state* $s_1 \in \mathcal{S}$ and two conditional probability functions: the *state-transition function* p and *context-emission function* q, defining conditional probabilities $p(s' \mid s, a)$ and $q(x \mid s)$ for all $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$, $x \in \mathcal{X}$.[1]

The model may further include a distribution of reward conditioned on context and action. However, rewards do not play a role in the central task of the paper, which is the exploration of all latent states. Therefore, we omit rewards from our formalism, but we discuss in a few places how our techniques apply in the presence of rewards (for a thorough discussion see Appendix B).

We consider episodic learning tasks with a finite horizon H. In each episode, the environment starts in the state s_1. In the step $h \in [H]$ of an episode, the environment generates a context $x_h \sim q(\cdot|s_h)$, the agent observes the context x_h (but not the state s_h), takes an action a_h, and the environment transitions to a new state $s_{h+1} \sim p(\cdot|s_h, a_h)$. The sequence $(s_1, x_1, a_1, \dots, s_H, x_H, a_H, s_{H+1}, x_{H+1})$ generated in an episode is called a *trajectory*. We emphasize that a learning agent does not observe components s_h from the trajectory.

[1] For continuous context spaces, $q(\cdot \mid s)$ describes a density function relative to a suitable measure (e.g., Lebesgue measure).

So far, our description resembles that of a partially observable Markov decision process (POMDP). To finish the definition of BMDP, and distinguish it from a POMDP, we make the following assumption:

Assumption 2.1 (Block structure)**.** *Each context x uniquely determines its generating state s. That is, the context space $\mathcal{X}$ can be partitioned into disjoint blocks $\mathcal{X}_s$, each containing the support of the conditional distribution $q(\cdot \mid s)$.*

The sets $\mathcal{X}_s$ are unique up to the sets of measure zero under $q(\cdot \mid s)$. In the paper, we say "*for all $x \in \mathcal{X}_s$*" to mean "*for all $x \in \mathcal{X}_s$ up to a set of measure zero under $q(\cdot \mid s)$.*"

The block structure implies the existence of a *perfect decoding function* $f^* : \mathcal{X} \to \mathcal{S}$, which maps contexts into their generating states. This means that a BMDP is indeed an MDP with the transition operator $P(x' \mid x, a) = q\big(x' \mid f^*(x')\big)p\big(f^*(x') \bigm| f^*(x), a\big)$. Hence the contexts x observed by the agent form valid Markovian states, but the size of $\mathcal{X}$ is too large, so only learning the MDP parameters in the smaller, latent space $\mathcal{S}$ is tractable.

We note that Assumption 2.1 or similar MDP-structures have been previously studied by Krishnamurthy et al. (2016) and Dann et al. (2018). It can naturally model visual grid-world-like environments often studied in empirical RL (e.g. Johnson et al., 2016), as well as noisy observations of the latent state due to imperfect sensors.

To streamline our analysis, we make a standard assumption for episodic settings. We assume that $\mathcal{S}$ can be partitioned into disjoints sets $\mathcal{S}_h$, $h \in [H+1]$, such that $p(\cdot \mid s, a)$ is supported on $\mathcal{S}_{h+1}$ whenever $s \in \mathcal{S}_h$. We refer to h as the *level* and assume that it is observable as part of the context, so the context space is also partitioned into sets $\mathcal{X}_h$. We use notation $\mathcal{S}_{[h]} = \cup_{\ell \in [h]} \mathcal{S}_\ell$ for the set of states up to level h, and similarly define $\mathcal{X}_{[h]} = \cup_{\ell \in [h]} \mathcal{X}_\ell$.

We assume that $|\mathcal{S}_h| \le M$. We seek learning algorithms that scale polynomially in parameters M, K and H, but do not explicitly depend on $|\mathcal{X}|$, which might be infinite.

2.2. Solution Concept: Cover of Exploratory Policies

In this paper, we focus on the problem of exploration. Specifically, for each state $s \in \mathcal{S}$, we seek an agent strategy for reaching that state s. We formalize an agent strategy as an *h-step policy*, which is a map $\pi : \mathcal{X}_{[h]} \to \mathcal{A}$ specifying which action to take in each context up to step h. When executing an h-step policy π with $h < H$, an agent acts according to π for h steps and then arbitrarily until the end of the episode (e.g., according to a specific default policy).

For an h-step policy π, we write $\mathbb{P}^\pi$ to denote the probability distribution over h-step trajectories induced by π. We write $\mathbb{P}^\pi(\mathcal{E})$ for the probability of an event $\mathcal{E}$. For example, $\mathbb{P}^\pi(s)$ is the probability of reaching the state s when executing π.

We also consider randomized strategies, which we formalize as *policy mixtures*. An h-step *policy mixture* η is a distribution over h-step policies. When executing η, an agent randomly draws a policy $\pi \sim \eta$ at the beginning of the episode, and then follows π throughout the episode. The induced distribution over h-step trajectories is denoted $\mathbb{P}^{\eta}$.

Our algorithms create specific policies and policy mixtures via concatenation. Specifically, given an h-step policy π, we write $\pi \odot a$ for the $(h+1)$-step policy that executes π for h steps and chooses action a in step $h+1$. Similarly, if η is a policy mixture and ν a distribution over $\mathcal{A}$, we write $\eta \odot \nu$ for the policy mixture equivalent to first sampling and following a policy according to η and then independently sampling and following an action according to ν.

We finally introduce two key concepts related to exploration: *maximum reaching probability* and *policy cover*.

Definition 2.1 (Maximum reaching probability.)**.** *For any $s \in \mathcal{S}$, its* maximum reaching probability $\mu(s)$ *is*

$$\mu(s) := \max_{\pi} \mathbb{P}^{\pi}(s),$$

where the maximum is taken over all maps $\mathcal{X}_{[H]} \to \mathcal{A}$. The policy attaining the maximum for a given s is denoted π^*_s.[2]

Without loss of generality, we assume that all the states are reachable, i.e., $\mu(s) > 0$ for all s. We write $\mu_{\min} = \min_{s \in \mathcal{S}} \mu(s)$ for the $\mu(s)$ value of the hardest-to-reach state. Since $\mathcal{S}$ is finite and all states are reachable, $\mu_{\min} > 0$.

Given maximum reaching probabilities, we formalize the task of finding policies that reach states s as the task of finding an *ϵ–policy cover* in the following sense:

Definition 2.2 (Policy cover of the state space)**.** *We say that a set of policies Π_h is an* ϵ–policy cover *of $\mathcal{S}_h$ if for all $s \in \mathcal{S}_h$ there exists an $(h-1)$-step policy $\pi \in \Pi_h$ such that $\mathbb{P}^{\pi}(s) \geq \mu(s) - \epsilon$. A set of policies Π is an* ϵ–policy cover *of $\mathcal{S}$ if it is an ϵ–policy cover of $\mathcal{S}_h$ for all $h \in [H+1]$.*

Intuitively, we seek a policy cover of a small size, typically $O(|\mathcal{S}|)$, and with a small ϵ. Given such a cover, we can reach every state with the largest possible probability (up to ϵ) by executing each policy from the cover in turn. This enables us to collect a dataset of observations and rewards at all (sufficiently) reachable states s and further obtain a policy that maximizes any reward (details in Appendix B).

3. Embedding Approach

A key challenge in solving the BMDP exploration problem is the lack of access to the latent state s. Our algorithms work by explicitly learning a decoding function f which maps contexts to the corresponding latent states. This appears to be a hard unsupervised learning problem, even under the block-structure assumption, unless we make strong assumptions about the structure of $\mathcal{X}_s$ or about the emission distributions $q(\cdot \mid s)$. Here, instead of making assumptions about q or $\mathcal{X}_s$, we make certain "separability" assumptions about the latent transition probabilities p. Thus, we retain a broad flexibility to model rich context spaces, and also obtain the ability to efficiently learn a decoding function f. In this section, we define key components of our approach and formally state the separability assumption.

3.1. Embeddings and Function Approximation

In order to construct the decoding function f, we learn low-dimensional representations of contexts as well as latent states in a shared space, namely Δ_{MK}. We learn embedding functions $\mathbf{g} : \mathcal{X} \to \Delta_{MK}$ for contexts and $\boldsymbol{\phi} : \mathcal{S} \to \Delta_{MK}$ for states, with the goal that $\mathbf{g}(x)$ and $\boldsymbol{\phi}(s)$ should be close if and only if $x \in \mathcal{X}_s$. Such embedding functions always exist due to the block-structure: for any set of distinct vectors $\{\boldsymbol{\phi}(s)\}_{s \in \mathcal{S}}$, it suffices to define $\mathbf{g}(x) = \boldsymbol{\phi}(s)$ for $x \in \mathcal{X}_s$.

As we see later in this section, embedding functions $\boldsymbol{\phi}$ and $\mathbf{g}$ can be constructed via an essentially supervised approach, assuming separability. The state embedding $\boldsymbol{\phi}$ is a lower complexity object (a tuple of at most $|\mathcal{S}|$ points in Δ_{MK}), whereas the context embedding $\mathbf{g}$ has a high complexity for even moderately rich context spaces. Therefore, as is standard in supervised learning, we limit attention to functions $\mathbf{g}$ from some class $\mathcal{G} \subseteq \{\mathcal{X} \to \Delta_{MK}\}$, such as generalized linear models, tree ensembles, or neural nets. This is a form of function approximation where the choice of $\mathcal{G}$ includes any inductive biases about the structure of the contexts. By limiting the richness of $\mathcal{G}$, we can generalize across contexts as well as control the sample complexity of learning. At the same time, $\mathcal{G}$ needs to include embedding functions that reflect the block structure. Allowing a separate $\mathbf{g}_h \in \mathcal{G}$ for each level, we require realizability in the following sense:

Assumption 3.1 (Realizability)**.** *For any $h \in [H+1]$ and $\boldsymbol{\phi} : \mathcal{S}_h \to \triangle_{MK}$, there exists $\mathbf{g}_h \in \mathcal{G}$ such that $\mathbf{g}_h(x) = \boldsymbol{\phi}(s)$ for all $x \in \mathcal{X}_s$ and $s \in \mathcal{S}_h$.*

In words, the class $\mathcal{G}$ must be able to match any state-embedding function $\boldsymbol{\phi}$ across all blocks $\mathcal{X}_s$. To satisfy this assumption, it is natural to consider classes $\mathcal{G}$ obtained via a composition $\boldsymbol{\phi}' \circ f$ where f is a decoding function from some class $\mathcal{F} \subseteq \{\mathcal{X} \to \mathcal{S}\}$ and $\boldsymbol{\phi}'$ is any mapping $\mathcal{S} \to \triangle_{MK}$. Conceptually, f first decodes the context x to a state $f(x)$ which is then embedded by $\boldsymbol{\phi}'$ into $\triangle_{MK}$. The realizability assumption is satisfied as long as $\mathcal{F}$ contains a perfect decoding function f^*, for which $f^*(x) = s$ whenever $x \in \mathcal{X}_s$. The core representational power of $\mathcal{G}$ is thus driven by $\mathcal{F}$, the class of candidate decoding functions f.

Given such a class $\mathcal{G}$, our goal is find a suitable context-

[2] It suffices to consider maps $\mathcal{X}_{[h]} \to \mathcal{A}$ for $s \in \mathcal{S}_{h+1}$.

embedding function in $\mathcal{G}$ using a number of trajectories that is proportional to $\log|\mathcal{G}|$ when $\mathcal{G}$ is finite, or a more general notion of complexity such as a log covering number when $\mathcal{G}$ is infinite. Throughout this paper, we assume that $\mathcal{G}$ is finite as it serves to illustrate the key ideas, but our approach generalizes to the infinite case using standard techniques.

As we alluded to earlier, we learn context embeddings $\mathbf{g}_h$ by solving supervised learning problems. In fact, we only require the ability to solve least squares problems. Specifically, we assume access to an algorithm for solving vector-valued least-squares regression over the class $\mathcal{G}$. We refer to such an algorithm as the ERM oracle:

Definition 3.1 (ERM Oracle)**.** *Let $\mathcal{G}$ be a function class that maps $\mathcal{X}$ to $\triangle_{MK}$. An* empirical risk minimization oracle (ERM oracle) *for $\mathcal{G}$ is any algorithm that takes as input a data set $D = \{(x_i, \mathbf{y}_i)\}_{i=1}^n$ with $x_i \in \mathcal{X}$, $\mathbf{y}_i \in \triangle_{MK}$, and computes $\operatorname{argmin}_{\mathbf{g}\in\mathcal{G}} \sum_{(x,\mathbf{y})\in D} \|\mathbf{g}(x) - \mathbf{y}\|^2$.*

3.2. Backward Probability Vectors and Separability

For any distribution $\mathbb{P}$ over trajectories, we define *backward probabilities* as the conditional probabilities of the form $\mathbb{P}(s_{h-1}, a_{h-1} \mid s_h)$—note that conditioning is the opposite of transitions in p. For the backward probabilities to be defined, we do not need to fully specify a full distribution over trajectories, only a distribution ν over (s_{h-1}, a_{h-1}). For any such distribution ν, any $s \in \mathcal{S}_{h-1}$, $a \in \mathcal{A}$ and $s' \in \mathcal{S}_h$, the backward probability is defined as

$$b_\nu(s, a \mid s') = \frac{p(s' \mid s, a)\,\nu(s, a)}{\sum_{\tilde{s},\tilde{a}} p(s' \mid \tilde{s}, \tilde{a})\,\nu(\tilde{s}, \tilde{a})}. \tag{1}$$

For a given $s' \in \mathcal{S}_h$, we collect the probabilities $b_\nu(s, a \mid s')$ across all $s \in \mathcal{S}_{h-1}$, $a \in \mathcal{A}$ into the *backward probability vector* $\mathbf{b}_\nu(s') \in \triangle_{MK}$, padding with zeros if $|\mathcal{S}_{h-1}| < M$. Backward probability vectors are at the core of our approach, because they correspond to the state embeddings $\boldsymbol{\phi}(s)$ approximated by our algorithms. Our algorithms require that $\mathbf{b}_\nu(s')$ for different states $s' \in \mathcal{S}_h$ are sufficiently separated from one other for a suitable choice of ν:

Assumption 3.2 (γ-Separability)**.** *There exists $\gamma > 0$ such that for any $h \in \{2, \dots, H+1\}$ and any distinct $s', s'' \in \mathcal{S}_h$, the backward probability vectors with respect to the uniform distribution are separated by a margin of at least γ, i.e., $\|\mathbf{b}_\nu(s') - \mathbf{b}_\nu(s'')\|_1 \geq \gamma$, where $\nu = U(\mathcal{S}_{h-1} \times \mathcal{A})$.*

In Appendix F we show that the uniform distribution above can be replaced with any distribution supported on $\mathcal{S}_{h-1}\times\mathcal{A}$, although the margins γ would be different.

The key property that makes vectors $\mathbf{b}_\nu(s')$ algorithmically useful is that they arise as solutions to a specific least squares problem with respect to data generated by a policy whose marginal distribution over (s_{h-1}, a_{h-1}) matches ν. Let $\mathbf{e}_{(s,a)}$ denote the vector of the standard basis in $\mathbb{R}^{MK}$ corresponding to the coordinate indexed by $(s, a) \in \mathcal{S}_{h-1} \times \mathcal{A}$. Then the following statement holds:

Theorem 3.1. *Let ν be a distribution supported on $\mathcal{S}_{h-1} \times \mathcal{A}$ and let $\tilde{\nu}$ be a distribution over (s, a, x') defined by sampling $(s, a) \sim \nu$, $s' \sim p(\cdot \mid s, a)$, and $x' \sim q(\cdot \mid s')$. Let*

$$\mathbf{g}_h \in \operatorname*{argmin}_{\mathbf{g}\in\mathcal{G}} \mathbb{E}_{\tilde{\nu}} \left[\left\| \mathbf{g}(x') - \mathbf{e}_{(s,a)} \right\|^2 \right]. \tag{2}$$

Then, under Assumption 3.1, every minimizer $\mathbf{g}_h$ satisfies $\mathbf{g}_h(x') = \mathbf{b}_\nu(s')$ for all $x' \in \mathcal{X}_{s'}$ and $s' \in \mathcal{S}_h$.

The distribution $\tilde{\nu}$ is exactly the marginal distribution induced by a policy whose marginal distribution over (s_{h-1}, a_{h-1}) matches ν. Any minimizer $\mathbf{g}_h$ yields context embeddings corresponding to state embeddings $\boldsymbol{\phi}(s') = \mathbf{b}_\nu(s')$. Our algorithms build on Theorem 3.1: they replace the expectation by an empirical sample and obtain an approximate minimizer $\hat{\mathbf{g}}_h$ by invoking an ERM oracle.

4. Algorithm for Separable BMDPs

With the main components defined, we can now derive our algorithm for learning a policy cover in a separable BMDP.

The algorithm proceeds inductively, level by level. On each level h, we learn the following objects:

- The set of discovered latent states $\widehat{\mathcal{S}}_h \subseteq [M]$ and a decoding function $\hat{f}_h : \mathcal{X} \to \widehat{\mathcal{S}}_h$, which allows us to identify latent states at level h from observed contexts.
- The estimated transition probabilities $\hat{p}(\hat{s}_h \mid \hat{s}_{h-1}, a)$ across all $\hat{s}_{h-1} \in \widehat{\mathcal{S}}_{h-1}$, $a \in \mathcal{A}$, $\hat{s}_h \in \widehat{\mathcal{S}}_h$.
- A set of $(h-1)$-step policies $\Pi_h = \{\pi_{\hat{s}}\}_{\hat{s}\in\widehat{\mathcal{S}}_h}$.

We establish a correspondence between the discovered states and true states via a bijection α_h, under which the functions $\hat{f}_h$ accurately decode contexts into states, the probability estimates $\hat{p}$ are close to true probabilities, and Π_h is an ϵ–policy cover of $\mathcal{S}_h$. Specifically, we prove the following statement for suitable accuracy parameters ϵ_{f}, ϵ_{p} and ϵ:

Claim 4.1. *There exists a bijection $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$ such that the following conditions are satisfied for all $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$, $a \in \mathcal{A}$, $\hat{s}' \in \widehat{\mathcal{S}}_h$, and $s = \alpha_{h-1}(\hat{s}_{h-1})$, $s' = \alpha_h(\hat{s}')$, where α_{h-1} is the bijection for the previous level:*

$$\textit{Accuracy of } \hat{f}_h\textit{:} \quad \mathbb{P}_{x'\sim q(\cdot|s')}\big[\hat{f}_h(x') = \hat{s}'\big] \geq 1 - \epsilon_{\mathrm{f}}, \tag{3}$$

Accuracy of $\hat{p}$:

$$\sum_{\hat{s}''\in\widehat{\mathcal{S}}_h,\ s''=\alpha_h(\hat{s}'')} \left|\hat{p}(\hat{s}'' \mid \hat{s}, a) - p(s'' \mid s, a)\right| \leq \epsilon_{\mathrm{p}}, \tag{4}$$

$$\textit{Coverage by } \Pi_h\textit{:} \quad \mathbb{P}^{\pi_{\hat{s}'}}(s') \geq \mu(s') - \epsilon. \tag{5}$$

Algorithm 1 constructs $\widehat{\mathcal{S}}_h$, $\hat{f}_h$, $\hat{p}$ and Π_h level by level. Given these objects up to level $h-1$, the construction for the

Algorithm 1 PCID (**P**olicy **C**over via **I**nductive **D**ecoding)

1: **Input**:
  N_{g}: sample size for learning context embeddings
  N_{ϕ}: sample size for learning state embeddings
  N_{p}: sample size for estimating transition probabilities
  $\tau > 0$: a clustering threshold for learning latent states
2: **Output**: policy cover $\Pi = \Pi_1 \cup \cdots \cup \Pi_{H+1}$
3: Let $\widehat{\mathcal{S}}_1 = \{s_1\}$. Let $\hat{f}_1(x) = s_1$ for all $x \in \mathcal{X}$.
4: Let $\Pi_1 = \{\pi_0\}$ where π_0 is the trivial 0-step policy.
5: Initialize $\hat{p}$ to an empty mapping.
6: **for** $h = 2, \ldots, H+1$ **do**
7: Let $\eta_h = U(\Pi_{h-1}) \odot U(\mathcal{A})$
8: Execute η_h for N_{g} times. $D_{\mathrm{g}} = \{\hat{s}^i_{h-1}, a^i_{h-1}, x^i_h\}_{i=1}^{N_{\mathrm{g}}}$ for $\hat{s}_{h-1} = \hat{f}_{h-1}(x_{h-1})$.
9: Learn $\hat{\mathbf{g}}_h$ by calling ERM oracle on input D_{g}: $\hat{\mathbf{g}}_h = \operatorname{argmin}_{\mathbf{g} \in \mathcal{G}} \sum_{(\hat{s},a,x') \in D_{\mathrm{g}}} \left\| \mathbf{g}(x') - \mathbf{e}_{(\hat{s},a)} \right\|^2$.
10: Execute η_h for N_{ϕ} times. $\mathcal{Z} = \{\widehat{\mathbf{z}}_i = \hat{\mathbf{g}}_h(x^i_h)\}_{i=1}^{N_{\phi}}$.
11: Learn $\widehat{\mathcal{S}}_h$ and the state embedding map $\widehat{\boldsymbol{\phi}}_h : \widehat{\mathcal{S}}_h \to \mathcal{Z}$ by clustering $\mathcal{Z}$ with threshold τ (see Algorithm 2).
12: Define $\hat{f}_h(x') = \operatorname{argmin}_{\hat{s} \in \widehat{\mathcal{S}}_h} \left\| \widehat{\boldsymbol{\phi}}(\hat{s}) - \hat{\mathbf{g}}_h(x') \right\|_1$.
13: Execute η_h for N_{p} times. $D_{\mathrm{p}} = \{\hat{s}^i_{h-1}, a^i_{h-1}, \hat{s}^i_h\}_{i=1}^{N_{\mathrm{p}}}$ for $\hat{s}_{h-1} = f_{h-1}(x_{h-1})$, $\hat{s}_h = f_h(x_h)$.
14: Define $\hat{p}(\hat{s}_h \mid \hat{s}_{h-1}, a_{h-1})$ equal to empirical conditional probabilities in D_{p}.
15: **for** $\hat{s}' \in \widehat{\mathcal{S}}_h$ **do**
16: Run Algorithm 3 with inputs $\hat{p}$ and $\hat{s}'$ to obtain $(h-1)$-step policy $\psi_{\hat{s}'} : \widehat{\mathcal{S}}_{[h-1]} \to \mathcal{A}$.
17: Set $\pi_{\hat{s}'}(x_\ell) = \psi_{\hat{s}'}(\hat{f}_\ell(x_\ell))$, $\ell \in [h-1]$, $x_\ell \in \mathcal{X}_\ell$.
18: **end for**
19: Let $\Pi_h = (\pi_{\hat{s}})_{\hat{s} \in \widehat{\mathcal{S}}_h}$.
20: **end for**

Algorithm 2 Clustering to Find Latent-state Embeddings.

Input: Data points $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1}^n$ and threshold $\tau > 0$.
Output: Cluster indices $\widehat{\mathcal{S}}$ and centers $\widehat{\boldsymbol{\phi}} : \widehat{\mathcal{S}} \to \mathcal{Z}$.
Let $\widehat{\mathcal{S}} = \emptyset$, $k = 0$ (number of clusters).
while $\mathcal{Z} \neq \emptyset$ **do**
  Pick any $\mathbf{z} \in \mathcal{Z}$ (a new cluster center).
  Let $\mathcal{Z}' = \{\mathbf{z}' \in \mathcal{Z} : \|\mathbf{z} - \mathbf{z}'\|_1 \leq \tau\}$.
  Add cluster: $k \leftarrow k+1$, $\widehat{\mathcal{S}} \leftarrow \widehat{\mathcal{S}} \cup \{k\}$, $\widehat{\boldsymbol{\phi}}(k) = \mathbf{z}$.
  Remove the new cluster: $\mathcal{Z} \leftarrow \mathcal{Z} \setminus \mathcal{Z}'$.
end while

next level h proceeds in the following three steps, annotated with the lines in Algorithm 1 where they appear:

(1) Regression step: learn $\hat{\mathbf{g}}_h$ (lines 7–9). We collect a dataset of trajectories by repeatedly executing a specific policy mixture η_h. We use $\hat{f}_{h-1}$ to identify $\hat{s}_{h-1} = \hat{f}_{h-1}(x_{h-1})$ on each trajectory, obtaining samples $(\hat{s}_{h-1}, a_{h-1}, x_h)$ from $\tilde{\nu}$ induced by η_h. The context embedding $\hat{\mathbf{g}}_h$ is then obtained by solving the empirical version of (2).

Our specific choice of η_h ensures that each state s_{h-1} is reached with probability at least $(\mu_{\min} - \epsilon)/M$, which is bounded away from zero if ϵ is sufficiently small. The uniform choice of actions then guarantees that each state on the next level is also reached with sufficiently large probability.

(2) Clustering step: learn $\widehat{\boldsymbol{\phi}}$ **and** $\hat{f}_h$ (lines 10–12). Thanks to Theorem 3.1, we expect that $\hat{\mathbf{g}}_h(x') \approx \mathbf{g}_h(x') = \mathbf{b}_\nu(s')$ for the distribution $\nu(\hat{s}_{h-1}, a_{h-1})$ induced by η_h.[3] Thus, all contexts x' generated by the same latent state s' have embedding vectors $\hat{\mathbf{g}}_h(x')$ close to each other and to $\mathbf{b}_\nu(s')$. Thanks to separability,[4] we can therefore use clustering to identify all contexts generated by the same latent state, and this procedure is sample-efficient since the embeddings are low-dimensional vectors. Each cluster corresponds to some latent state s' and any vector $\hat{\mathbf{g}}_h(x')$ from that cluster can be used to define the state embedding $\widehat{\boldsymbol{\phi}}(s')$. The decoding function $\hat{f}_h$ is defined to map any context x' to the state s' whose embedding $\widehat{\boldsymbol{\phi}}(s')$ is the closest to $\hat{\mathbf{g}}_h(x')$.

(3) Dynamic programming: construct Π_h (lines 13–19). Finally, with the ability to identify states at level h via $\hat{f}_h$, we can use collected trajectories to learn an approximate transition model $\hat{p}(\hat{s}' \mid \hat{s}, a)$ up to level h. This allows us to use dynamic programming to find policies that (approximately) optimize the probability of reaching any specific state $s' \in \mathcal{S}_h$. The dynamic programming finds policies $\psi_{\hat{s}'}$ that act by directly observing decoded latent states. The policies $\pi_{\hat{s}'}$ are obtained by composing $\psi_{\hat{s}'}$ with the decoding functions $\{\hat{f}_\ell\}_{\ell \in [h-1]}$.

The next theorem guarantees that with a polynomial number of samples, Algorithm 1 finds a small ϵ–policy cover.[5]

Theorem 4.1 (Sample Complexity of Algorithm 1)**.** *Fix any* $\epsilon = O\left(\frac{\mu_{\min}^3 \gamma}{M^4 K^3 H}\right)$ *and a failure probability* $\delta > 0$*. Set* $N_{\mathrm{g}} = \tilde{\Omega}\left(\frac{M^4 K^4 H \log|\mathcal{G}|}{\epsilon \mu_{\min}^3 \gamma^2}\right)$, $N_{\phi} = \tilde{\Theta}\left(\frac{MK}{\mu_{\min}}\right)$, $N_{\mathrm{p}} = \tilde{\Omega}\left(\frac{M^2 K H^2}{\mu_{\min} \epsilon^2}\right)$, $\tau = \frac{\gamma}{30MK}$*. Then with probability at least* $1-\delta$*, Algorithm 1 returns an* ϵ*–policy cover of* $\mathcal{S}$*, with size at most* MH*.*

In addition to dependence on the usual parameters like M, K, H and $1/\epsilon$, our sample complexity also scales inversely with the separability margin γ and the worst-case

[3] Theorem 3.1 uses distributions ν and $\tilde{\nu}$ over true states s_{h-1}, but its analog also holds for distributions over $\hat{s}_{h-1}$, as long as decoding is approximately correct at the previous level.

[4] Although Assumption 3.2 is stated w.r.t. the uniform distribution, in Appendix F we show that it automatically implies separatability under any fully supported distribution.

[5] The $\tilde{O}(\cdot)$, $\tilde{\Omega}(\cdot)$, and $\tilde{\Theta}(\cdot)$ notation suppresses factors that are polynomial in $\log M$, $\log K$, $\log H$ and $\log(1/\delta)$.

Algorithm 3 Dynamic Programming for Reaching a State

Input: target state $\hat{s}^* \in \widehat{\mathcal{S}}_h$, dynamics $\hat{p}(\hat{s}' \mid \hat{s}, a)$ for all $\hat{s} \in \widehat{\mathcal{S}}_\ell$, $a \in \mathcal{A}$, $\hat{s}' \in \widehat{\mathcal{S}}_{\ell+1}$, $\ell \in [h-1]$.
Output: policy $\psi : \widehat{\mathcal{S}}_{[h-1]} \to \mathcal{A}$ maximizing $\hat{\mathbb{P}}^{\psi}(\hat{s}^*)$.
Let $v(\hat{s}^*) = 1$ and let $v(\hat{s}) = 0$ for all other $\hat{s} \in \widehat{\mathcal{S}}_h$.
for $\ell = h-1, h-2, \ldots, 1$ **do**
  for $\hat{s} \in \widehat{\mathcal{S}}_\ell$ **do**
    $\psi(\hat{s}) = \max_{a \in \mathcal{A}} \left[\sum_{\hat{s}' \in \widehat{\mathcal{S}}_{\ell+1}} v(\hat{s}')\, \hat{p}(\hat{s}' \mid \hat{s}, a) \right]$.
    $v(\hat{s}) = \sum_{\hat{s}' \in \mathcal{S}_{\ell+1}} v(\hat{s}')\, \hat{p}(\hat{s}' \mid \hat{s}, a = \psi(\hat{s}))$.
  end for
end for

reaching probability $\mu_{\min}$. While the exact dependence on these parameters is potentially improvable, Appendix F suggest that some inverse dependence is unavoidable for our approach. Compared with Azizzadenesheli et al. (2016a), there is no explicit dependence on $|\mathcal{X}|$, although they make spectral assumptions instead of the explicit block structure.

4.1. Deterministic BMDPs

As a special case of general BMDPs, many prior works study the case of deterministic transitions, that is, $p(s' \mid s, a) = 1$ for a unique state s' for each s, a. Also, many simulation-based empirical RL benchmarks exhibit this property. We refer to these BMDPs as deterministic, but note that only the transitions p are deterministic, not the emissions q. In this special case, the algorithm and guarantees of the previous section can be improved, and we present this specialization here, both for a direct comparison with prior work and potential usability in deterministic environments.

To start, note that $\mu_{\min} = 1$ and $\gamma = 2$ in any deterministic BDMP. The former holds as any reachable state is reached with probability one. For the latter, if (s, a) transitions to s', then (s, a) cannot appear in the backward distribution of any other state s''. Consequently, the backward probabilities for distinct states $s' \in \mathcal{S}_h$ must have disjoint support over $(s, a) \in \mathcal{S}_{h-1} \times \mathcal{A}$, and thus their ℓ_1 distance is exactly two.

Deterministic transitions allow us to obtain the policy cover with $\epsilon = 0$; that is, we learn policies that are guaranteed to reach any given state s with probability one. Moreover, it suffices to consider policies with simple structure: those that execute a fixed sequence of actions. Also, since we have access to policies reaching states in the prior level with probability one, there is no need for a decoding function $\hat{f}_{h-1}$ when learning states and context embeddings on level h. The final, more technical implication of determinism (which we explain below) is that it allows us to boost the accuracy of the context embedding in the clustering step, leading to improved sample complexity.

The details are presented in Algorithm 4. At each level

Algorithm 4 PCID for Deterministic BMDPs

1: **Input**:
  N_{g}: sample size for learning context embeddings
  N_{b}: sample size for boosting embedding accuracy
  $\tau > 0$: a clustering threshold for learning latent states
2: **Output**: policy cover $\Pi = \Pi_1 \cup \cdots \cup \Pi_{H+1}$
3: Let $\widehat{\mathcal{S}}_1 = \{s_1\}$, $\Pi_1 = \{\pi_0\}$ for the 0-step policy π_0.
4: **for** $h = 2, \ldots, H+1$ **do**
5: Let $\eta = U(\Pi_{h-1}) \odot U(\mathcal{A})$
6: Execute η_h for N_{g} times. $D_{\mathrm{g}} = \{\hat{s}^i_{h-1}, a^i_{h-1}, x^i_h\}_{i=1}^{N_{\mathrm{g}}}$ where $\hat{s}_{h-1}$ is the index of $\pi_{\hat{s}_{h-1}}$ sampled by η.
7: Learn $\hat{\mathbf{g}}_h$ by calling the ERM oracle on input D_{g}: $\hat{\mathbf{g}}_h = \operatorname{argmin}_{\mathbf{g} \in \mathcal{G}} \sum_{(\hat{s}, a, x') \in D_{\mathrm{g}}} \left\| \mathbf{g}(x') - \mathbf{e}_{(\hat{s}, a)} \right\|^2$.
8: Initialize $\mathcal{Z} = \emptyset$ (dataset for learning latent states).
9: **for** $(\pi, a) \in \Pi_{h-1} \times \mathcal{A}$ **do**
10: Execute D_{b} for N_{b} times, set $D_{\mathrm{b}} = \{x^i_h\}_{i=1}^{N_{\mathrm{b}}}$.
11: Set $\mathbf{z}_{\pi \odot a} = \sum_{x \in D_{\mathrm{b}}} \hat{\mathbf{g}}_h(x) / |D_{\mathrm{b}}|$, add $\mathbf{z}_{\pi \odot a}$ to $\mathcal{Z}$.
12: **end for**
13: Learn $\widehat{\mathcal{S}}_h$ and the state embedding map $\widehat{\phi}_h : \widehat{\mathcal{S}}_h \to \mathcal{Z}$ by clustering $\mathcal{Z}$ with threshold τ (see Algorithm 2).
14: Set $\Pi_h = (\pi_{\hat{s}})_{\hat{s} \in \widehat{\mathcal{S}}_t}$ where $\pi_{\hat{s}} = \pi \odot a$ if $\widehat{\phi}_h(\hat{s}) = \mathbf{z}_{\pi \odot a}$.
15: **end for**

$h \in [H+1]$, we construct the following objects:

- A set of discovered states $\widehat{\mathcal{S}}_h$.
- A set of $(h-1)$-step policies $\Pi_h = \{\pi_{\hat{s}}\}_{\hat{s} \in \widehat{\mathcal{S}}_h}$.

We proceed inductively and for each level h prove that the following claim holds with a high probability:

Claim 4.2. *There exists a bijection $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$ such that $\pi_{\hat{s}}$ reaches $\alpha_h(\hat{s})$ with probability one.*

This implies that $\widehat{\mathcal{S}}_h$ can be viewed as a latent state space, and Π_h is an ϵ–policy cover of $\mathcal{S}_h$ with $\epsilon = 0$.

To construct these objects for next level h, Algorithm 4 proceeds in three steps similar to Algorithm 1 for the stochastic case. The regression step, that is, learning of $\hat{\mathbf{g}}_h$ (lines 6–8), is identical. The clustering step (lines 9–15) is slightly more complicated. We boost the accuracy of the learned context embedding $\hat{\mathbf{g}}_h$ by repeatedly sampling contexts that are guaranteed to be emitted from the same latent state (because they result from the same sequence of actions), and taking an average. This step allows us to get away with a lower accuracy of $\hat{\mathbf{g}}_h$ compared with Algorithm 1. Finally, the third step, learning of Π_h (line 16), is substantially simpler. Since any action sequence reaching a given cluster can be picked as a policy to reach the corresponding latent state, dynamic programming is not needed.

The following theorem characterizes the sample complexity of Algorithm 4. It shows we only need $\tilde{O}\big(M^2 K^2 H \log |\mathcal{G}|\big)$

samples to find a policy cover with $\epsilon = 0$.

Theorem 4.2 (Sample Complexity of Algorithm 4)**.** *Set $\tau = 0.01$, $N_{\mathrm{g}} = \tilde{\Omega}(M^2K^2 \log|\mathcal{G}|)$ and $N_{\mathrm{b}} = \tilde{\Omega}(MK)$. Then with probability at least $1-\delta$, Algorithm 4 returns an ϵ–policy cover of $\mathcal{S}$, with $\epsilon = 0$ and size at most MH.*

The policy cover we compute can be used within a PAC RL algorithm to optimize a reward. As one example, if the reward depends on the latent state, we can use the policy cover to reach each state-action pair and collect $O(1/\epsilon^2)$ samples to estimate the expected reward for this state-action to accuracy ϵ. Thus, using at most $O(MKH/\epsilon^2)$ samples in addition to those needed by Algorithm 4, we can find the trajectory with the largest expected reward within an $H\epsilon$ error. To summarize (see also Appendix B):

Corollary 4.1. *With probability at least $1-\delta$, Algorithm 4 can be used to find an ϵ-suboptimal policy using at most $\tilde{O}\big(M^2K^2H\log|\mathcal{G}| + MKH^3/\epsilon^2\big)$ trajectories from a deterministic BMDP.*

We can now compare Corollary 4.1 with the related work of Dann et al. (2018). Our result significantly improves dependence on M, H and ϵ compared with their $O(M^3H^8K/\epsilon^5)$ bound, although their function-class complexity term is not directly comparable to ours, as their work approximates optimal value functions and policies, while we approximate ideal decoding functions.

5. Experiments

We perform an empirical evaluation of our decoding-based algorithms in six challenging RL environments—some meeting the BMDP assumptions and some not—with two choices for the function class $\mathcal{G}$. We compare our algorithm, which operates directly on rich observations, against two tabular algorithms that operate on the latent state.

The environments. All environments share the same latent structure, and are a form of "combination lock," with H levels, 3 states per level, and 4 actions. Non-zero reward is only achievable from states $s_{1,h}$ and $s_{2,h}$. From $s_{1,h}$ and $s_{2,h}$ one action leads with probability $1-\alpha$ to $s_{1,h+1}$ and with probability α to $s_{2,h+1}$, another has the flipped behavior, and the remaining two lead to $s_{3,h+1}$. All actions from $s_{3,h}$ lead to $s_{3,h+1}$. The "good" actions are randomly assigned for every state. From $s_{1,H}$ and $s_{2,H}$, two actions receive $\mathrm{Ber}(1/2)$ reward; all others provide zero reward. The start state is $s_{1,1}$. We consider deterministic variant ($\alpha = 0$) and stochastic variant ($\alpha = 0.1$). (See Appendix C.)

The environments are designed to be difficult for exploration. For example, the deterministic variant has 2^H paths with non-zero reward, but 4^H paths in total, so random exploration requires exponentially many trajectories.

We also consider two observation processes, which we use only for our algorithm, while the baselines operate directly on the latent state space. In *Lock-Bernoulli*, the observation space is $\{0,1\}^{H+3}$ where the first 3 coordinates are reserved for one-hot encoding of the state and the last H coordinates are drawn iid from $\mathrm{Ber}(1/2)$. This space meets the BMDP assumptions and can be perfectly decoded via linear functions. Note that the space is not partitioned across time, which our algorithms track internally. In *Lock-Gaussian*, the observation space is $\mathbb{R}^{H+3}$. As before the first 3 coordinates are reserved for one-hot encoding of the state, but this encoding is corrupted with Gaussian noise. Formally, if the agent is at state $s_{i,h}$ the observation is $\mathbf{e}_i + \mathbf{v} \in \mathbb{R}^{3+H}$, where $\mathbf{e}_i$ is one of the first three standard basis vectors and $\mathbf{v}$ has $\mathcal{N}(0,\sigma^2)$ entries. We consider $\sigma \in \{0.1, 0.2, 0.3\}$. Note that these environments *do not* satisfy Assumption 3.1 since the emission distributions cannot be perfectly separated.

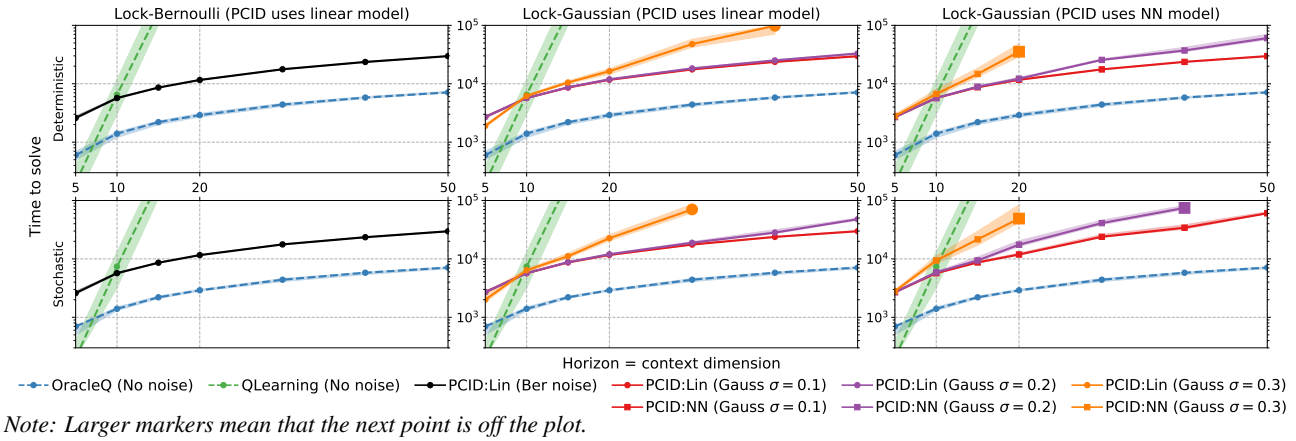
Baselines, hyperparameters. We compare our algorithms against two *tabular* approaches that cheat by directly accessing the latent states. The first, ORACLEQ, is the Optimistic Q-Learning algorithm of Jin et al. (2018), which has a near-optimal regret bound in tabular environments and serves as a skyline.[6] The second, QLEARNING, is tabular Q-learning with ϵ-greedy exploration. This algorithm serves as a baseline: any algorithm with strategic exploration should vastly outperform QLEARNING, even though it is cheating.

Each algorithm has two hyperparameters that we tune. In our algorithm (PCID), we use k-means clustering instead of Algorithm 2, so one of the hyperparameters is the number of clusters k. The second one is the number of trajectories n to collect in each outer iteration. For ORACLEQ, these are the learning rate α and a confidence parameter c. For QLEARNING, these are the learning rate α and $\epsilon_{\text{frac}} \in [0,1]$, a fraction of the 100K episodes over which to anneal the exploration probability linearly from 1 down to 0.01.

For both *Lock-Bernoulli* and *Lock-Gaussian*, we experiment with linear decoding functions, which we fit via ordinary least squares. For *Lock-Gaussian* only, we also use two-layer neural networks. Specifically, these functions are of the form $f(\mathbf{x}) = \mathbf{W}_2^\top \mathrm{sigmoid}(\mathbf{W}_1^\top \mathbf{x} + \mathbf{c})$ with the standard sigmoid activation, where the inner dimension is set to the clustering hyper-parameter k. These networks are trained using AdaGrad with a fixed learning rate of 0.1, for a maximum of 5K iterations. See Appendix C for more details on hyperparameters and training.

Experimental setup. We run the algorithms on all environments with varying H, which also influences the dimension of the observation space. Each algorithm runs for 100K episodes and we say that it has *solved the lock* by episode t if at round t its running-average reward is $\geq 0.25 = 0.5V^\star$.

[6] We use the Hoeffding version, which is conceptually much simpler, but statistically slightly worse.



Note: Larger markers mean that the next point is off the plot.

Figure 1. Time-to-solve against problem difficulty for the Lock environment with two observation processes and function classes. Left: *Lock-Bernoulli* environment. Center: *Lock-Gaussian* with linear functions, Right: *Lock-Gaussian* with neural networks. Top row: deterministic latent transitions. Bottom row: stochastic transitions with switching probability 0.1. ORACLEQ and QLEARNING are cheating and operate directly on latent states.

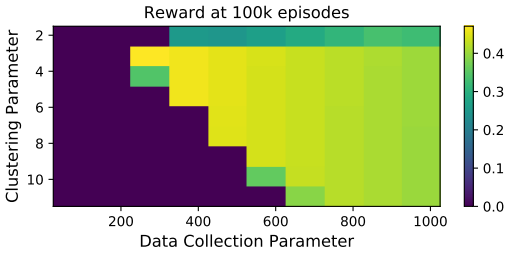


Figure 2. Sensitivity analysis for PCID on *Lock-Bernoulli* with $H = 20$, showing robustness to overestimating hyperparameters.

The *time-to-solve* is the smallest t for which the algorithm has solved the lock. For each hyperparameter, we run 25 replicates with different randomizations of the environment and seeds, and we plot the median time-to-solve of the best hyperparameter setting (along with error bands corresponding to 90^{th} and 10^{th} percentiles) against the horizon H.

Results. The results are in Figure 1 in a log-linear plot. First, QLEARNING works well for small horizon problems but cannot solve problems with $H \geq 15$ within 100K episodes, which is not surprising.[7] The performance curve for QLEARNING is linear, revealing an exponential sample complexity, and demonstrating that these environments cannot be solved with naïve exploration. As a second observation, ORACLEQ performs extremely well, and as we verify in Appendix C demonstrates a linear scaling with H.[8]

In *Lock-Bernoulli*, PCID is roughly a factor of 5 worse than the skyline ORACLEQ for all values of H, but the curves have similar behavior. In Appendix C, we verify a *near-linear* scaling with H, even better than predicted by our theory. Of course PCID is an exponential improvement over QLEARNING with ϵ-greedy exploration here.

In *Lock-Gaussian* with linear functions, the results are similar for the low-noise setting, but the performance of PCID degrades as the noise level increases. For example, with noise level $\sigma = 0.3$, it fails to solve the stochastic problem with $H = 40$ in 100K episodes. On the other hand, the performance is still quite good, and the scaling represents a dramatic improvement over QLEARNING.

Finally, PCID with neural networks is less robust to noise and stochasticity in *Lock-Gaussian*. Here, with $\sigma = 0.3$ the algorithm is unable to solve the $H = 30$ problem, both with and without stochasticity, but still does quite well with $\sigma \in \{0.1, 0.2\}$. The scaling with H is still quite favorable.

Sensitivity analysis. Lastly, we perform a simple sensitivity analysis to assess how the hyperparameters influence the behavior of PCID. In Figure 2 we display a heat-map showing the running-average reward (taking median over 25 replicates) of the algorithm on the stochastic *Lock-Bernoulli* environment with $H = 20$ as we vary both n and k. The best parameter choice here is $k = 3$ and $n = 300$. As we expect, if we under-estimate either k or n the algorithm fails, either because it cannot identify all latent states, or it does not collect enough data to solve the induced regression problems. On the other hand, the algorithm is quite robust to over-estimating both parameters, with a graceful degradation in performance.

Summary. We have shown on several rich-observation environments with both linear and non-linear functions that PCID scales to large-horizon rich-observation problems.

[7] We actually ran QLEARNING for 1M episodes and found it solves $H = 15$ with 170K episodes.

[8] This is incomparable with the result in Jin et al. (2018) since we are not measuring regret here.

It dramatically outperforms tabular QLEARNING with ϵ-greedy exploration, and is roughly a factor of 5 worse than ORACLEQ, an extremely effective tabular method, run on the corresponding tabular environment. Finally, the performance degrades gracefully as the assumptions are violated, and the algorithm is fairly robust to hyperparameter choices.

References

Antos, A., Szepesvári, C., and Munos, R. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 2008.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. Reinforcement learning of POMDPs using spectral methods. In *Conference on Learning Theory*, 2016a.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. Reinforcement learning in rich-observation MDPs using spectral methods. *arxiv:1611.03907*, 2016b.

Bagnell, J. A., Kakade, S. M., Schneider, J. G., and Ng, A. Y. Policy search by dynamic programming. In *Advances in Neural Information Processing Systems*, 2004.

Brafman, R. I. and Tennenholtz, M. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 2002.

Dann, C., Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. On oracle-efficient PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, 2018.

Ernst, D., Geurts, P., and Wehenkel, L. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 2005.

Givan, R., Dean, T., and Greig, M. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 2003.

Hallak, A., Di-Castro, D., and Mannor, S. Model selection in Markovian processes. In *International Conference on Knowledge Discovery and Data Mining*, 2013.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 2010.

Jiang, N., Kulesza, A., and Singh, S. Abstraction selection in model-based reinforcement learning. In *International Conference on Machine Learning*, 2015.

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*, 2017.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is Q-learning provably efficient? In *Advances in Neural Information Processing Systems*, 2018.

Johnson, M., Hofmann, K., Hutton, T., and Bignell, D. The Malmo Platform for artificial intelligence experimentation. In *International Joint Conference on Artificial Intelligence*, 2016.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 2002.

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

Lattimore, T. and Hutter, M. PAC bounds for discounted MDPs. In *International Conference on Algorithmic Learning Theory*, 2012.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 2015.

Oh, J., Singh, S., and Lee, H. Value prediction network. In *Advances in Neural Information Processing Systems*, 2017.

Ortner, R., Maillard, O.-A., and Ryabko, D. Selecting near-optimal approximate state representations in reinforcement learning. In *International Conference on Algorithmic Learning Theory*, 2014.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. Deep exploration via bootstrapped DQN. In *Advances in Neural Information Processing Systems*, 2016.

Ostrovski, G., Bellemare, M. G., Oord, A. v. d., and Munos, R. Count-based exploration with neural density models. In *International Conference on Machine Learning*, 2017.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017.

Silver, D., van Hasselt, H., Hessel, M., Schaul, T., Guez, A., Harley, T., Dulac-Arnold, G., Reichert, D., Rabinowitz, N., Barreto, A., and Degris, T. The predictron: End-to-end learning and planning. In *International Conference on Machine Learning*, 2017.

Weissman, T., Ordentlich, E., Seroussi, G., Verdu, S., and Weinberger, M. J. Inequalities for the L1 deviation of the empirical distribution. *Hewlett-Packard Labs, Tech. Rep*, 2003.

Whitt, W. Approximations of dynamic programs, I. *Mathematics of Operations Research*, 1978.

A. Comparison of BMDPs with other related frameworks

The problem setup in a BMDP is closely related to the literature on state abstractions, as our decoding function can be viewed as an abstraction over the rich context space. Since we learn the decoding function instead of assuming it given, it is worth comparing to the literature on state abstraction learning. The most popular notion of abstraction in model-based RL is bisimulation (Whitt, 1978; Givan et al., 2003), which is more general than our setup since our context is sampled i.i.d. conditioned on the hidden state (the irrelevant factor discarded by a bisimulation may not be i.i.d.). Such generality comes with a cost as learning good abstractions turns out to be very challenging. The very few results that come with finite sample guarantees can only handle a small number of candidate abstractions (Hallak et al., 2013; Ortner et al., 2014; Jiang et al., 2015). In contrast, we are able to learn a good decoding function from an exponentially large and unstructured family (that is, the decoding functions $g_h \in \mathcal{G}$ combined with the state encodings ϕ).

The setup and algorithmic ideas in our paper are related to the work of Azizzadenesheli et al. (2016a;b), but we are able to handle continuous observation spaces with no direct dependence on the number of unique contexts due to the use of function approximation. The recent setup of Contextual Decision Processes (CDPs) with low Bellman rank, introduced by Jiang et al. (2017) is a strict generalization of BMDPs (the Bellman rank of any BMDP is at most M). The additional assumptions made in our work enable the development of a computationally efficient algorithm, unlike in their general setup. Most similar to our work, Dann et al. (2018) study a subclass of CDPs with low Bellman rank where the transition dynamics are deterministic.[9] However, instead of the deterministic dynamics in Dann et al., we consider stochastic dynamics with certain reachability and separability conditions. As we note in Section 4, these assumptions are trivially valid under deterministic transitions. In terms of the realizability assumptions, Assumption 3.1 posits the realizability of a decoding function, while Dann et al. (2018) assume realizability of the optimal value function. These assumptions are not directly comparable, but are both reasonable if the decoding and value functions implicitly first map the contexts to hidden states, followed by a tabular function as discussed after Assumption 3.1. Finally as noted by Dann et al., certain empirical RL benchmarks such as visual grid world are captured reasonably well in our setting.

On the empirical side, (Pathak et al., 2017) learn a encoding function that compresses the rich obervations to a low-dimensional representation, which serves a similar purpose as our decoding function, using prediction errors in the low-dimensional space to drive exploration. This approach has weaknesses, as it cannot cope with stochastic transition structures. Given this, our work can also be viewed as a rigorous fix for these types of empirical heuristics.

B. Incorporating Rewards in BMDPs

At a high level, there are two natural choices for modeling rewards in a BMDP. In some cases, the rewards might only depend on the latent state. This is analogous to how rewards are typically modeled in the POMDP literature, for instance and respects the semantics that s is indeed a valid state to describe an optimal policy or value function. For such problems, finding a near optimal policy or value function building on Algorithms 1 or 4 is relatively straightforward. Note that along with the policy cover, our algorithms implicitly construct an approximately correct dynamics model $\hat{p}$ in the latent state space as well as decoding functions $\hat{f}$ which map contexts to the latent states generating them with a small error probability. While these objects are explicit in Algorithm 1, they are implicit in Algorithm 4 since each policy in the cover reaches a unique latent state with probability 1 so that we do not need any decoding function as highlighted before. Indeed for deterministic BMDPs, we do not need the dynamics model at all given the policy cover to maximize a state-dependent reward as shown in Corollary 4.1. For stochastic BMDPs, given any reward function, we can simply plan within the dynamics model over the latent states to obtain a near-optimal policy as a function of the latent state. We construct a policy $\widehat{\pi}$ as a function of contexts by first decoding the context using $\hat{f}$ and then applying the near-optimal policy over latent states found above. As we show in the following sections, there are parameters ϵ_{f} and ϵ_{p} controlled by our algorithms, such that the policy found using the procedure described above is at most $O\big(H(\epsilon_{\mathrm{f}} + \epsilon_{\mathrm{p}})\big)$ suboptimal.

In the second scenario where the reward depends on contexts, the optimal policies and value functions cannot be constructed using the latent states alone. However, our policy cover can still be used to generate a good exploration dataset for subsequent use in off-policy RL algorithms, as it guarantees good coverage for each state-action pair. Concretely, if we use value-function approximation, then the dataset can be fed into an approximate dynamic programming (ADP) algorithm (e.g., FQI Ernst et al., 2005). Given a good exploration dataset, these approaches succeed under certain representational assumptions

[9] While not explicitly assumed in their work, the assumption of the optimal policy and value functions depending only on the current observation and not hidden state is most reasonable when the observations are disjoint across hidden states like in this work.

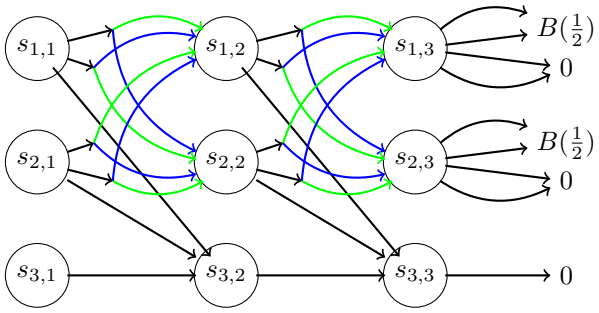


Figure 3. Lock transition diagram. The process is layered with time moving from left to right. Green arrows denote high probability transitions (either 1.0, or 0.9) while blue arrows denote low probability transitions (0.0 or 0.1). The agent starts in $s_{1,1}$. All states have four actions (all actions have the same effect for $s_{3,h}$), and the action labels are randomized for each replicate.

on the value-function class (Antos et al., 2008). Similarly, one can use PSDP style policy learning methods from such a dataset (Bagnell et al., 2004).

We conclude this subsection by observing that in reward maximization for RL, most works fall into either seeking a PAC or a regret guarantee. Our approach of first constructing a policy cover and then learning policies or value functions naturally aligns with the PAC criterion, but not with regret minimization. Nevertheless, as we see in our empirical evaluation, for challenging RL benchmarks, our approach still has a good performance in terms of regret. We wrap up this section by discussing the relationship of the BMDP framework with similar related problem settings in the literature.

C. Experimental Details and Reproducibility Checklist

C.1. Implementation Details

Environment transition diagram. The hidden state transition diagram for the Lock environment is displayed in Figure 3.

Our implementation of PCID follows Algorithm 1, with a few small differences. First, we set $N_g = N_\phi = N_p = n$, where n is a tuned hyperparameter. The first data collection step in Line 8 is as described: uniform over $\Pi_{h-1} \circ \mathcal{A}$ for n samples. The oracle in Line 9 is implemented differently for each representation as we detail below. Then rather than collect n additional samples in Line 10, we simply re-use the data from Line 8. For clustering, as mentioned, we use K-means clustering rather than the canopy-style subroutine described in Algorithm 2. We describe this in more detail below. We re-use data in lieu of the last data-collection step in Line 13, and the transition probabilities are estimated simpy via empirical frequencies. Finally, the policies Π_h are learned via dynamic programming on the learned latent transition model.

The two steps that require further clarification are the implementation of the oracle and the clustering step.

Oracle Implementation and representation. Whenever we use PCID with a linear representation, we use unregularized linear regression, e.g., ordinary least squares. Since we have vector-valued predictions, we performan linear regression independently on each coordinate. Formally, with data matrix $X \in \mathbb{R}^{n \times d}$ and targets $Y \in \mathbb{R}^{n \times p}$ the parameter matrix $\hat{\beta} \in \mathbb{R}^{d \times p}$ is

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y$$

We solve for $\hat{\beta}$ exactly, modulo standard numerical methods for performing the matrix inverse (specifically, `numpy.linalg.pinv`). Note that we do not add an intercept term to this problem.

When we use a neural network oracle, the representation is always $f(x) = W_2^\top \text{sigmoid}(W_1^\top x + c)$. For dimensions, if the clustering hyper-parameter is k, the observation space has dimension d, and the targets for the regression problem have dimension p, then the weight matrices have $W_2 \in \mathbb{R}^{k \times p}, W_1 \in \mathbb{R}^{d \times k}$ and the intercept term is $c \in \mathbb{R}^k$. $\text{sigmoid}(z) = (1 + e^{-z})^{-1}$ is the standard sigmoid activation, and we always use the square loss. To fit the model, we use AdaGrad with a fixed learning rate multiplier of 0.1. With output dimension p, each iteration of optimization makes p updates to the model, one for each output dimension in ascending order. As a rudimentary convergence test, we compute the total training loss

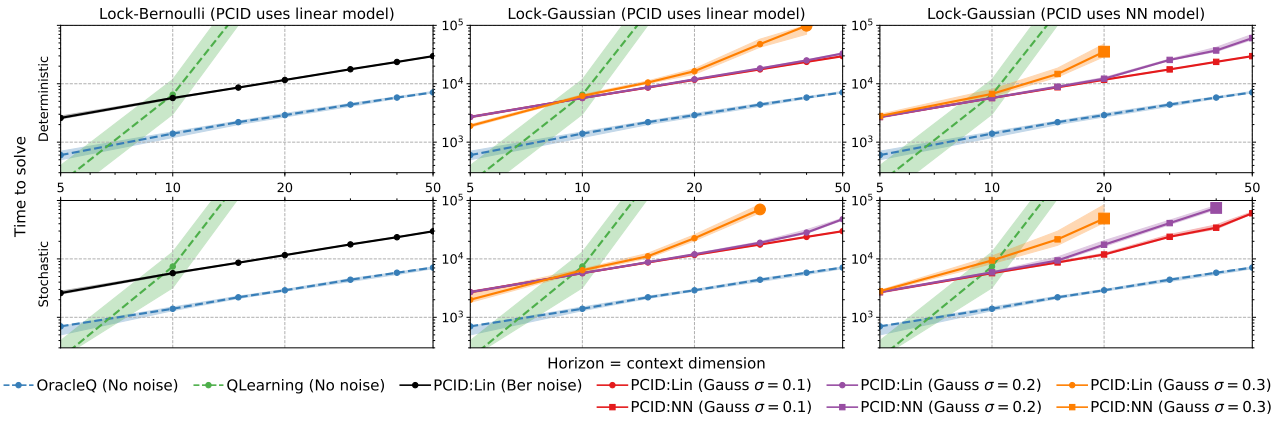


Figure 4. Time-to-solve against problem difficulty for the `Lock` environment with two different observation processes and function classes, plotted now in a log-log plot. The curves confirm a linear scaling with difficult for both PCID and ORACLEQ.

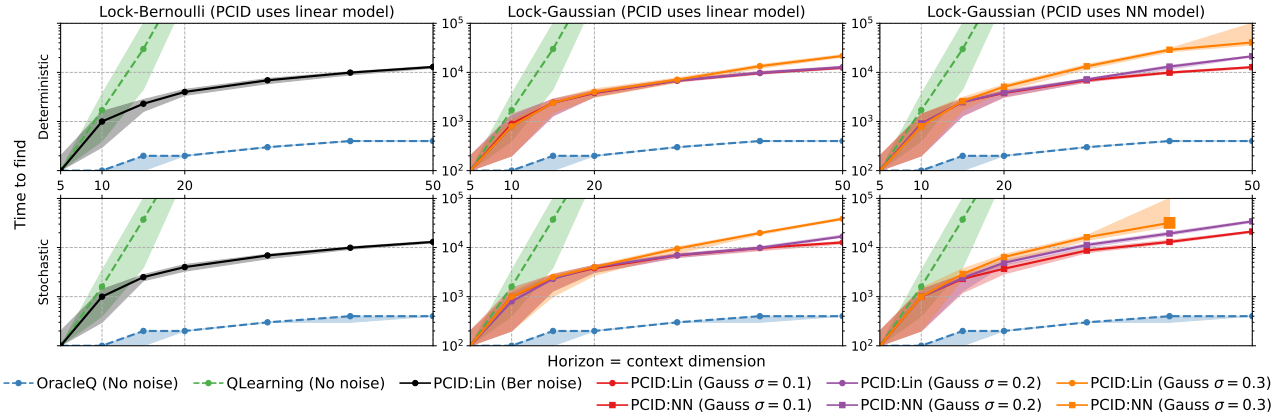


Figure 5. Time-to-find the goal against problem difficulty for the `Lock` environment with two different observation processes and function classes. Left: Lock-Bernoulli environment. Center: Lock-Gaussian with linear functions, Right: Lock-Gaussian with neural networks. Top row: deterministic latent transitions. Bottom row: stochastic transitions with switching probability 0.1. ORACLEQ and QLEARNING operate directly on hidden states, and hence are invariant to observation processes.

(over all output dimensions) at each iteration. For each $t \in \mathbb{N}$, we check if the training loss at iteration $100t$ is within 10^{-3} of the training loss at round $100(t-1)$. If so, we terminate optimization. We always terminate after 5000 iterations. Our neural network model and training is implemented in `pytorch`.

Clustering. We use the `scikit-learn` implementation of K-means clustering for clustering in the latent embedding space, with a simple model selection subroutine as a wrapper to tune the number of clusters. Starting with k set to the hyperparameter used as input to PCID, we run K-means, searching for k clusters, and we check if each found clusters has at least 30 points. If not, we decrease k and repeat.

C.2. Additional Results

In Figure 4 we plot exactly the same results as in Figure 1 except we visualize the results on a log-log plot. This verifies the linear scaling with H for both ORACLEQ and PCID in *Lock-Bernoulli* and *Lock-Gaussian* with linear functions. The slope for the line-of-best fit for ORACLEQ in the deterministic setting is 1.065 and in the stochastic setting it is 1.013. For ORACLEQ, this corresponds to the exponent on H in the sample complexity. On *Lock-Bernoulli*, PCID has slope 1.051 in

both settings, in the log-log scale, as above this corresponds to the exponent on H in our sample complexity, but since we have bound $d = H$ in these experiments, a linear dependence on H is substantially better than what our theory predicts.

In Figure 5 we use a different performance measure to compare the three algorithms, but all other details are identical to the results in Figure 1. Here we measure the *time-to-find*, which is the first episode for which the agent has non-zero total reward. Since the environments have no immediate reward, and almost all trajectories receive zero reward, this metric more closely corresponds to solving the exploration problem, while *time-to-solve* requires exploration and exploitation. We use time-to-solve in the main text because it is a better fit for the baseline algorithms.

As before, we plot the median time-to-find with error bars corresponding to 90^{th} and 10^{th} percentiles for the best hyperparameter, over 25 replicates, for each algorithm and in each environment. As sanity checks, ORACLEQ always finds the goal extremely quickly, and QLEARNING always fails for $H = 20$, which is unsurprising. Qualitatively the results for PCID are similar to those in Figure 1, but notice that with neural network representation, PCID almost always finds the goal in 100K episodes, even if it is unable to accumulate high reward. This suggests either a failure in *exploitation*, which is not the focus of this work, or that the agent would solve the problem with a few more episodes.

C.3. Reproducibility Checklist

- **Data collection process.** There was no dataset collection for this paper, but see below for details about environments and how results were collected.
- **Datasets/Environments.** Environments are implemented in the OpenAI Gym API. Source code for environments are included with submission and will be made publicly available.
- **Train/Validation/Test Split.** Our performance metrics are akin to regret, and require no train/test split. Nevertheless, we used different random seeds for development and for the final experiment.
- **Excluded data.** No data was excluded.
- **Hyperparmeters.** For ORACLEQ and QLEARNING we consider learning rates in $\{1^{-x} : x \in \{-4, \ldots, 0\}\}$. For ORACLEQ we chooose the confidence parameter from the same set. For QLEARNING the exploration parameter, the fraction of the learning process over which to linearly decay the exploration probability, is chosen from $\{0.0001, 0.001, 0.01, 0.1, 0.5\}$. For PCID, in Figures 1 and 5, we set the K-means parameter to 3 and we choose $n \in \{100, 200, \ldots, 900, 1000\}$. Hyperparameters were selected as follows: for each environment/horizon pair we choose the hyperparameter with best 90^{th} percentile performance. If the 90^{th} percentile for all hyperparameters exceeds the training time, we choose the hyperparameter with the best median performance. If this fails we optimize for 10^{th} percentile performance.
- **Evaluation Runs.** We always perform 25 replicates.
- **Experimental Protocol.** All algorithms (with various hyperparameter configurations) are run in each environment (with different featurization and stochasticity) for 100K episodes. In each of the 25 replicates we change the random seed for both the environment and the algorithm. We record total reward every 100 episodes. Since ORACLEQ and QLEARNING operate directly on the hidden states, we do not re-run with different featurizations.
- **Reported Results.** In Figures 1 and 5, we display the *time-to-solve* and *time-to-find* for each algorithm. Time-to-solve is the first episode t (rounded up to the nearest 100) for which the agent's running-average reward is at least $0.5V^\star$. Time to find is the first episode t (rounded up to the nearest 100) for which the agent has non-zero reward. For central tendency, we plot the median of these values over the 25 replicates. For error bars we plot the 90^{th} and 10^{th} percentile. In Figure 3, we plot the median running-average reward after 100K episodes. There are no error bars in this plot.
- **Computing Infrastructure.** All experiments were performed on a Linux compute cluster. Relevant software packages and versions are: python 3.6.7, numpy 1.14.3, scipy 1.1.0, scikit-learn 0.19.1, torch 0.4.0, gym 0.10.9, matplotlib 1.5.1.

D. Proofs of Deterministic BMDP

We begin with proofs for deterministic BMDPs as they are simpler and some of the arguments are reused in the more general stochastic case.

The following theorem shows the relation between number of samples and the risk. Since we are in the deterministic setting, given (s, a), the next hidden state is uniquely determined, we abuse the notation and let $s' = p(s, a)$

Theorem D.1. *Let $\hat{\mathbf{g}}_h$ be the function defined in line 8 of Algorithm 4. For any $\epsilon > 0$, if $n = \Omega\left(\frac{1}{\epsilon}\log\left(\frac{|\mathcal{G}|}{\delta}\right)\right)$, then with probability at least $1-\delta$ over the training samples D_{g}, we have*

$$\mathbb{E}_{(s,a)\sim U(\mathcal{S}_{h-1}\times\mathcal{A}),s'=p(s,a),x'\sim q(\cdot|s')}\left[\|\hat{\mathbf{g}}_h(x') - \mathbf{b}_U(s')\|_2^2\right] \le \epsilon.$$

Proof of Theorem D.1. The proof is a simple combination of empirical risk minimization analysis and Bernstein's inequality. Let Q_U denote the joint distribution over (s, a, s', x') such that $(s, a) \sim U(\mathcal{S}_{h-1} \times \mathcal{A}), s' = p(s, a), x' \sim q(\cdot \mid s')$. Define the population risk as

$$R(\mathbf{g}) = \mathbb{E}_{(s,a,s',x')\sim Q_U}\left[\left\|\mathbf{g}(x') - \mathbf{e}_{(s,a)}\right\|_2^2\right]$$

We also define the empirical risk as

$$\widehat{R}(\mathbf{g}) = \frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{g}(x_i') - \mathbf{e}_{(s_i,a_i)}\right\|_2^2 \triangleq \frac{1}{n}\sum_{i=1}^{n} L_i(\mathbf{g}).$$

Note $L_i(\mathbf{g}) \le 2\left(\|\mathbf{g}(x_i')\|_2^2 + \left\|\mathbf{e}_{(s_i,a_i)}\right\|_2^2\right) \le 4$ because $\|\mathbf{g}(x_i')\|_2^2 \le \|\mathbf{g}(x_i')\|_1^2 = 1$ and $\left\|e_{(s_i,a_i)}\right\|_2^2 = 1$. Recall $(s_i, a_i, s_i', x_i') \sim Q_U$. Recall that the minimizer $\mathbf{g}^*$ of $R(\mathbf{g})$ satisfies $\mathbf{g}^*(x') = \mathbf{b}_U(s')$ if $x' \sim q(\cdot \mid s')$. We bound the second moment of the excess risk $L_i(\mathbf{g}) - L_i(\mathbf{g}^*)$.

$$\begin{aligned}
\mathbb{E}\left[(L_i(\mathbf{g}) - L_i(\mathbf{g}^*))^2\right] =& \mathbb{E}\left[\left(\left\|\mathbf{g}(x') - \mathbf{e}_{(s,a)}\right\|_2^2 - \left\|\mathbf{g}^*(x') - \mathbf{e}_{(s,a)}\right\|_2^2\right)^2\right] \\
=& \mathbb{E}\left[\left(\left(\mathbf{g}(x') - \mathbf{g}^*(x')\right)^\top\left(\mathbf{g}(x') + \mathbf{g}^*(x') - 2\mathbf{e}_{(s,a)}\right)\right)^2\right] \\
\le& \mathbb{E}\left[\left(\|\mathbf{g}(x') - \mathbf{g}^*(x')\|_2\left\|\mathbf{g}(x') + \mathbf{g}^*(x') - 2\mathbf{e}_{(s,a)}\right\|_2\right)^2\right] \\
\le& 16\mathbb{E}\left[\|\mathbf{g}(x') - \mathbf{g}^*(x')\|_2^2\right] \\
=& 16\left(R(\mathbf{g}) - R(\mathbf{g}^*)\right).
\end{aligned}$$

where the expectation is taken over Q_U and the inequality we used $\left\|\mathbf{g}(x') + \mathbf{g}^*(x') - 2\mathbf{e}_{(s,a)}\right\|_2 \le \|\mathbf{g}(x')\|_2 + \|\mathbf{g}^*(x')\|_2 + 2\left\|\mathbf{e}_{(s,a)}\right\|_2 \le \|\mathbf{g}(x')\|_1 + \|\mathbf{g}^*(x')\|_1 + 2\left\|\mathbf{e}(s,a)\right\|_1 = 4$. Now we apply Bernstein inequality on the random variable $\hat{R}(\mathbf{g}) - \hat{R}(\mathbf{g}^*) - (R(\mathbf{g}) - R(\mathbf{g}^*)$ and obtain that if $n = \Omega\left(\frac{1}{\epsilon}\log\frac{|\mathcal{G}|}{\delta}\right)$ we have with probability at least $1-\delta$, for all $\mathbf{g} \in \mathcal{G}$

$$\left|\hat{R}(\mathbf{g}) - \hat{R}(\mathbf{g}^*) - (R(\mathbf{g}) - R(\mathbf{g}^*))\right| \le C\left(\sqrt{\frac{(R(\mathbf{g}) - R(\mathbf{g}^*))\log(|\mathcal{G}|/\delta)}{n}} + \frac{\log(|\mathcal{G}|)/\delta)}{n}\right)$$

for a universal constant $C > 0$. Note by definition $\hat{g}_h$ satisfies $\hat{R}(g) - \hat{R}(g^*) \le 0$ and $R(\hat{g}_h) - R(g^*) \le 0$ so we have

$$|R(\mathbf{g}_h) - R(g^*)| \le C\left(\sqrt{\frac{(R(\mathbf{g}_h) - R(\mathbf{g}^*))\log(|\mathcal{G}|/\delta)}{n}} + \frac{\log(|\mathcal{G}|)/\delta)}{n}\right).$$

It is easy to see $|R(\mathbf{g}_h) - R(\mathbf{g}^*)| = O\left(\frac{\log(|\mathcal{G}|/\delta)}{n}\right)$. Plugging in our choice of n, we prove the theorem.

□

Theorem D.2 (Restatement of Theorem 4.2)**.** *Set $\tau = 0.01$, $N_{\mathrm{g}} = \tilde{\Omega}(M^2K^2\log|\mathcal{G}|)$ and $N_\phi = \tilde{\Omega}(MK)$. Then with probability at least $1-\delta$, Algorithm 4 returns an ϵ–policy cover of $\mathcal{S}$, with $\epsilon = 0$.*

Proof of Theorem 4.2. The proof is by induction on levels. Our two induction hypothese are

- For $h' = 1, \ldots, h-1$, $\widehat{\mathcal{S}}_{h'}$ and $\mathcal{S}_{h'}$ are bijective, i.e., there exists a bijective function $\alpha : \widehat{\mathcal{S}}_{h'} \to \mathcal{S}_{h'}$.
- For $h' = 1, \ldots, h-1$, $\Pi_{h'}$ covers the state space with the scale equals to 0 (c.f. Definition 2.2).

Note these two hypotheses imply Claim 4.2.

To prove the induction, for the base case, this is true because the starting state is fixed. Now we prove the case $h' = h$.

For simplicity, we let Q_U be a distribution over (s, a, s', x') that $(s, a) \sim U\left[\mathcal{S}_{h-1} \times \mathcal{A}\right], s' = p(s, a), x' \sim q(\cdot \mid s')$. Note because $\mathcal{S}_{h-1}$ and $\widehat{\mathcal{S}}_{h-1}$ are bijective and Q_U can be also viewed as a distribution over (s, a, s', x') that the probability of the event $(\hat{s}, a, s', x')$ is the same as the event $(\alpha^{-1}(s), a, s', x')$. Therefore, notation-wise, in the proof of this theorem, s is equivalently to $\alpha^{-1}(s)$ and $\hat{s}$ is equivalent to $\alpha(\hat{s})$.

By Theorem D.1, we know if $N_{\mathrm{g}} = O\left(\frac{1}{\epsilon}\log\left(\frac{|\mathcal{G}|H}{\delta}\right)\right)$, we have with probability at least $1 - \frac{\delta}{H}$,

$$\mathbb{E}_{(s,a,s',x') \sim Q_U}\left[\left\|\mathbf{g}_h(x') - \mathbf{b}_U(s')\right\|_2^2\right] \leq \epsilon.$$

Recall because of the induction hypothesis and the definition of our exploration policy, we sample (s, a) uniformly, we must have for any $s' \in \mathcal{S}_h$

$$\mathbb{E}_{x' \sim q(\cdot|s')}\left[\left\|\mathbf{b}_U(s') - \mathbf{g}_h(x')\right\|_2^2\right] \leq KM\epsilon.$$

By Jensen's inequality, this implies that for $s' \in \mathcal{S}_h$,

$$\left\|\mathbf{b}_U(s') - \mathbb{E}_{x' \sim q(\cdot|s')}\left[\mathbf{g}_h(x')\right]\right\|_2^2 \leq KM\epsilon.$$

By AM-GM inequality, we know for any vector $\mathbf{v}$, $\|\mathbf{v}\|_1^2 \leq \dim(\mathbf{v})\|\mathbf{v}\|_2^2$. Therefore we have for any $s' \in \mathcal{S}_h$

$$\left\|\mathbf{b}_U(s') - \mathbb{E}_{x' \sim q(\cdot|s')}\left[\mathbf{g}_h(x')\right]\right\|_1^2 \leq K^2M^2\epsilon$$

Choosing $\epsilon = \frac{\tau^2}{100K^2M^2}$, we have with probability $1 - \delta$, for any $h = 0, \ldots, H-1$, $s' \in \mathcal{S}_h$,

$$\left\|\mathbf{b}_U(s') - \mathbb{E}_{s'=p(s,a), x' \sim q(\cdot|s')}\left[\mathbf{g}_h(x')\right]\right\|_1 \leq \tau/10.$$

The above analysis shows the learned $\hat{\mathbf{g}}_h$ has small error for all level and all states.

Next, by standard Hoeffding inequality, we know if $N_\phi = O\left(\frac{MK\log(MKH/\delta)}{\tau^2}\right)$ with probability at least $1 - \frac{\delta}{H}$, for $(s, a) \in \mathcal{S}_{h-1} \times \mathcal{A}$, we have

$$\left\|\mathbf{z}_{s\odot a} - \mathbb{E}_{s'=p(s,a), x' \sim q(\cdot|s')}\left[\mathbf{g}_h(x')\right]\right\|_1 \leq \tau/10.$$

Now consider an iteration in Algorithm 2. For any $\pi \odot a$, let s_π denotes state reached by following the policy π and let $s' = p(s_\pi, a)$. If there exists (π', a') with $s' = p(s_{\pi'}, a')$ and $\mathbf{z}_{\pi'\odot a'} \in \widehat{\mathcal{S}}_h$, then we know $\|\mathbf{z}_{a\circ\pi} - \mathbf{z}_{a'\circ\pi'}\|_1 \leq \frac{\tau}{5}$. Thus $\hat{z}_{a\circ\pi}$ will not be added to $\widehat{\mathcal{S}}_h$. On the other hand, suppose for all $\mathbf{z}_{\pi'\odot a'} \in \widehat{\mathcal{S}}_h$, $s' \neq p(s_{\pi'}, a')$. Let $s'' = p(s_{\pi'}, a')$. We know $\|\mathbf{z}_{\pi\odot a} - \mathbf{z}_{\pi'\odot a'}\|_1 \geq \|\mathbf{b}_U(s') - \mathbf{b}_U(s'')\|_1 - \tau/5 \geq 2 - \tau/5 \geq \tau$. Thus $\mathbf{z}_{\pi\odot a}$ will be added to $\widehat{\mathcal{S}}_h$. Note the above reasonings imply $\mathcal{S}_h$ and $\widehat{\mathcal{S}}_h$ are bijective and from Algorithm 4, it is clear that for all $s' \in \mathcal{S}_h$ we have stored one path (policy) in Π_h that can reach s'. Thus we prove our induction hypotheses at level h. Lastly we use union bound over $h = 1, \ldots, H$ and finish the proof.

□

Corollary D.1 (Restatement of Corollary 4.1)**.** *With probability at least $1-\delta$, Algorithm 4 can be used to find an ϵ-suboptimal policy using at most $\tilde{O}\big(M^2K^2H\log|\mathcal{G}| + MKH^3/\epsilon^2\big)$ trajectories from a deterministic BMDP.*

Proof of Corollary 4.1. For a fixed state action pair (s, a), we collect $\frac{1}{\epsilon^2H^2}\log(\frac{MHK}{\delta})$ samples. Using Hoeffding inequality, we know with probability at least $1 - \frac{\delta}{MHK}$, our estimated $\hat{r}(s, a)$ of this state-action pair satisfies

$$|\hat{r}(s, a) - r(s, a)| \leq \frac{1}{H\epsilon}.$$

Taking union bound over $h \in [H], s \in \mathcal{S}_h, a \in \mathcal{A}$, we know with probability at least $1-\delta$, for all state-action pair, we have

$$|\hat{r}(s,a) - r(s,a)| \leq \frac{1}{H\epsilon}. \tag{6}$$

Now let $(a_1, \ldots, a_H)$ be the sequence of actions that maximizes the total reward based on estimated reward. Let $\widehat{R}$ be the total estimated reward if we execute $(a_1, \ldots, a_H)$ and let R be the true reward. By Equation (6), we know with probability at least $1-\delta$ over the training samples, we have

$$\left|\widehat{R} - R\right| \leq \epsilon.$$

Now denote let $(a_1^*, \ldots, a_H^*)$ be the sequence of actions that maximizes the true total reward . Let $\widehat{R}^*$ be the total estimated reward if we execute $(a_1^*, \ldots, a_H^*)$ and let R^* be the true reward. Applying Equation (6) again, we know

$$\left|\widehat{R}^* - R^*\right| \leq \epsilon.$$

Now note

$$\begin{aligned} R - R^* =& R - \widehat{R} + \widehat{R} - \widehat{R}^* + \widehat{R}^* - R^* \\ \geq& R - \widehat{R} + \widehat{R}^* - R^* \\ \geq& -\left|R - \widehat{R}\right| - \left|\widehat{R}^* - R^*\right| \\ \geq& -2\epsilon. \end{aligned}$$

Rescaling ϵ we finish the proof. □

E. Proof of Theorem 3.1

Theorem E.1 (Restatement of Theorem 3.1)**.** *Let ν be a distribution supported on $\mathcal{S}_{h-1} \times \mathcal{A}$ and let $\tilde{\nu}$ be a distribution over (s, a, x') defined by sampling $(s, a) \sim \nu$, $s' \sim p(\cdot \mid s, a)$, and $x' \sim q(\cdot \mid s')$. Let*

$$\mathbf{g}_h \in \operatorname*{argmin}_{\mathbf{g} \in \mathcal{G}} \mathbb{E}_{\tilde{\nu}} \left[\left\| \mathbf{g}(x') - \mathbf{e}_{(s,a)} \right\|^2 \right].$$

Then, under Assumption 3.1, every minimizer $\mathbf{g}_h$ satisfies $\mathbf{g}_h(x') = \mathbf{b}_\nu(s')$ for all $x' \in \mathcal{X}_{s'}$ and $s' \in \mathcal{S}_h$.

Proof of Theorem 3.1. First, note that $\mathbf{b}_\nu(s')$ is the conditional mean of $(s, a) \in \mathcal{S}_{h-1} \times \mathcal{A}$ given s'. By the optimality of conditional mean in minimizing the least squares loss, we have

$$\mathbf{b}_\nu(s') = \operatorname*{argmin}_{V \in \mathbb{R}^{MK}} \mathbb{E}_{(s,a,s') \sim \tilde{\nu}} \left[\left\| V - e_{(s,a)} \right\|_2^2 \;\middle|\; s_h = s' \right].$$

Now we consider the embedding function $\boldsymbol{\phi}(s') = \mathbf{b}_\nu(s')$. Since $\boldsymbol{\phi}$ is a tabular mapping from $\mathcal{S}_h$ to $\triangle_{MK}$, it minimizes the unconditional squared loss, under any distribution over s'. Furthermore, by Assumption 3.1, we know there exists one $\mathbf{g}_h \in \mathcal{G}$ which satisfies that

$$\mathbf{g}_h(x') = \boldsymbol{\phi}(s') \text{ if } x' \sim q(\cdot \mid s').$$

Combing these facts we have

$$\mathbf{g}_h \in \operatorname*{argmin}_{\mathbf{g} \in \mathcal{G}} \mathbb{E}_{\tilde{\nu}} \left[\left\| \mathbf{g}(x') - \mathbf{e}_{(s,a)} \right\|^2 \right]$$

where we have moved from conditional to unconditional expectations in the last step using the tabular structure of ϕ as discussed above. This concludes the proof. □

F. Justification of Assumption 3.2 and Dependency on $\mu_{\min}$

The following theorem shows some separability assumption is necessary for exploration methods based on the backward conditional probability representation.

Theorem F.1 (Necessary and Sufficient Condition for State Identification Using Distribution over Previous State Action Pair)**.** *Fix $h \in \{2, \dots, H+1\}$ and let $s'_1, s'_2 \in \mathcal{S}_h$.*

- *Let U denote the uniform distribution over $\mathcal{S}_{h-1} \times \mathcal{A}$, if the backward probability satisfies $\mathbf{b}_U(s'_1) = \mathbf{b}_U(s'_2)$, then for any $\nu \in \triangle(\mathcal{S}_{h-1} \times \mathcal{A})$ we have*

$$\mathbf{b}_\nu(s'_1) = \mathbf{b}_\nu(s'_2).$$

- *If the transition probability satisfies $\mathbf{b}_U(s'_1) \neq \mathbf{b}_U(s'_2)$, then for any $\nu \in \triangle(\mathcal{S}_{h-1} \times \mathcal{A})$ that satisfies $\nu(s,a) > 0$ for any $(s,a) \in \mathcal{S}_{h-1} \times \mathcal{A}$,we have*

$$\mathbf{b}_\nu(s'_1) \neq \mathbf{b}_\nu(s'_2).$$

Proof of Theorem F.1. By Bayes rule, for any $s' \in \mathcal{S}_h$ we have

$$b_\nu(s,a|s') \propto \mathbb{P}_\nu(s,a)\, p(s'|s,a).$$

Let $\mathbf{p}_{s'} := [p(s' \mid s,a)]_{(s,a)\in\mathcal{S}_{h-1}\times\mathcal{A}}$. In matrix form,

$$\begin{aligned}\mathbf{b}_\nu(s') &\propto \mathbf{diag}(\nu)\mathbf{p}_{s'} \propto \mathbf{diag}(\nu)\,\frac{1}{MK}\,\mathbf{p}_{s'} \\ &\propto \mathbf{diag}(\nu)\mathbf{diag}(U)\mathbf{p}_{s'} \propto \mathbf{diag}(\nu)\mathbf{b}_U(s'). \end{aligned} \tag{7}$$

When $\mathbf{b}_U(s'_1) = \mathbf{b}_U(s'_2)$, $\mathbf{diag}(\nu)\mathbf{b}_U(s'_1) = \mathbf{diag}(\nu)\mathbf{b}_U(s'_2)$, which implies that $\mathbf{b}_\nu(s'_1) = \mathbf{b}_\nu(s'_2)$. This proves the first claim.

For the second claim, assume towards contradiction that there exists $\nu > 0$ such that $\mathbf{b}_\nu(s'_1) = \mathbf{b}_\nu(s'_2)$. From Equation (7) we have

$$\mathbf{b}_U(s'_1) \propto \mathbf{diag}(\nu)^{-1}\mathbf{b}_\nu(s'_1) = \mathbf{diag}(\nu)^{-1}\mathbf{b}_\nu(s'_2) \propto \mathbf{b}_U(s'_2),$$

which implies that $\mathbf{b}_U(s'_1) = \mathbf{b}_U(s'_2)$ and contradicts the condition of the claim. □

It shows if the backward probability induced by the uniform distribution over previous state-action pair cannot separate states at the current level, then the backward probability induced by any other distribution cannot do this either. Therefore, if in Assumption 3.2, $\gamma = 0$, by Theorem F.1, there is no way to differentiate s'_1 and s'_2.

The next lemma shows if there exists a margin induced by the uniform distribution, for any non-degenerate distribution we also have a margin.

Lemma F.1. *Let $\nu \in \triangle(\mathcal{S}_{h-1} \times \mathcal{A})$ with $\nu(s,a) \geq \tau$. Then under the Assumption 3.2 we have for any $s'_1, s'_2 \in \mathcal{S}_h$*

$$\|\mathbf{b}_\nu(s'_1) - \mathbf{b}_\nu(s'_2)\|_1 \geq \frac{\tau\gamma}{2}.$$

Proof of Lemma F.1. Recall

$$\mathbf{b}_\nu(s'_1) = \frac{\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)}{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)\|_1}.$$

Therefore we have

$$\|\mathbf{b}_\nu(s'_1) - \mathbf{b}_\nu(s'_2)\|_1 = \left\| \frac{\left\| \mathbf{diag}(\nu)\left(\mathbf{b}_U(s'_1) - \frac{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)\|_1}{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_2)\|_1}\cdot \mathbf{b}_U(s'_2)\right)\right\|_1}{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)\|_1} \right\|_1$$

$$\geq \min_{(s,a)} \nu(s,a) \left\| \left(\mathbf{b}_U(s'_1) - \frac{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)\|_1}{\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_2)\|_1} \cdot \mathbf{b}_U(s'_2) \right) \right\|_1$$
$$\geq \frac{\tau\gamma}{2}$$

where the first inequality we used Hölder 's inequality and the fact that $\|\mathbf{diag}(\nu)\mathbf{b}_U(s'_1)\|_1 \leq 1$ and the second inequality we used Lemma H.1. □

The following example shows the inverse dependency on $\mu_{\min}$ is unavoidable. Consider the following setting. At level $h-1$, there are two states $\mathcal{S}_{h-1} = \{s_1, s_2\}$ and there is only one action $\mathcal{A} = \{a\}$. There are two states at level $h = \{s'_1, s'_2\}$. The transition probability is

$$p(\cdot|\cdot) = \begin{pmatrix} 0.5 & 0.5 \\ 0.1 & 0.9 \end{pmatrix}.$$

where the first row represents s_1, the second row represents s_2, the first column represents s'_1 and the second column represents s'_2. By Theorem F.1, because the transition probability from s_1 to s'_1 and s'_2, we can only use s_2 to differentiate s_1, s'_2. However, if $\mu(s_2) = \exp(-\frac{1}{\epsilon})$, i.e., for all policy, the probability of getting to s_2 is exponentially small, then we cannot use s_2 for exploration and thus we cannot differentiate s'_1 and s'_2.

G. Proof of Theorem 4.1 and Claim 4.1

We prove the theorem by induction. We first provide a high-level outline of the proof, and then present the technical details. At each level $h \in [H]$, we establish that Claim 4.1 holds. For convenience, we break up the claim into three conditions corresponding to its different assertions, and establish each in turnup to a small failure probability. The first one is on the learned states and the decoding function.

Condition G.1 (Bijection between learned and true states)**.** *There exists $\epsilon_{\mathrm{f}} < \frac{1}{2}$ such that there is a bijective mapping $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$ for which*

$$\mathbb{P}_{x \sim q(\cdot|\alpha_h(\hat{s}))} \left[\hat{f}_h(x) = \hat{s} \right] \geq 1 - \epsilon_{\mathrm{f}}. \tag{8}$$

In words, this condition states that every estimated latent state $\hat{s}$ roughly corresponds to a true latent state $\alpha_h(\hat{s})$, when we use the decoding function $\hat{f}_h$. This is because all but an ϵ_{f} fraction of contexts drawn from $\alpha_h(\hat{s})$ are decoded to their true latent state, and for each latent state s, there is a distinct estimated state $\alpha_h^{-1}(s)$ as the map α_h is a bijection. For simplicity, we define $\mathbf{p}(s,a) \in \mathbb{R}^M$ to be the forward transition distribution over $\mathcal{S}_h$ for $s \in \mathcal{S}_{h-1}$ and $a \in \mathcal{A}$. We abuse notation to similarly use $\mathbf{p}(\hat{s},a) \in \mathbb{R}^M$ to be the vector $\{\mathbb{P}(s \mid \hat{s}, a)\}_{s \in \mathcal{S}_h}$ of conditional probabilities $\mathcal{S}_h$ for $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$ and $a \in \mathcal{A}$. Note that unlike $s \in \mathcal{S}_{h-1}$, $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$ is not a Markovian state and hence the conditional probability vector $\mathbf{p}(\hat{s},a)$ depends on the specific distribution over $\widehat{\mathcal{S}}_{h-1} \times \mathcal{A}$. In the following we will use $\mathbf{p}^{\nu}(\hat{s},a)$ to emphasize this dependency where ν is the distribution, where ν is a distribution over $\widehat{\mathcal{S}}_{h-1} \times \mathcal{A}$.

In the proof, we often compare two vectors indexed by $\mathcal{S}_h$ and $\widehat{\mathcal{S}}_h$. We will assume the order of the indices of these two vectors are matched according to α_h.

The second condition is on our estimated transition probability. This condition ensures our estimation has small error.

Condition G.2 (Approximately Correct Transition Probability)**.** *For any $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$, $a \in \mathcal{A}$, we have*

$$\|\widehat{\mathbf{p}}(\hat{s},a) - \mathbf{p}(s,a)\|_1 \leq \epsilon_{\mathrm{p}} \triangleq \min\left\{ \frac{\mu_{\min}\gamma}{10M^3HK}, \frac{\epsilon\mu_{\min}}{10H} \right\}.$$

The following lemma shows if the induction hypotheses hold, then we can prove main theorem.

Lemma G.1. *Assume Condition G.1 and G.2 hold for all $h \in [H]$. For any $h \in [H]$ and $s \in \mathcal{S}_h$, there exists $\hat{s} \in \widehat{\mathcal{S}}_h$ that the policy $\pi_{\hat{s}}$ satisfies $\mathbb{P}^{\pi_{\hat{s}}}(s) \geq \mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}$.*

Based on this lemma, since $\epsilon_{\mathrm{f}} \leq \frac{\epsilon}{3H}$ and $\epsilon_{\mathrm{p}} \leq \frac{\epsilon}{3H}$, we prove that the algorithm outputs a policy cover with parameter ϵ, completing the proof of Theorem 4.1.

Note that Condition G.1, Condition G.2 and Lemma G.1 together imply Claim 4.1.

In the rest of this section, we prove focus on establishing that these conditions hold inductively.

Analysis of Base Case $h = 1$ Since by assumption we know we are starting from s_1 and we set $\widehat{\mathcal{S}}_1 = \{s_1\}$, Conditions G.1 and G.2 directly hold. Note that the transition operator in this case simply corresponds to the degenerate distribution $\mathbf{p}_1$ with $\mathbf{p}_1(s_1) = \widehat{\mathbf{p}}_1(s_1) = 1$.

Now supposing that the induction hypotheses hold for $h_1 = 1, \ldots, h-1$, we focus on level h. We next show that Conditions G.1 and G.2 hold with probability at least $1 - \frac{\delta}{H}$. This suffices to ensure an overall failure probability of at most $1 - \delta$ as asserted in Theorem 4.1 via a union bound.

Establishing Condition G.1. In order to establish the condition, we need to show that our decoding function $\hat{f}_h$ predicts the underlying latent state correctly almost always. We do this in two steps. Since the functions $\hat{f}_h$ are derived based on $\hat{\mathbf{g}}_h$ and $\widehat{\phi}_h$, we analyze the properties of these two objects in the following two lemmas. In order to state the first lemma, we need some additional notation. Note that η_h and $\hat{f}_{h-1}$ induce a distribution over $\mathcal{S}_{h-1} \times \widehat{\mathcal{S}}_{h-1} \times \mathcal{A} \times \mathcal{S}_h$. We denote this distribution as ν_h. With this distribution, we define the conditional backward probability $\hat{\mathbf{b}}_{\nu_h} \colon \mathcal{S}_h \to \triangle\left(\widehat{\mathcal{S}}_{h-1} \times \mathcal{A}\right)$ as

$$\hat{b}_{\nu_h}(\hat{s}, a \mid s_1') = \frac{p_{h-1}^{\nu_h}(s_1' \mid \hat{s}, a)\mathbb{P}^{\nu_h}(\hat{s}, a)}{\sum_{\hat{s}_1, a_1} p_{h-1}^{\nu_h}(s_1' \mid \hat{s}_1, a_1)\mathbb{P}^{\nu_h}(\hat{s}, a_1)}. \tag{9}$$

Recall that $\mathbf{p}_{h-1}^{\nu_h}$ above refers to the distribution over s_1' according the transition dynamics, when $\hat{s}, a$ are induced by ν_h.

With this notation, we have the following lemma.

Lemma G.2. *Assume $\epsilon_{\mathrm{f}} \le \frac{\mu_{\min}^3 \gamma}{100 M^4 K^3}$. Then the distributions $\hat{b}_{\nu_h}(\hat{s}, a | s')$ are well separated for any pair $s_1', s_2' \in \mathcal{S}_h$:*

$$\left\|\hat{\mathbf{b}}_{\nu_h}(s_1') - \hat{\mathbf{b}}_{\nu_h}(s_2')\right\|_1 \ge \frac{\mu_{\min}\gamma}{3MK}. \tag{10}$$

Furthermore, if $N_{\mathrm{g}} = \Omega\left(\frac{M^3K^3}{\epsilon_{\mathrm{f}}\mu_{\min}^3\gamma^2}\log\left(\frac{|\mathcal{G}|H}{\delta}\right)\right)$, with probability at least $1 - \delta/H$, for every $s' \in \mathcal{S}_h$, $\hat{\mathbf{g}}_h$ satisfies

$$\mathbb{P}_{x' \sim q(\cdot|s')}\left[\left\|\hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \ge \frac{\gamma\mu_{\min}}{100MK}\right] \le \epsilon_{\mathrm{f}}. \tag{11}$$

The first part of Lemma G.2 tell us that the latent states at level h are well separated if we embed them using $\phi(s') = \hat{\mathbf{b}}_{\nu_h}(s')$ as the state embedding. The second part guarantees that our regression procedure estimates this representation accurately. Together, these assertions imply that any two contexts from the same latent state (up to an ϵ_{f} fraction) are close to each other, while contexts from two different latent states are well-separated. Formally, with probability at least $1 - \frac{\delta}{H}$ over the N_{g} training data:

1. For any $s' \in \mathcal{S}_h$ and $x_1', x_2' \sim q(\cdot \mid s')$, we have with probability at least $1 - 2\epsilon_{\mathrm{f}}$ over the emission process
$$\|\hat{\mathbf{g}}_h(x_1') - \hat{\mathbf{g}}_h(x_2')\|_1 \le \frac{\mu_{\min}\gamma}{50MK}. \tag{12}$$
2. For any $s_1', s_2' \in \mathcal{S}_h$ such that $s_1' \neq s_2'$, $x_1' \sim q(\cdot \mid s_1')$ and $x_2' \sim q(\cdot \mid s_2')$, we have with probability at least $1 - 2\epsilon_{\mathrm{f}}$ over the emission process
$$\|\hat{\mathbf{g}}_h(x_1') - \hat{\mathbf{g}}_h(x_2')\|_1 \ge \frac{\mu_{\min}\gamma}{4MK}. \tag{13}$$

In other words, the mapping of contexts, as performed through the functions $\hat{\mathbf{g}}_h$ should be easy to cluster with each cluster roughly corresponding to a true latent state. Our next lemma guarantees that with enough samples for clustering, this is indeed the case.

Lemma G.3 (Sample Complexity of the Clustering Step)**.** *If $N_\phi = \Theta\left(\frac{MK}{\mu_{\min}}\log(\frac{MH}{\delta})\right)$ and $\epsilon_{\mathrm{f}} \le \frac{\delta}{100HN_\phi}$ we have with probability at least $1 - \frac{\delta}{H}$, (1) for every $s' \in \mathcal{S}_h$, there exists at least one point $\mathbf{z} \in \mathcal{Z}$ such that $\mathbf{z} = \hat{\mathbf{g}}_h(x')$ with $x' \sim q(\cdot \mid s')$ and $\left\|\hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \le \frac{\mu_{\min}\gamma}{100MK}$ and (2) for every $\mathbf{z} = \hat{g}_h(x') \in \mathcal{Z}$ with $x' \sim q(\cdot \mid s')$, $\left\|\hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \le \frac{\mu_{\min}\gamma}{100MK}$.*

Based on Lemmas G.2 and G.3, we can establish that Condition G.1 holds with high probability. Note that Condition G.1 consists of two parts. The first part states that there exists a bijective map $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$. The second part states that the decoding error is small. To prove the first part, we explicitly construct the map α_h and show it is bijective. We define $\alpha_h : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$ as

$$\alpha_h(\hat{s}') = \operatorname*{argmin}_{s \in \mathcal{S}_h} \left\| \boldsymbol{\phi}(s') - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \tag{14}$$

First observe that for any $\hat{s}' \in \widehat{\mathcal{S}}_h$, by the second conclusion of Lemma G.3, we know there exists $s' \in \mathcal{S}_h$ such that

$$\left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s') \right\| \leq \frac{\gamma \mu_{\min}}{100MK}.$$

This also implies for any $s'' \neq s'$,

$$\left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s'') \right\| \geq \left\| \boldsymbol{\phi}(s'') - \boldsymbol{\phi}(s'') \right\| - \left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s') \right\| \geq \frac{\gamma \mu_{\min}}{4MK}.$$

Therefore we know $\alpha_h(\hat{s}') = s'$, i.e., α_h always maps the learned state to the correct original state.

We now prove α_h is injective, i.e., $\alpha(\hat{s}') \neq \alpha_h(\hat{s}'')$ for $\hat{s}' \neq \hat{s}'' \in \widehat{\mathcal{S}}_h$. Suppose there are $\hat{s}', \hat{s}'' \in \widehat{\mathcal{S}}_h$ such that $\alpha_h(\hat{s}') = \alpha_h(\hat{s}'') = s'$ for some $s' \in \mathcal{S}_h$. Then using the second conclusion of Lemma G.3, we know

$$\left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \widehat{\boldsymbol{\phi}}(\hat{s}'') \right\|_1 \leq \left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s') \right\|_1 + \left\| \boldsymbol{\phi}(s') - \widehat{\boldsymbol{\phi}}(\hat{s}'') \right\|_1 \leq \frac{\gamma \mu_{\min}}{50MK}.$$

However, we know by Algorithm 2, every $\hat{s}' \neq \hat{s}'' \in \widehat{\mathcal{S}}_h$ must satisfy

$$\left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \widehat{\boldsymbol{\phi}}(\hat{s}'') \right\|_1 > \tau = \frac{\gamma \mu_{\min}}{30MK}.$$

This leads to a contradiction and thus α_h is injective.

Next we prove α_h is surjective, i.e., for every $s' \in \mathcal{S}_h$, there exists $\hat{s}' \in \widehat{\mathcal{S}}_h$ such that $\alpha_h(\hat{s}') = s'$. The first conclusion in Lemma G.3 guarantees that for each latent state $s' \in \mathcal{S}_h$, there exists $\mathbf{z} = \hat{\mathbf{g}}(x') \in \mathcal{Z}$ with $x' \sim q(\cdot \mid s')$. The second conclusion of Lemma G.3 guarantees that

$$\left\| \mathbf{z} - \boldsymbol{\phi}(s') \right\|_1 \leq \frac{\gamma \mu_{\min}}{100MK}.$$

Now we first assert that all points in a cluster are emitted from the same latent state by combining Equation 10, the second part of Lemma G.3 and our setting of τ. Now the second part of Lemma G.3 implies that there exists $\hat{s}' \in \widehat{\mathcal{S}}_h$ such that $\left\| \mathbf{z} - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \leq \frac{\mu_{\min}\gamma}{50MK}$, since $\mathbf{z}$ and $\widehat{\boldsymbol{\phi}}(\hat{s}')$ correspond to $\hat{\mathbf{g}}$ evaluated on two different contexts in the same cluster. Therefore we have

$$\left\| \boldsymbol{\phi}(s') - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \leq \left\| \boldsymbol{\phi}(s') - \mathbf{z} \right\|_1 + \left\| \mathbf{z} - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \leq \frac{\mu_{\min}\gamma}{30MK}$$

Now we can show that $\alpha_h(\hat{s}') = s'$. To do this, we show that $\widehat{\boldsymbol{\phi}}(\hat{s}')$ is closer to $\boldsymbol{\phi}(s')$ than the embedding of any state in $\mathcal{S}_h$. Using the second conclusion of Lemma G.3 and Equation 10 we know for any $s'' \neq s'$

$$\left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s'') \right\|_1 \geq \left\| \boldsymbol{\phi}(s') - \boldsymbol{\phi}(s'') \right\|_1 - \left\| \widehat{\boldsymbol{\phi}}(\hat{s}') - \boldsymbol{\phi}(s') \right\|_1 \geq \frac{\gamma \mu_{\min}}{4MK}.$$

We know $s' = \operatorname{argmin}_{s_1 \in \mathcal{S}_h} \left\| \widehat{\boldsymbol{\phi}}(s_1) - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1$. Therefore, by the definition of α_h we know $\alpha_h(\hat{s}') = s$. Now we have finished the proof of the first part of Condition G.1.

For the second part of Condition G.1, note for any $s' \in \mathcal{S}_h$ and $x' \sim q(\cdot \mid s')$, by Lemma G.2, we know with probability at least $1 - \epsilon_{\mathrm{f}}$ over the emission process we have

$$\left\| \hat{\mathbf{g}}_h(x') - \boldsymbol{\phi}(s') \right\|_1 \leq \frac{\gamma \mu_{\min}}{100MK}.$$

For $\hat{s}' = \alpha_h^{-1}(s')$, we have

$$\left\| \hat{\mathbf{g}}_h(x') - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \leq \left\| \hat{\mathbf{g}}_h(x') - \boldsymbol{\phi}(s') \right\|_1 + \left\| \boldsymbol{\phi}(s') - \widehat{\boldsymbol{\phi}}(\hat{s}') \right\|_1 \leq \frac{\gamma \mu_{\min}}{50MK}.$$

On the other hand, for $\hat{s}'' \in \widehat{\mathcal{S}}_h$ with $\hat{s}'' \neq \alpha_h^{-1}(s')$, we have

$$\left\|\hat{\mathbf{g}}_h(x') - \widehat{\boldsymbol{\phi}}(\hat{s}'')\right\|_1 \geq -\left\|\hat{\mathbf{g}}_h(x') - \boldsymbol{\phi}(s')\right\|_1 + \left\|\boldsymbol{\phi}(s') - \boldsymbol{\phi}(\alpha_h(\hat{s}''))\right\|_1 - \left\|\boldsymbol{\phi}(\alpha_h(\hat{s}'')) - \widehat{\boldsymbol{\phi}}(\hat{s}'')\right\|_1 \geq \frac{\gamma\mu_{\min}}{4MK}.$$

Therefore we have with probability at least $1 - \epsilon_{\mathrm{f}}$

$$\hat{f}_h(x') = \operatorname*{argmin}_{\hat{s}' \in \widehat{\mathcal{S}}_h} \left\|\widehat{\boldsymbol{\phi}}(\hat{s}') - \hat{\mathbf{g}}_h(x')\right\|_1 = \alpha_h^{-1}(s'),$$

which is equivalent to the second part of Condition G.1.

Establishing Condition G.2. This part of our analysis is relatively more traditional, as we are effectively estimating a probability distribution from empirical counts in a tabular setting. The only care needed is to correctly handle the decoding errors due to which our count estimates for frequencies have a slight bias. The following lemma guarantees that Condition G.2 holds.

Lemma G.4. *If* $\epsilon_{\mathrm{f}} \leq \frac{\epsilon_{\mathrm{p}}\mu_{\min}}{10M^2}$ *and if* $N_{\mathrm{p}} = \Omega\left(\frac{M^2K}{\epsilon_{\mathrm{p}}^2}\log\frac{MHK}{\delta}\right)$*, we have that with probability at least* $1 - \frac{\delta}{H}$ *for every* $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$, $a \in \mathcal{A}$

$$\left\|\widehat{\mathbf{p}}(\hat{s}, a) - \mathbf{p}(\alpha_{h-1}(\hat{s}), a)\right\|_1 \leq \epsilon_{\mathrm{p}}, \tag{15}$$

In the following we present proof details.

G.1. Proof details for Theorem 4.1 and Claim 4.1

We first define some notations on policies that will be useful in our analysis. First, consider a policy ψ^{true} over the true hidden states for $h = 1, \ldots, H$:

$$\psi^{true} : \mathcal{S}_h \to \mathcal{A}, \psi^{true}(s) = a.$$

By the one-to-one correspondence between $\widehat{\mathcal{S}}_h$ and $\mathcal{S}_h$ ($\hat{s}$ and $\alpha(\hat{s})$),[10] ψ^{true} also induces a policy over the learned hidden states

$$\psi^{learned} : \widehat{\mathcal{S}}_h \to \mathcal{A}, \psi^{learned}(\hat{s}) = \psi^{true}(\alpha(\hat{s})).$$

Next, we let $f_1, \ldots, f_H$ be the decoding functions for the true states, i.e.,

$$f_h : \mathcal{X}_h \to \mathcal{S}_h, f_h(x) = s \text{ if and only if } x \sim q(\cdot \mid S).$$

Recall by Condition G.1, we also have approximately correct decoding functions: $\hat{f}_1, \ldots, \hat{f}_H$, which satisfy for all $h \in [H]$ and $s \in \mathcal{S}_h$

$$\hat{f}_h : \mathcal{X} \to \widehat{\mathcal{S}}_h, \mathbb{P}_{x \sim q(\cdot|s)}\left[\hat{f}_h(x) = \alpha^{-1}(s)\right] \geq 1 - \epsilon_{\mathrm{f}}.$$

Now we consider two policies induced by the policies on the hidden states and the decoding function

$$\begin{aligned} \pi^{true} &: \mathcal{X}_h \to \mathcal{A} \quad \pi^{true}(x) = \psi^{true}(f_h(x)) \\ \pi^{learned} &: \mathcal{X}_h \to \mathcal{A} \quad \pi^{learned}(x) = \psi^{learned}(\hat{f}_h(x)) \end{aligned}$$

The following figure shows the relations among these objects

$$\begin{array}{ccc} \psi^{true} : \mathcal{S}_h \to \mathcal{A} & & \overset{f_h}{\Rightarrow} \quad \pi^{true} : \mathcal{X}_h \to \mathcal{A} \\ \Downarrow \alpha & & \end{array}$$

[10] In this following we drop the subscript of α because the one-to-one correspondence is clear.

$$\psi^{learned} : \widehat{\mathcal{S}}_h \to \mathcal{A} \qquad \overset{\hat{f}_h}{\Rightarrow} \qquad \pi^{learned} : \mathcal{X}_h \to \mathcal{A}$$

In our algorithm we maintain estimations of the transition probabilities of the learned states

$$\{\hat{p}_h(\hat{s}' \mid \hat{s}, a)\}_{h\in[H],\hat{s}\in\widehat{\mathcal{S}}_{h-1},a\in\mathcal{A},\hat{s}'\in\widehat{\mathcal{S}}_h}$$

Given these estimated transition probabilities, a policy over the learned hidden states $\psi^{learned}$, and a target learned state $\hat{s}$, we have an estimation of the reaching probability $\hat{\mathbb{P}}^{\psi^{learned}}(s)$ which can be computed by dynamic programming as in the standard tabular MDP. With these notations, we can prove the following useful lemma.

Lemma G.5. *For any state $\hat{s} \in \widehat{\mathcal{S}}_h$, we have*

$$\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right| \le 2H\epsilon_{\mathrm{f}}.$$

Proof of Lemma G.5. Fixing any state $\hat{s} \in \widehat{\mathcal{S}}_h$, for any event $\mathcal{E}$ we have

$$\begin{aligned}
&\mathbb{P}^{\pi^{true}}(\mathcal{E}, \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) \\
=&\mathbb{P}^{\pi^{true}}\left(\alpha(\hat{f}_1(x_1)) = f_1(x_1)\right)\mathbb{P}^{\pi^{true}}\left(\alpha(\hat{f}_2(x_2)) = f_2(x_2)|\alpha(\hat{f}_1(x_1)) = f_1(x_1)\right) \\
&\cdots\mathbb{P}^{\pi^{true}}\left(\alpha(\hat{f}_h(x_h)) = f_h(x_h)|\alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_{h-1}(x_{h-1})) = f_{h-1}(x_{h-1})\right) \\
&\cdot\mathbb{P}^{\pi^{learned}}(\mathcal{E}|\alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) \\
=&\mathbb{P}^{\pi^{learned}}\left(\alpha(\hat{f}_1(x_1)) = f_1(x_1)\right)\mathbb{P}^{\pi^{learned}}\left(\alpha(\hat{f}_2(x_2)) = f_2(x_2)|\alpha(\hat{f}_1(x_1)) = f_1(x_1)\right) \\
&\cdots\mathbb{P}^{\pi^{learned}}\left(\alpha(\hat{f}_h(x_h) = f_h(x_h)|\alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_{h-1}(x_{h-1})) = f_{h-1}(x_{h-1})\right) \\
&\cdot\mathbb{P}^{\pi^{learned}}(\mathcal{E}|\alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) \\
=&\mathbb{P}^{\pi^{learned}}(\mathcal{E}, \alpha(\hat{f}_1(x_1)) = f_1(x_1)), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)).
\end{aligned}$$

because the event $\left\{\alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)\right\}$ happens and under this event π^{true} and $\pi^{learned}$ choose the same action at every level so the induced probability distribution is the same. Now we bound the target error.

$$\begin{aligned}
&\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right| \\
\le&\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{true}}(\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h))\right| \\
&+\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right| \\
=&\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{true}}(\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h))\right| \\
&+\left|\mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right|
\end{aligned}$$

To bound the first term, notice that the event $\{s_h = \alpha(\hat{s})\}$ is a superset of

$$\left\{s_h = \alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h))\right\}.$$

and

$$\begin{aligned}
&\{s_h = \alpha(\hat{s})\} \setminus \left\{s_h = \alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h))\right\} \\
=&\left\{s_h = \alpha(\hat{s}), \exists h_1 \in [h], \alpha(\hat{f}_{h_1}(x_{h_1}) \ne f_{h_1}(x_{h_1})\right\}
\end{aligned}$$

Therefore, we can bound

$$\begin{aligned}
&\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{true}}(\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h)) \\
=&\mathbb{P}^{\pi^{true}}\left(\alpha(\hat{s}), \exists h_1 \in [h], \alpha(\hat{f}_{h_1}(x_{h_1}) \neq f_{h_1}(x_{h_1})\right) \\
\leq&\mathbb{P}^{\pi^{true}}\left(\exists h_1 \in [h], \alpha(\hat{f}_{h_1}(x_{h_1}) \neq f_{h_1}(x_{h_1})\right) \\
\leq&\sum_{h_1=1}^{h} \mathbb{P}^{\pi^{true}}(\alpha(\hat{f}_{h_1}(x_{h_1})) \neq f_{h_1}(x_{h_1})) \\
\leq&h\epsilon_{\mathrm{f}} \\
\leq&H\epsilon_{\mathrm{f}}
\end{aligned}$$

Similarly, we can bound

$$\left|\mathbb{P}^{\pi^{learned}}\left[\alpha(\hat{s}), \alpha(\hat{f}_1(x_1)) = f_1(x_1), \ldots, \alpha(\hat{f}_h(x_h)) = f_h(x_h))\right] - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right| \leq H\epsilon_{\mathrm{f}}.$$

Combing these two inequalities we have

$$\left|\mathbb{P}^{\pi^{true}}(\alpha(\hat{s})) - \mathbb{P}^{\pi^{learned}}(\alpha(\hat{s}))\right| \leq 2H\epsilon_{\mathrm{f}}. \qquad \square$$

Now we are ready to prove some consequences of this result which will be used in the remainder of the proof.

Lemma G.6 (Restatement of Lemma G.1)**.** *Assume Conditions G.1 and G.2 hold for all $h \in [H]$. For any $h \in [H]$ and $s \in \mathcal{S}_h$, there exists $\hat{s} \in \widehat{\mathcal{S}}_h$ that the policy $\pi_{\hat{s}}$ satisfies $\mathbb{P}^{\pi_{\hat{s}}}(s) \geq \mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}$.*

Proof of Lemma G.1. For any given $s \in \mathcal{S}_h$, by our induction hypothesis, we know there exists $\hat{s} \in \widehat{\mathcal{S}}_h$ such that $\alpha(\hat{s}) = s$.

Now we lower bound $\mathbb{P}^{\pi_{\hat{s}}}(s)$. First recall $\pi_{\hat{s}}$ is of the form $\psi_{\hat{s}}(\hat{f}_{h_1}(x_{h_1}))$ for $1 \leq h_1 \leq h-1$, $x_{h_1} \in \mathcal{X}$ and $\psi_{\hat{s}}$ maximizes the reaching probability to $\hat{s}$ given estimated transition probabilities. To facilitate our analysis, we define an auxiliary policy for $h_1 = 1, \ldots, h-1$

$$\bar{\pi}_{\hat{s}} : \mathcal{X} \to \mathcal{A}, \bar{\pi}_{\hat{s}}(x_{h_1}) = \psi_{\hat{s}}(\alpha^{-1}(f_h(x_{h_1})))$$

i.e., we composite $\psi_{\hat{s}}$ with the true decoding function. We also define $\psi_{\hat{s}} \circ \alpha^{-1} : \mathcal{S}_h \to \mathcal{A}$, i.e., this policy acts on the true hidden state that it first maps a true hidden state to the corresponding learned state and then applies policy $\psi_{\hat{s}}$. Next, we let $\psi_s : \mathcal{S}_{h_1} \to \mathcal{A}$ be the policy that maximizes the reaching probability of s (based on the true transition dynamics) and define

$$\pi_s : \mathcal{X} \to \mathcal{A}, \pi_s(x_{h_1}) = \psi_s(f_h(x_{h_1})).$$

Note this is the policy that maximizes the reaching probability to s. We also define $\psi_s \circ \alpha : \widehat{\mathcal{S}}_h \to \mathcal{A}$, i.e., this policy acts on the learned hidden state that it first maps a learned hidden state to the corresponding true hidden state and then applies policy ψ_s.

We will use the following correspondence in conjunction with Lemma H.2 to do the analysis:

$$\begin{aligned}
\mathcal{S}_h &\Leftrightarrow \widehat{\mathcal{S}}_h, \\
\mathbf{p}_h &\Leftrightarrow \widehat{\mathbf{p}}_h, \\
\psi_{\hat{s}} \circ \alpha^{-1} &\Leftrightarrow \psi_{\hat{s}}, \\
\psi_s &\Leftrightarrow \psi_s \circ \alpha.
\end{aligned}$$

Now we can lower bound $\mathbb{P}^{\pi_{\hat{s}}}(s)$.

$$\begin{aligned}
\mathbb{P}^{\pi_{\hat{s}}}(s) \geq&\mathbb{P}^{\bar{\pi}_{\hat{s}}}(s) - 2H\epsilon_{\mathrm{f}} && \text{(Lemma G.5)} \\
=&\mathbb{P}^{\psi_{\hat{s}} \circ \alpha^{-1}}(s) - 2H\epsilon_{\mathrm{f}} && \text{(definition of } \bar{\pi}^{\hat{s}}\text{, probability refers to true hidden state dynamics)}
\end{aligned}$$

$$
\begin{aligned}
&\geq \hat{\mathbb{P}}^{\psi_{\hat{s}}}(\hat{s}) - 2H\epsilon_{\mathrm{f}} - H\epsilon_{\mathrm{p}} && \text{(Lemma H.2, probability refers to estimated transition probability)}\\
&\geq \hat{\mathbb{P}}^{\psi_s \circ \alpha}(\hat{s}) - 2H\epsilon_{\mathrm{f}} - H\epsilon_{\mathrm{p}} && (\psi_{\hat{s}} \text{ maximizes the probability to } \hat{s} \text{ w.r.t. } \hat{\mathbb{P}})\\
&\geq \mathbb{P}^{\psi_s}(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}} && \text{(Lemma H.2, probability refers to the true hidden state dynamics)}\\
&= \mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}. && \square
\end{aligned}
$$

In the following we prove Lemma G.2. We first collect some basically properties of the exploration policy η_h.

Lemma G.7. *If $\epsilon_{\mathrm{f}} \leq \frac{\mu_{\min}}{100H}$ and $\epsilon_{\mathrm{p}} \leq \frac{\mu_{\min}}{100H}$, we have $\mathbb{P}^{\eta_h}(\hat{s}) \geq \frac{\mu_{\min}}{2M}$ for any $\hat{s} \in \hat{\mathcal{S}}_{h-1}$.*

Proof of Lemma G.7. By Lemma G.1 we know $\mathbb{P}^{\hat{\pi}_{\hat{s}}}(s) \geq \mu_{\min} - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}$. Notice

$$
\begin{aligned}
\mathbb{P}^{\pi_{\hat{s}}}(\hat{s}) \geq& \mathbb{P}^{\pi_{\hat{s}}}(\hat{s}, s)\\
\geq& (\mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}})(1 - \epsilon_{\mathrm{f}})\\
\geq& (\mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}) \cdot 0.99.
\end{aligned}
$$

Since η_h uniformly samples from policies $\{\pi_{\hat{s}}\}_{\hat{s} \in \widehat{\mathcal{S}}_{h-1}}$, we have

$$
\mathbb{P}^{\eta_h}(\hat{s}) \geq \frac{(\mu(s) - 2H\epsilon_{\mathrm{f}} - 2H\epsilon_{\mathrm{p}}) \cdot 0.99}{M}.
$$

Lastly, plugging in the assumption on ϵ_{f} and ϵ_{p}, we prove the lemma. $\square$

Lemma G.8. *If $\epsilon_{\mathrm{f}} \leq \frac{\mu_{\min}}{100H}$ and $\epsilon_{\mathrm{p}} \leq \frac{\mu_{\min}}{100H}$, we have $\mathbb{P}^{\eta_h}(s') \geq \frac{\mu(s')}{2MK} \geq \frac{\mu_{\min}}{2MK}$ for any $s' \in \mathcal{S}_h$.*

Proof of Lemma G.8. By Lemma G.1 we know for any $s \in \mathcal{S}_{h-1}$, we have one policy $\pi_{\hat{s}}$ such that $\mathbb{P}^{\pi_{\hat{s}}}(s) \geq \frac{\mu(s)}{2}$ because ϵ_{f} and ϵ_{p} are sufficiently small. Since for η_h, we uniformly sample a state $\hat{s} \in \widehat{\mathcal{S}}_{h-1}$, we know for all state $s \in \mathcal{S}_{h-1}$, $\mathbb{P}^{\eta_h}(s) \geq \frac{\mu(s)}{2M}$. Thus because we uniformly sample actions, we have $\mathbb{P}^{\eta_h}(s, a) \geq \frac{\mu(s)}{2MK}$ for every $(s, a) \in \mathcal{S}_{h-1} \times \mathcal{A}$. Let $\pi_{s'}$ be that policy such that $\mathbb{P}^{\pi_{s'}} = \mu(s')$. Note we have

$$
\begin{aligned}
\mathbb{P}^{\eta_h}(s') =& \sum_{s \in \mathcal{S}_{h-1}, a \in \mathcal{A}} (s' \mid s, a) \mathbb{P}^{\eta_h}(s, a)\\
=& \sum_{s \in \mathcal{S}_{h-1}, a \in \mathcal{A}} p(s' \mid s, a) \mathbb{P}^{\pi_{s'}}(s, a) \cdot \frac{\mathbb{P}^{\eta_h}(s, a)}{\mathbb{P}^{\pi_{s'}}(s, a)}\\
\geq& \sum_{s \in \mathcal{S}_{h-1}, a \in \mathcal{A}} P(s' \mid s, a) \mathbb{P}^{\pi_{s'}}(s, a) \cdot \frac{\frac{\mu(s)}{2MK}}{\mu(s)}\\
=& \frac{\mu(s')}{2MK}\\
\geq& \frac{\mu_{\min}}{2MK}.
\end{aligned}
$$

$\square$

Now we ready to prove Lemma G.2.

Lemma G.9 (Restatement of Lemma G.2)**.** *Assume $\epsilon_{\mathrm{f}} \leq \frac{\mu_{\min}^3 \gamma}{100 M^4 K^3}$. Then the distributions $\hat{b}_{\nu_h}(\hat{s}, a | s')$ are well separated for any pair $s'_1, s'_2 \in \mathcal{S}_h$:*

$$
\left\| \hat{\mathbf{b}}_{\nu_h}(s'_1) - \hat{\mathbf{b}}_{\nu_h}(s'_2) \right\|_1 \geq \frac{\mu_{\min} \gamma}{3MK}.
$$

Furthermore, if $N_{\mathrm{g}} = \Omega\left(\frac{M^3 K^3}{\epsilon_{\mathrm{f}} \mu_{\min}^3 \gamma^2} \log\left(\frac{|\mathcal{G}| H}{\delta}\right)\right)$ we have with probability at least $1 - \delta/H$, for every $s' \in \mathcal{S}_h$, $\hat{\mathbf{g}}_h$ satisfies

$$
\mathbb{P}_{x' \sim q(\cdot | s')}\left[\left\| \hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s') \right\|_1 \geq \frac{\gamma \mu_{\min}}{100MK} \right] \leq \epsilon_{\mathrm{f}}.
$$

Proof of Lemma G.2. We first prove the property on $\hat{\mathbf{b}}_{\nu_h}$. First by Lemma G.7 and our definition of η_h, we know $\mathbb{P}^{\eta_h}(s,a) \geq \frac{\mu_{\min}}{2MK}$ for any $s \in \mathcal{S}_{h-1}$ and $a \in \mathcal{A}$. Recall

$$\mathbf{b}_{\nu_h}(s,a \mid s_1') = \frac{p_{h-1}(s_1' \mid s,a)\mathbb{P}^{\nu_h}(s,a)}{\sum_{s_1,a_1} p_{h-1}(s_1' \mid s_1,a_1)\mathbb{P}^{\nu_h}(s,a_1)}.$$

Invoking Lemma F.1, we have for any $s_1', s_2' \in \mathcal{S}_h$

$$\|\mathbf{b}_{\nu_h}(s_1') - \mathbf{b}_{\nu_h}(s_2')\|_1 \geq \frac{\mu_{\min}\gamma}{2MK}.$$

Next we show $\left\|\mathbf{b}_{\nu_h}(s') - \hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \leq \frac{\mu_{\min}\gamma}{6MK}$ for all $s' \in \mathcal{S}_h$. Note this implies the first part of the lemma. Consider a vector $\mathbf{Q}(s') \in \mathbb{R}^{|\mathcal{S}_{h-1}\times\mathcal{A}|}$ with each entry defined as

$$Q(s')_{(s,a)} = p_{h-1}(s' \mid s,a)\mathbb{P}^{\nu_h}(s,a).$$

Similarly we define $\widehat{Q}(s') \in \mathbb{R}^{|\mathcal{S}_{h-1}\times\mathcal{A}|}$ with each entry being

$$\widehat{Q}(s')_{(\hat{s},a)} = p_{h-1}^{\nu_h}(s' \mid \hat{s},a)\mathbb{P}^{\nu_h}(\hat{s},a).$$

It will be convenient to assume that entries in $Q(s')$ and $\widehat{Q}(s')$ are ordered such that the $\widehat{Q}(s')_{(\hat{s},a)}$ corresponds to $Q(s')_{(\alpha(\hat{s}),a)}$. Our strategy is to bound $\|\mathbf{Q}(s') - \widehat{\mathbf{Q}}(s')\|_1$, then invoke Lemma H.4 which gives the perturbation bound on the normalized vectors. We calculate the point-wise perturbation.

$$\begin{aligned}
&p_{h-1}(s' \mid \alpha(\hat{s}),a)\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a) - p_{h-1}^{\nu_h}(s' \mid \hat{s},a)\mathbb{P}^{\nu_h}(\hat{s},a) \\
=&\mathbb{P}(\alpha(\hat{s}),a)\left(p_{h-1}^{\nu_h}(s' \mid \alpha(\hat{s}),a) - p_{h-1}^{\nu_h}(s' \mid \hat{s},a)\right) + p_{h-1}^{\nu_h}(s' \mid \hat{s},a)\left(\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a) - \mathbb{P}^{\nu_h}(\hat{s},a)\right).
\end{aligned}$$

For the second term, we can directly bound

$$\begin{aligned}
|\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a) - \mathbb{P}^{\nu_h}(\hat{s},a)| =& \frac{1}{K}\left|\mathbb{P}^{\nu_h}(\hat{s}) - \mathbb{P}^{\nu_h}(\alpha(\hat{s}))\right| \\
=& \frac{1}{K}\left|\sum_{s_1\in\mathcal{S}_{h-1}} \mathbb{P}^{\nu_h}(\hat{s},s_1) - \sum_{\hat{s}_1\in\widehat{\mathcal{S}}_{h-1}} \mathbb{P}^{\nu_h}(\alpha(\hat{s}),\hat{s}_1)\right| \\
\leq& \frac{1}{K}\max\left\{\sum_{s_1\in\mathcal{S}_{h-1},s_1\neq\alpha(\hat{s})} \mathbb{P}^{\nu_h}(\hat{s},s_1), \sum_{\hat{s}_1\in\widehat{\mathcal{S}}_{h-1},\hat{s}_1\neq\hat{s}} \mathbb{P}^{\nu_h}(\alpha(\hat{s}),\hat{s}_1)\right\}
\end{aligned}$$

Note

$$\begin{aligned}
\sum_{s_1\in\mathcal{S}_{h-1},s_1\neq\alpha(\hat{s})} \mathbb{P}^{\nu_h}(\hat{s},s_1) =& \sum_{s_1\in\mathcal{S}_{h-1}} \mathbb{P}^{\nu_h}(s_1)\mathbb{P}_{x\sim q(\cdot|s_1)}\left[\hat{f}_h(x) = \hat{s}\right] \\
\leq& \sum_{s_1\in\mathcal{S}_{h-1}} \mathbb{P}^{\nu_h}(s_1)\epsilon_{\mathrm{f}} \\
\leq& \epsilon_{\mathrm{f}}
\end{aligned}$$

where the first inequality we used the induction hypothesis on the decoding error and the second inequality we used $\sum_{s_1\in\mathcal{S}_{h-1}} \mathbb{P}^{\nu_h}(s_1) \leq 1$. Similarly we can bound $\sum_{\hat{s}_1\in\widehat{\mathcal{S}}_{h-1},\hat{s}_1\neq\hat{s}} \mathbb{P}^{\nu_h}(\alpha(\hat{s}),\hat{s}_1) \leq \epsilon_{\mathrm{f}}$. Therefore, we have $|\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a) - \mathbb{P}^{\nu_h}(\hat{s},a)| \leq \frac{\epsilon_{\mathrm{f}}}{K}$. For the first term, note

$$p_{h-1}^{\nu_h}(s' \mid \alpha(\hat{s}),a) - p_{h-1}^{\nu_h}(s' \mid \hat{s},a) = \frac{\mathbb{P}^{\nu_h}(s',\hat{s},a)}{\mathbb{P}^{\nu_h}(\hat{s},a)} - \frac{\mathbb{P}^{\nu_h}(s',\alpha(\hat{s}),a)}{\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a)}.$$

We already have bound the deviation on the denominator.

$$|\mathbb{P}^{\nu_h}(s',\hat{s},a) - \mathbb{P}^{\nu_h}(s',\alpha(\hat{s}),a)| = \left|\sum_{s_1\in\mathcal{S}_{h-1}} \mathbb{P}^{\nu_h}(s',s_1,\hat{s},a) - \sum_{\hat{s}_1\in\widehat{\mathcal{S}}_{h-1}} \mathbb{P}^{\nu_h}(s',\hat{s}_1,\alpha(\hat{s}),a)\right|$$

$$
\begin{aligned}
&= \left| \sum_{s_1 \in \mathcal{S}_{h-1}, s_1 \neq \alpha(\hat{s})} \mathbb{P}^{\nu_h}(s', s_1, \hat{s}, a) - \sum_{\hat{s}_1 \in \widehat{\mathcal{S}}_{h-1}, \hat{s}_1 \neq \hat{s}} \mathbb{P}^{\nu_h}(s', \hat{s}_1, \alpha(\hat{s}), a) \right| \\
&\leq \max \left\{ \sum_{s_1 \in \mathcal{S}_{h-1}, s_1 \neq \alpha(\hat{s})} \mathbb{P}^{\nu_h}(s', s_1, \hat{s}, a), \sum_{\hat{s}_1 \in \widehat{\mathcal{S}}_{h-1}, \hat{s}_1 \neq \hat{s}} \mathbb{P}^{\nu_h}(s', \hat{s}_1, \alpha(\hat{s}), a) \right\} \\
&\leq \max \left\{ \sum_{s_1 \in \mathcal{S}_{h-1}, s_1 \neq \alpha(\hat{s})} \mathbb{P}^{\nu_h}(s_1, \hat{s}, a), \sum_{\hat{s}_1 \in \widehat{\mathcal{S}}_{h-1}, \hat{s}_1 \neq \hat{s}} \mathbb{P}^{\nu_h}(\hat{s}_1, \alpha(\hat{s}), a) \right\} \\
&= \frac{1}{K} \max \left\{ \sum_{s_1 \in \mathcal{S}_{h-1}, s_1 \neq \alpha(\hat{s})} \mathbb{P}^{\nu_h}(\hat{s}, s_1), \sum_{\hat{s}_1 \in \widehat{\mathcal{S}}_{h-1}, \hat{s}_1 \neq \hat{s}} \mathbb{P}^{\nu_h}(\alpha(\hat{s}), \hat{s}_1) \right\} \\
&\leq \frac{\epsilon_{\mathrm{f}}}{K}.
\end{aligned}
$$

Recall we have $\mathbb{P}^{\nu_h}(s,a) \geq \frac{\mu_{\min}}{2MK}$, so applying Lemma H.3 on $\frac{\mathbb{P}^{\nu_h}(s',\hat{s},a)}{\mathbb{P}^{\nu_h}(\hat{s},a)} - \frac{\mathbb{P}^{\nu_h}(s',\alpha(\hat{s}),a)}{\mathbb{P}^{\nu_h}(\alpha(\hat{s}),a)}$, we have

$$
\left| p_{h-1}^{\nu_h}(s_1' \mid \hat{s}, a) - p_{h-1}(s_1' \mid \alpha(\hat{s}), a) \right| \leq \frac{4M\epsilon_{\mathrm{f}}}{\mu_{\min}}. \tag{16}
$$

Therefore we have

$$
\left| p_{h-1}(s' \mid \alpha(\hat{s}), a) \mathbb{P}^{\nu_h}(\alpha(\hat{s}), a) - p_{h-1}^{\nu_h}(s' \mid \hat{s}, a) \mathbb{P}^{\nu_h}(\hat{s}, a) \right| \leq \frac{5M\epsilon_{\mathrm{f}}}{\mu_{\min}}.
$$

Thus we have

$$
\left\| \mathbf{Q}(s') - \widehat{\mathbf{Q}}(s') \right\|_1 \leq \frac{5M^2K\epsilon_{\mathrm{f}}}{\mu_{\min}}.
$$

By Lemma G.8, we know

$$
\left\| \mathbf{Q}(s') \right\|_1 = \sum_{(s,a) \in \mathcal{S}_{h-1} \times \mathcal{A}} p_{h-1}(s' \mid s, a) \mathbb{P}^{\nu_h}(s,a) = \mathbb{P}^{\eta_h}(s') \geq \frac{\mu_{\min}}{2MK}.
$$

Therefore applying Lemma H.4 on $\mathbf{Q}(s')$ and $\widehat{\mathbf{Q}}(s')$, we have

$$
\left\| \hat{\mathbf{b}}_{\nu_h}(s') - \mathbf{b}_{\nu_h}(s') \right\|_1 \leq \frac{100M^3K^2\epsilon_{\mathrm{f}}}{\mu_{\min}^2}.
$$

Since $\epsilon_{\mathrm{f}} \leq \frac{\mu_{\min}^3\gamma}{100M^4K^3}$, it follows that $\left\| \hat{\mathbf{b}}_{\nu_h}(s') - \mathbf{b}_{\nu_h}(s') \right\|_1 \leq \frac{\mu_{\min}\gamma}{6MK}$. Note that $\hat{\mathbf{b}}_{\nu_h}(s')$ is a conditional probability, we can apply the same arguments used in proving Theorem 3.1 to show

$$
\mathbf{g}_h(x') = \hat{\mathbf{b}}_{\nu}(s') \text{ for } x' \sim q(\cdot \mid s').
$$

Now we prove the second part of the Theorem about $\hat{\mathbf{g}}_h$. For simplicity, we set $\epsilon' = \frac{\mu_{\min}^3\gamma^2\epsilon_{\mathrm{f}}}{20000M^4K^4}$ in the following analysis. Using the same argument as Theorem 4.2, since we know $N_{\mathrm{g}} = \Omega\left(\frac{M^4K^4}{\epsilon_{\mathrm{f}}\mu_{\min}^3\gamma^2}\log(\frac{|\mathcal{G}|}{\delta})\right) = \Omega\left(\frac{1}{\epsilon'}\log\frac{|\mathcal{G}|}{\delta}\right)$, we have

$$
\mathbb{E}_{(\hat{s},a)\sim\nu_h, s'\sim\mathbf{p}^{\eta_h}(\cdot|\hat{s},a), x'\sim q(\cdot|s')} \left[\left\| \hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s') \right\|_2^2 \right] \leq \epsilon'
$$

Therefore, since we know by Lemma G.8 for any $s' \in \mathcal{S}_h$, $\mathbb{P}^{\eta_h}(s') \geq \frac{\mu_{\min}}{2MK}$, we have for all s'

$$
\mathbb{E}_{x'\sim q(\cdot|s')} \left[\left\| \hat{\mathbf{g}}_h(x') - \hat{\mathbf{b}}_{\nu_h}(s') \right\|_2^2 \right] \leq \frac{2MK\epsilon'}{\mu_{\min}}.
$$

By Markov's inequality, we have

$$\mathbb{P}_{x'\sim q(\cdot|s')}\left(\left\|\hat{\mathbf{g}}_h(x')-\hat{\mathbf{b}}_{\nu_h}(s')\right\|_2^2 \geq \frac{\gamma^2\mu_{\min}^2}{10000M^3K^3}\right) \leq \frac{20000M^4K^4\epsilon'}{\mu_{\min}^3\gamma^2} \leq \epsilon_{\mathrm{f}}$$

Using the fact that $\|\cdot\|_1 \leq \sqrt{MK}\|\cdot\|_2$, we have

$$\mathbb{P}_{x'\sim q(\cdot|s')}\left(\left\|\hat{\mathbf{g}}_h(x')-\hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \geq \frac{\gamma\mu_{\min}}{100MK}\right) \leq \epsilon_{\mathrm{f}}.$$

□

Lemma G.10 (Restatement of Lemma G.3)**.** *If $N_\phi = \Theta\left(\frac{MK}{\mu_{\min}}\log(\frac{MH}{\delta})\right)$ and $\epsilon_{\mathrm{f}} \leq \frac{\delta}{100HN_\phi}$ we have with probability at least $1-\frac{\delta}{H}$, (1) for every $s'\in\mathcal{S}_h$, there exists at least one point $\mathbf{z}\in\mathcal{Z}$ such that $\mathbf{z}=\hat{\mathbf{g}}_h(x')$ with $x'\sim q(\cdot\mid s')$ and $\left\|\hat{\mathbf{g}}_h(x')-\hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \leq \frac{\mu_{\min}\gamma}{100MK}$ and (2) for every $\mathbf{z}=\hat{g}_h(x')\in\mathcal{Z}$ with $x'\sim q(\cdot\mid s')$, $\left\|\hat{\mathbf{g}}_h(x')-\hat{\mathbf{b}}_{\nu_h}(s')\right\|_1 \leq \frac{\mu_{\min}\gamma}{100MK}$.*

Proof of Lemma G.3. For any state $s'\in\mathcal{S}_h$, by Lemma G.8 we know $\mathbb{P}^{\eta_h}(s') \geq \frac{\mu_{\min}}{2MK}$. The probability of not seeing one context generated from this state is upper bounded by $\left(1-\frac{\mu_{\min}}{2MK}\right)^{N_\phi} \leq \frac{\delta}{2MH}$. Now taking union bound over $\mathcal{S}_h$, we know with probability at least $1-\frac{\delta}{2H}$, we get one context from every state. Furthermore, because we know $\epsilon_{\mathrm{f}} \leq \frac{\delta}{100HN_\phi}$, by union bound over N_ϕ samples, we know we can decode every context correctly with probability at least $1-\frac{\delta}{2H}$. □

Lemma G.11 (Restatement of Lemma G.4)**.** *If $\epsilon_{\mathrm{f}} \leq \frac{\epsilon_{\mathrm{p}}\mu_{\min}}{10M^2}$ and if $N_{\mathrm{p}} = \Omega\left(\frac{M^2K}{\epsilon_{\mathrm{p}}^2}\log\frac{MHK}{\delta}\right)$, we have that with probability at least $1-\frac{\delta}{H}$ for every $\hat{s}\in\widehat{\mathcal{S}}_{h-1}$, $a\in\mathcal{A}$*

$$\|\widehat{\mathbf{p}}(\hat{s},a)-\mathbf{p}(\alpha_{h-1}(\hat{s}),a)\|_1 \leq \epsilon_{\mathrm{p}}, \tag{17}$$

Proof of Lemma G.4. Using Equation (16) and the decoding error bound on $\hat{f}_h$, we know for any $\hat{s}\in\widehat{\mathcal{S}}_{h-1}, a\in\mathcal{A}, \hat{s}'\in\widehat{\mathcal{S}}_{h-1}$, we have

$$|p^{\eta_h}(\hat{s}'\mid\hat{s},a)-p(\alpha_h(\hat{s}')\mid\alpha_{h-1}(\hat{s}),a)| \leq \frac{4M\epsilon_{\mathrm{f}}}{\mu_{\min}}.$$

Summing over $\widehat{\mathcal{S}}_h$, we have

$$\sum_{\hat{s}'\in\widehat{\mathcal{S}}_h}|p^{\eta_h}(\hat{s}'\mid\hat{s},a)-p(\alpha_h(\hat{s}')\mid\alpha_{h-1}(\hat{s}),a)| \leq \frac{4M^2\epsilon_{\mathrm{f}}}{\mu_{\min}}.$$

Next we bound $\|\widehat{\mathbf{p}}^{\eta_h}(\hat{s},a)-\mathbf{p}^{\eta_h}(\hat{s},a)\|_1$. By Lemma G.7, we know for every $(\hat{s},a)\in\widehat{\mathcal{S}}_{h-1}\times\mathcal{A}$, $\mathbb{P}^{\eta_h}(\hat{s},a) \geq \frac{\mu_{\min}}{2MK}$. For each pair, by Theorem I.1, we need $\Omega\left(\frac{M}{\epsilon_{\mathrm{p}}^2}\right)$ samples. Thus in total we need $N_p = \Omega\left(\frac{M^2K}{\mu_{\min}\epsilon_{\mathrm{p}}^2}\log\frac{MHK}{\delta}\right)$ to make $\|\widehat{\mathbf{p}}(\hat{s},a)-\mathbf{p}^{\eta_h}(\hat{s},a)\|_1 \leq \frac{\epsilon_{\mathrm{p}}}{10}$. Now combining these two inequalities we have the desired result. □

H. Technical Lemmas

Lemma H.1. *For any two vectors $u,v\in\mathbb{R}_+^d$ with $\|u\|_1 = \|v\| = 1$ and $\|u-v\|_1 = \gamma$, we have for any $\alpha > 0$, $\|\alpha u - v\|_1 \geq \frac{\gamma}{2}$.*

Proof of Lemma H.1. Denote $S_+ = \{i\in[d]|u_i > v_i\}$ and $S_- = \{i\in[d]|u_i < v_i\}$. Because $\|u-v\|_1 = \gamma$, we know

$$\sum_{i\in S_+}(u_i-v_i)+\sum_{i\in S_-}(v_i-u_i) = \gamma.$$

Also note that

$$\sum_{i\in S_+}(u_i-v_i)-\sum_{i\in S_-}(v_i-u_i) = \|u\|_1-\|v\|_1 = 0.$$

Therefore,

$$\sum_{i\in S_+}(u_i - v_i) = \sum_{i\in S_-}(v_i - u_i) = \frac{\gamma}{2}.$$

If $\alpha \geq 1$, we know

$$\|\alpha u - v\|_1 \geq \sum_{i\in S_+} \alpha u_i - v_i \geq \frac{\gamma}{2}$$

and if $\alpha < 1$, we know

$$\|\alpha u - v\|_1 \geq \sum_{i\in S_+} v_i - \alpha u_i \geq \frac{\gamma}{2}.$$

We finish the proof. □

Lemma H.2. *[Error Propagation Lemma for Tabular MDPs] Consider two tabular MDPs, $\mathcal{M}$ and $\widehat{\mathcal{M}}$. Let $\mathcal{S}_1, \ldots, \mathcal{S}_H$ be the state space of $\mathcal{M}$ and $\widehat{\mathcal{S}}_1, \ldots, \widehat{\mathcal{S}}_H$ be the for $\widehat{\mathcal{M}}$. The state spaces satisfy that for every $h \in [H]$, $\mathcal{S}_h$ and $\widehat{\mathcal{S}}_h$ are bijective, i.e., there exists a bijective function $\alpha : \widehat{\mathcal{S}}_h \to \mathcal{S}_h$. Let $\mathcal{A}$ be $\mathcal{M}$ and $\widehat{\mathcal{M}}$'s shared action space. For $h = 1, \ldots, H$, let $\mathbf{p}_h$ be the forward operator for $\mathcal{M}$ and $\widehat{\mathbf{p}}_h$ be the forward operator model $\mathcal{M}$. For any policy on $\psi : \mathcal{S}_h \to \mathcal{A}$ for $\mathcal{M}$, because the $\mathcal{S}_h$ and $\widehat{\mathcal{S}}_h$ are bijective, ψ induces a policy for $\widehat{\mathcal{M}}$, $\hat{\psi} : \widehat{\mathcal{S}}_h \to \mathcal{A}$ that satisfies $\psi(\alpha_h(\hat{s})) = \hat{\psi}(\hat{s})$. Then if*

$$\|\widehat{\mathbf{p}}_h(\hat{s}, a) - \mathbf{p}_h(\alpha(\hat{s}), a)\|_1 \leq \epsilon$$

for all $h \in [H]$, $a \in \mathcal{A}$ and $\hat{s} \in \widehat{\mathcal{S}}_h$ (the indices of the vector $\widehat{\mathbf{p}}_h(\hat{s}, a)$ and $\widehat{\mathbf{p}}_h(\alpha(\hat{s}), a)$ are matched according to α), we have for any policy ψ for $\mathcal{M}$,

$$\sum_{s_h\in\mathcal{S}_h} \left|\hat{\mathbb{P}}_h^{\hat{\psi}}(\alpha^{-1}(s_h)) - \mathbb{P}_h^{\psi}(s_h)\right| \leq h\epsilon$$

Proof of Lemma H.2. We prove by induction.

$$\begin{aligned}
&\sum_{s_h\in\mathcal{S}_h} \left|\hat{\mathbb{P}}_h^{\psi}(\alpha^{-1}(s_h)) - \mathbb{P}_h^{\psi}(s_h)\right| \\
&= \sum_{s_h\in\mathcal{S}_h} \left| \sum_{s_{h-1}\in\mathcal{S}_{h-1}} \left(\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1}))\hat{p}_{h-1}(\alpha^{-1}(s_h) \mid \alpha^{-1}(s_{h-1}), \hat{\psi}(\alpha^{-1}(s_{h-1}))) - \mathbb{P}^{\psi}(s_{h-1})p(s_h \mid s_{h-1}, \psi(s_{h-1}))\right)\right| \\
&\leq \sum_{s_h\in\mathcal{S}_h} \left| \sum_{s_{h-1}\in\mathcal{S}_{h-1}} (\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1})) - \mathbb{P}^{\psi}(s_{h-1}))p(s_h \mid s_{h-1}, \psi(s_{h-1}))\right| \\
&\quad + \sum_{s_h\in\mathcal{S}_h} \sum_{s_{h-1}\in\mathcal{S}_{h-1}} \widehat{\mathbb{P}}(\alpha^{-1}(s_{h-1})) \left|\hat{p}(\alpha^{-1}(s_h) \mid \alpha^{-1}(s_{h-1}), \hat{\psi}(\alpha^{-1}(s_{h-1}))) - p(s_h \mid s_{h-1}, \psi(s_{h-1}))\right|
\end{aligned}$$

For the first term,

$$\begin{aligned}
&\sum_{s_h\in\mathcal{S}_h} \left| \sum_{s_{h-1}\in\mathcal{S}_{h-1}} (\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1})) - \mathbb{P}^{\psi}(s_{h-1}))p(s_h \mid s_{h-1}, \psi(s_{h-1}))\right| \\
&\leq \sum_{s_h\in\mathcal{S}_h} \sum_{s_{h-1}\in\mathcal{S}_{h-1}} \left|\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1})) - \mathbb{P}^{\psi}(s_{h-1})\right| p(s_h \mid s_{h-1}, \psi(s_{h-1})) \\
&= \sum_{s_{h-1}\in\mathcal{S}_{h-1}} \left(\left|\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1})) - \mathbb{P}^{\psi}(s_{h-1})\right| \Big(\sum_{s_h\in\mathcal{S}_h} p(s_h \mid s_{h-1}, \psi(s_{h-1}))\Big)\right) \\
&= \sum_{s_{h-1}\in\mathcal{S}_{h-1}} \left|\widehat{\mathbb{P}}^{\hat{\psi}}(\alpha^{-1}(s_{h-1})) - \mathbb{P}^{\psi}(s_{h-1})\right| \qquad \text{(transition probabilities sum up to 1)}
\end{aligned}$$

$$\leq (h-1)\epsilon. \qquad \text{(induction hypothesis)}$$

For the other term,

$$\begin{aligned}
&\sum_{s_h \in \mathcal{S}_h} \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \widehat{\mathbb{P}}(\alpha^{-1}(s_{h-1})) \left| \hat{p}(\alpha^{-1}(s_h) \mid \alpha^{-1}(s_{h-1}), \hat{\psi}(\alpha^{-1}(s_{h-1}))) - p(s_h \mid s_{h-1}, \psi(s_{h-1})) \right| \\
&= \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \widehat{\mathbb{P}}(\alpha^{-1}(s_{h-1})) \sum_{s_h \in \mathcal{S}_h} \left| \hat{p}(\alpha^{-1}(s_h) \mid \alpha^{-1}(s_{h-1}), \hat{\psi}(\alpha^{-1}(s_{h-1}))) - p(s_h \mid s_{h-1}, \psi(s_{h-1})) \right| \\
&= \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \widehat{\mathbb{P}}(\alpha^{-1}(s_{h-1})) \left\| \widehat{\mathbf{p}}(\alpha^{-1}(s_{h-1}), \hat{\psi}(\alpha^{-1}(s_{h-1}))) - \mathbf{p}(s_{h-1}, \psi(s_{h-1})) \right\|_1 \\
&\leq \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \widehat{\mathbb{P}}(\alpha^{-1}(s_{h-1}))\, \epsilon = \epsilon.
\end{aligned}$$

Combining these two inequalities we have the desired result. □

Lemma H.3. *For $a, b, c, d \in \mathbb{R}^+$ with $a \leq b$ and $c \leq d$, we have*

$$\left| \frac{a}{b} - \frac{c}{d} \right| \leq \frac{|d-b| + |a-c|}{\max\{b, d\}}.$$

Proof of Lemma H.4.

$$\begin{aligned}
\left| \frac{a}{b} - \frac{c}{d} \right| &= \left| \frac{ad - bc}{bd} \right| \\
&= \left| \frac{ad - ab + ab - bc}{bd} \right| \\
&= \left| \frac{a(d-b)}{bd} + \frac{a-c}{d} \right| \\
&\leq \frac{|d-b| + |a-c|}{d}.
\end{aligned}$$

By symmetry between b and d, we obtain the desired result. □

Lemma H.4. *For any two vector $\mathbf{p}, \mathbf{q} \in \mathbb{R}^d_+$, we have*

$$\left\| \frac{\mathbf{p}}{\|\mathbf{p}\|_1} - \frac{\mathbf{q}}{\|\mathbf{q}\|_1} \right\|_1 \leq \frac{2\|\mathbf{p} - \mathbf{q}\|_1}{\max\{\|\mathbf{p}\|_1, \|\mathbf{q}\|_1\}}.$$

Proof of Lemma H.4.

$$\begin{aligned}
\left\| \frac{\mathbf{p}}{\|\mathbf{p}\|_1} - \frac{\mathbf{q}}{\|\mathbf{q}\|_1} \right\|_1 &= \left\| \frac{\mathbf{p}\|\mathbf{q}\|_1 - \mathbf{q}\|\mathbf{p}\|_1}{\|\mathbf{p}\|_1 \|\mathbf{q}\|_1} \right\|_1 \\
&= \left\| \frac{\mathbf{p}\|\mathbf{q}\|_1 - \mathbf{q}\|\mathbf{q}\|_1 + \mathbf{q}\|\mathbf{q}\|_1 - \mathbf{q}\|\mathbf{p}\|_1}{\|\mathbf{p}\|_1 \|\mathbf{q}\|_1} \right\|_1 \\
&\leq \frac{\|\mathbf{p} - \mathbf{q}\|_1}{\|\mathbf{p}\|_1} + \frac{|\|\mathbf{p}\|_1 - \|\mathbf{q}\|_1|}{\|\mathbf{p}\|_1} \\
&\leq \frac{2\|\mathbf{p} - \mathbf{q}\|_1}{\|\mathbf{p}\|_1}.
\end{aligned}$$

By symmetry between $\mathbf{p}$ and $\mathbf{q}$, we obtain the desired result. □

Lemma H.5 (Perturbation of Point-wise Division Around Uniform Distribution)**.** *For any two vector $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^d_+$, we have*

$$\left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2}{\|\mathbf{p}_1 \oslash \mathbf{p}_2\|_1} - \begin{pmatrix} 1/d \\ \dots \\ 1/d \end{pmatrix} \right\|_1 \leq \frac{2\|\mathbf{p}_1 - \mathbf{p}_2\|_1}{d \min_s \mathbf{p}_2(s)}.$$

where $\oslash$ denotes pointwise division.

Proof of Lemma H.5. Let $d' = \|\mathbf{p}_1 \oslash \mathbf{p}_2\|_1$ and $\mathbf{1}$ be the all one vector of dimension d.

$$\begin{aligned}\left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2}{\|\mathbf{p}_1 \oslash \mathbf{p}_2\|_1} - \begin{pmatrix} 1/d \\ \dots \\ 1/d \end{pmatrix} \right\|_1 &= \left\| \frac{d\mathbf{p}_1 \oslash p_2 - d'\mathbf{1}}{d'd} \right\|_1 \\ &\le \left\| \frac{d\mathbf{p}_1 \oslash \mathbf{p}_2 - d'\mathbf{p}_1 \oslash \mathbf{p}_2}{dd'} \right\|_1 + \left\| \frac{d'\mathbf{p}_1 \oslash \mathbf{p}_2 - d'\mathbf{1}}{d'd} \right\|_1 \\ &= |d - d'| \frac{\|\mathbf{p}_1 \oslash \mathbf{p}_2\|_1}{dd'} + \left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2 - \mathbf{1}}{d} \right\|_1 \\ &= \frac{|d - d'|}{d} + \left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2 - \mathbf{1}}{d} \right\|_1 .\end{aligned}$$

Note for any $s \in [d]$, we have

$$\left| \frac{p_1(s)}{p_2(s)} - 1 \right| = \frac{|p_1(s) - p_2(s)|}{p_2(s)} \le \frac{|p_1(s) - p_2(s)|}{\min_{s_1 \in [d]} p_2(s_1)}.$$

Therefore, we have

$$\left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2 - \mathbf{1}}{d} \right\|_1 \le \frac{\sum_s |p_1(s) - p_2(s)|}{d \min_s p_2(s)} = \frac{\|\mathbf{p}_1 - \mathbf{p}_2\|_1}{d \min_s p_2(s)}.$$

Also note that, $\frac{|d'-d|}{d} = \frac{|\|\mathbf{p}_1 \oslash \mathbf{p}_2\|_1 - \|\mathbf{1}\|_1|}{d} \le \left\| \frac{\mathbf{p}_1 \oslash \mathbf{p}_2 - \mathbf{1}}{d} \right\|_1 \le \frac{\|\mathbf{p}_1 - \mathbf{p}_2\|_1}{d \min_s p_2(s)}$. Plugging in these two bounds we obtain our desired result. □

Lemma H.6 (Conditional Probability Perturbation Around Uniform Distribution)**.** *Let $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^d_+$ with $\|\mathbf{p}_1\|_1 = \|\mathbf{p}_2\|_1 = 1$ and $\mathbf{p}_2 = (1/d, \dots, 1/d)^\top$. Then for any $\mathbf{q} \in \mathbb{R}^d_+$ we have*

$$\left\| \frac{\mathbf{q} \odot \mathbf{p}_1}{\mathbf{q}^\top \mathbf{p}_1} - \frac{\mathbf{q} \odot \mathbf{p}_2}{\mathbf{q}^\top \mathbf{p}_2} \right\|_1 \le 2d \|\mathbf{p}_1 - \mathbf{p}_2\|_1$$

where $\odot$ represents point-wise product.

Proof of Lemma H.6. Note the left hand size is independent of the scale of $\mathbf{q}$, so without loss of generality we assume $\|\mathbf{q}\|_1 = 1$. We calculate the quantity of interest.

$$\begin{aligned}\frac{\mathbf{q} \odot \mathbf{p}_1}{\mathbf{q}^\top \mathbf{p}_1} - \frac{\mathbf{q} \odot \mathbf{p}_2}{\mathbf{q}^\top \mathbf{p}_2} &= \frac{\mathbf{q} \odot \mathbf{p}_1 (\mathbf{q}^\top \mathbf{p}_2) - \mathbf{q} \odot \mathbf{p}_2 (\mathbf{q}^\top \mathbf{p}_1)}{\mathbf{q}^\top \mathbf{p}_1 \cdot \mathbf{q}^\top \mathbf{p}_2} \\ &= \frac{\mathbf{q} \odot \mathbf{p}_1 (\mathbf{q}^\top \mathbf{p}_2 - \mathbf{q}^\top \mathbf{p}_1) + \mathbf{q}^\top \mathbf{p}_1 (\mathbf{q} \odot \mathbf{p}_1 - \mathbf{q} \odot \mathbf{p}_2)}{\mathbf{q}^\top \mathbf{p}_1 \cdot \mathbf{q}^\top \mathbf{p}_2}.\end{aligned}$$

By Hölder inequality, we have $\left|\mathbf{q}^\top \mathbf{p}_1 - \mathbf{q}^\top \mathbf{p}_2\right| \le \|\mathbf{q}\|_\infty \|\mathbf{p}_1 - \mathbf{p}_2\|_1$. Furthermore, note $\|\mathbf{q} \odot \mathbf{p}_1\|_1 = \mathbf{q}^\top \mathbf{p}_1$ because of the positivity and $\mathbf{q}^\top \mathbf{p}_2 = \frac{1}{d}$ because $\mathbf{p}_2$ is a uniform distribution. Now we can bound

$$\left\| \frac{\mathbf{q} \odot \mathbf{p}_1 (\mathbf{q}^\top \mathbf{p}_2 - \mathbf{q}^\top \mathbf{p}_1)}{\mathbf{q}^\top \mathbf{p}_1 \cdot \mathbf{q}^\top \mathbf{p}_2} \right\|_1 \le \frac{\|\mathbf{q}\|_\infty \|\mathbf{p}_1 - \mathbf{p}_2\|_1}{1/d} \le d \|\mathbf{p}_1 - \mathbf{p}_2\|_1 .$$

Next, apply Hölder inequality again, we have $\|\mathbf{q} \odot \mathbf{p}_1 - \mathbf{q} \odot \mathbf{p}_2\|_1 \le \|\mathbf{q}\|_\infty \|\mathbf{p}_1 - \mathbf{p}_2\|_1$. Therefore we can bound

$$\left\| \frac{\mathbf{q}^\top \mathbf{p}_1 (\mathbf{q} \odot \mathbf{p}_1 - \mathbf{q} \odot \mathbf{p}_2)}{\mathbf{q}^\top \mathbf{p}_1 \cdot \mathbf{q}^\top \mathbf{p}_2} \right\|_1 \le \frac{\|\mathbf{q}\|_\infty \|\mathbf{p}_1 - \mathbf{p}_2\|_1}{1/d} \le d \|\mathbf{p}_1 - \mathbf{p}_2\|_1 .$$

□

I. Concentration Inequalities

Theorem I.1 (L_1 distance concentration bound (Theorem 2.2 of (Weissman et al., 2003))). *Let p be a distribution over $\mathcal{A}$ with $|\mathcal{A}| = a$. Let $X_1, \ldots, X_m \sim p$ and $\hat{p}_{X^m}$ be the empirical distribution. Then we have*

$$\mathbb{P}\left(\|p - \hat{p}_{X^m}\|_1 \geq \epsilon\right) \leq (2^a - 2) \exp\left(-\frac{m\epsilon^2}{8}\right).$$

A directly corollary is the following sample complexity.

Corollary I.1. *if we have $m \geq 8\left(\frac{a}{\epsilon^2} \log \frac{1}{\delta}\right)$ samples, then with probability at least $1 - \delta$, we have $\|p - \hat{p}_{X^m}\|_1 \geq \epsilon$.*