Kinematic State Abstraction and Provably Efficient Rich-Observation Reinforcement Learning

Dipendra Misra, Mikael Henaff, Akshay Krishnamurthy, and John Langford

Microsoft Research, New York, NY

Abstract

We present an algorithm, HOMER, for exploration and reinforcement learning in rich observation environments that are summarizable by an unknown latent state space. The algorithm interleaves representation learning to identify a new notion of *kinematic state abstraction* with strategic exploration to reach new states using the learned abstraction. The algorithm provably explores the environment with sample complexity scaling polynomially in the number of latent states and the time horizon, and, crucially, with no dependence on the size of the observation space, which could be infinitely large. This exploration guarantee further enables sample-efficient global policy optimization for any reward function. On the computational side, we show that the algorithm can be implemented efficiently whenever certain supervised learning problems are tractable. Empirically, we evaluate HOMER on a challenging exploration problem, where we show that the algorithm is exponentially more sample efficient than standard reinforcement learning baselines.

1 Introduction

Modern reinforcement learning applications call for agents that operate directly from rich sensory information such as megapixel camera images. This rich information enables representation of detailed, high-quality policies and obviates the need for hand-engineered features. However, exploration in rich observation environments constitutes a fundamental challenge which, in general, is statistically intractable [21, 32, 30]. Despite this, many environments are highly structured and do admit sample efficient algorithms [23]; indeed, we may be able to summarize the environment with a simple state space and extract these states from raw observations. With such structure, we can leverage techniques from the well-studied tabular setting to explore the environment [19], efficiently recover the underlying dynamics [47], and optimize any reward function [29, 10, 48, 14, 5, 24]. But can we learn to decode a simpler state from raw observations alone?

The main difficulty is that learning a state decoder, or a compact representation, is intrinsically coupled with exploration. On one hand, we cannot learn a high-quality decoder without gathering comprehensive information from the environment, which may require a sophisticated exploration strategy. On the other hand, we cannot tractably explore the environment without an accurate decoder. These interlocking problems constitute a central challenge in reinforcement learning, and a provably effective solution remains elusive despite decades of research [36, 42, 26, 33, 7, 40] (See Section 7 for a discussion of related work).

In this paper, we provide a solution for a significant sub-class of problems known as Block Markov Decision Processes (MDPs) [16], in which the agent operates directly on rich observations that are generated from a small number of unobserved latent states. Our algorithm, HOMER, learns a new reward-free state

{dimisra, mihenaff, akshaykr, jcl}@microsoft.com

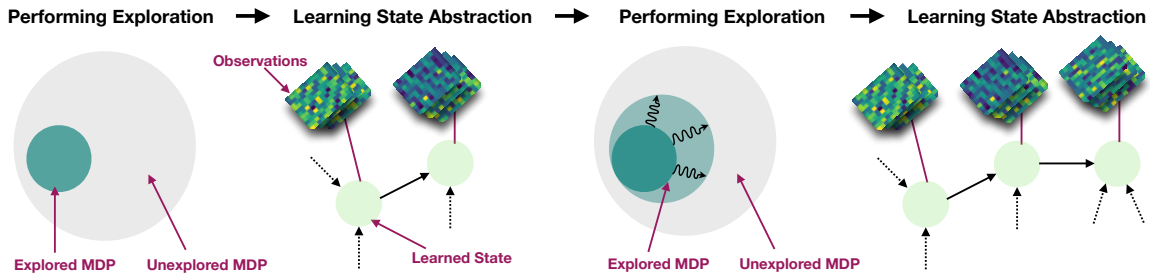


Figure 1: We introduce an algorithm HOMER that learns a set of exploration policies and a state abstraction function. The algorithm iterates between the learning to explore using existing state abstraction and refining the state abstraction based on exploration.

abstraction called *kinematic inseparability*, which it uses to drive exploration of the environment. Informally, kinematic inseparability aggregates observations that have the same forward and backward dynamics. Shared backward dynamics crucially implies that a single policy simultaneously maximizes the probability of observing a set of kinematically inseparable observations, which is useful for exploration. Shared forward dynamics is naturally useful for recovering the latent state space and model. Perhaps most importantly, we show that a kinematic inseparability abstraction can be recovered from a bottleneck in a regressor trained on a contrastive estimation problem derived from raw observations.

HOMER performs strategic exploration by training policies to visit each kinematically inseparable abstract state, resulting in a *policy cover*. These policies are constructed via a reduction to contextual bandits [6], using a dynamic-programming approach and a synthetic reward function that incentivizes reaching an abstract state. Crucially HOMER interleaves learning the state abstraction and policy cover in an inductive manner; we use the policies learned from a coarse abstraction to reach new states, which enables us to refine the state abstraction and learn new policies (See Figure 1 for a schematic). Each process is essential to the other. Once the policy cover is constructed, it can be used to efficiently gather the information necessary to find a near-optimal policy for any reward function.

We analyze the statistical and computational properties of HOMER in episodic Block MDPs. We prove that HOMER learns to visit every latent state and also learns a near-optimal policy for any given reward function with a number of trajectories that is polynomial in the number of latent states, actions, horizon, and the complexity of two function classes used by the algorithm. Crucially there is no explicit dependence on the observation space size. The main assumptions are that the latent states are reachable and that the function classes are sufficiently expressive. There are no identifiability or determinism assumptions beyond decodability of the Block MDP, resulting in significantly greater scope than prior work [16, 15]. On the computational side, HOMER operates in a reductions model and can be implemented efficiently whenever cetain supervised learning problems are tractable.

Empirically, we evaluate HOMER on a challenging RL problem with high-dimensional observations, precarious dynamics, and sparse, misleading rewards. The problem is googal-sparse: the probability of encountering an optimal reward through random search is 10^{-100}. HOMER recovers the underlying state abstraction for this problem and consistently finds a near-optimal policy, outperforming popular RL baselines which use naive exploration strategies [37, 44] or more sophisticated exploration bonuses [11], as well as the recent PAC-RL algorithm of [16].

2 Preliminaries

We consider reinforcement learning (RL) in a generalized episodic Block Markov Decision Processes (Block MDP) setting, first introduced by Du et al. [16]. A Block MDP $\mathcal{M}$ is described by a large (possibly infinite) observation space $\mathcal{X}$, a finite unobservable state space $\mathcal{S}$, a finite set of actions $\mathcal{A}$, and a time horizon $H \in \mathbb{N}$. The process has a starting state distribution $\mu \in \Delta(\mathcal{S})$[1], transition function $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, emission function $q : \mathcal{S} \to \Delta(\mathcal{X})$, and a reward function $R : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \Delta([0, 1])$. The agent interacts with the environment by repeatedly generating H-step trajectories $(s_1, x_1, a_1, r_1, \ldots, s_H, x_H, a_H, r_H)$ where $s_1 \sim \mu$, $s_{h+1} \sim T(\cdot|s_h, a_h)$, $x_h \sim q(s_h)$ and $r_h \sim r(x_h, a_h, x_{h+1})$ for all $h \in [H]$, and all actions are chosen by the agent. We set $R(x_H, a_H, x_{H+1}) = R(x_H, a_H)$ for all x_H, a_H as there is no x_{H+1}. We assume without loss of generality that for any trajectory $\sum_{h=1}^{H} r_h \leq 1$. The agent *does not* observe the states $s_1, \ldots, s_H$. As notation, we often denote sequences using the ":" operator, e.g., $s_{1:H} = (s_1, \ldots, s_H)$.

Without loss of generality, we partition $\mathcal{S}$ into subsets $\mathcal{S}_1, \ldots, \mathcal{S}_H$, where $\mathcal{S}_h$ are the only states reachable at time step h. We similarly partition $\mathcal{X}$ based on time step into $\mathcal{X}_1, \ldots, \mathcal{X}_H$. Formally, $T(\cdot \mid s, a) \in \Delta(\mathcal{S}_{h+1})$ and $q(s) \in \Delta(\mathcal{X}_h)$ when $s \in \mathcal{S}_h$. This partitioning may be internal to the agent as we can simply concatenate the time step to the states and observations. Let $\tau : \mathcal{X} \to [H]$ be the time step function, associating an observation to the time point where it is reachable.

A *policy* $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ chooses actions on the basis of observations and defines a distribution over trajectories. We use $\mathbb{E}_\pi[\cdot], \mathbb{P}_\pi[\cdot]$ to denote expectation and probability with respect to this distribution. The goal of the agent is to find a policy that achieves high expected reward. We define the value function and policy value as:

$$\forall h \in [H], s \in \mathcal{S}_h : \; V(s; \pi) := \mathbb{E}_\pi \left[\sum_{h'=h}^{H} r_{h'} \mid s_h = s \right], \qquad V(\pi) := \mathbb{E}_\pi \left[\sum_{h=1}^{H} r_h \right] = \mathbb{E}_{s_1 \sim \mu} \left[V(s_1; \pi) \right].$$

As the observation space is extremely large, we consider a function approximation setting, where the agent has access to a policy class $\Pi : \mathcal{X} \to \Delta(\mathcal{A})$. We further define the class of non-stationary policies $\Pi_{\mathrm{NS}} := \Pi^H$ to enable the agent to use a different policy for each time step: a policy $\pi = \pi_{1:H} = (\pi_1, \ldots, \pi_H) \in \Pi_{\mathrm{NS}}$ takes action a_h according to π_h.[2] The optimal policy in this class is $\pi^\star := \operatorname{argmax}_{\pi \in \Pi_{\mathrm{NS}}} V(\pi)$, and our goal is to find a policy with value close to the optimal value, $V(\pi^\star)$.

Environment assumptions. The key difference between Block MDPs and general Partially-Observed MDPs is a disjointness assumption, which removes partial observability effects and enables tractable learning.

Assumption 1. *The emission distributions for any two states $s, s' \in \mathcal{S}$ are disjoint, that is supp$(q(s)) \cap$ supp$(q(s')) = \emptyset$ whenever $s \neq s'$.*

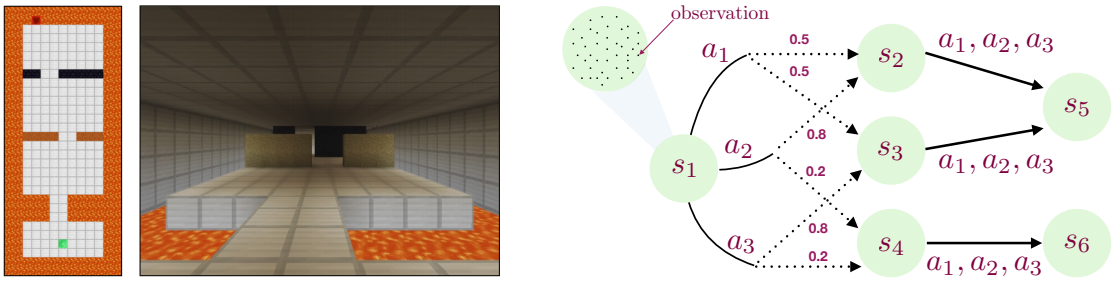
This disjointness assumption is used by Du et al. [16], who argue that it is a natural fit for visual grid-world scenarios such as in Figure 2a, which are common in empirical RL research. The name "Block MDP" arises since each hidden state s emits observations from a disjoint block $\mathcal{X}_s \subseteq \mathcal{X}$. The assumption allows us to define an *inverse mapping* $g^\star : \mathcal{X} \to \mathcal{S}$ such that for each $s \in \mathcal{S}$ and $x \in \mathrm{supp}(q(s))$, we have $g^\star(x) = s$. The agent *does not* have access to $g^\star$.

Apart from disjointness, the main environment assumption is that states are reachable with reasonable probability. To formalize this, we define a *maximum visitation probability* and *reachability parameter*:

$$\eta(s) := \max_{\pi \in \Pi_{\mathrm{NS}}} \mathbb{P}_\pi \left[s \right], \qquad \eta_{min} = \min_{s \in \mathcal{S}} \eta(s)$$

[1] Du et al. [16] assume the starting state is deterministic, which we generalize here.

[2] We will often consider a h-step non-stationary policy $(\pi_1, \ldots, \pi_h) \in \Pi^h$ when we only plan to execute this policy for h steps.



(a) (b)

Figure 2: Two example of Block MDPs. **Left:** An example from the Minecraft domain [25]. The agent's observation is given by the history of all observed raw images. The grid on the left shows the latent state space structure. **Right:** The agent starts deterministically in s_1 and can take three different actions $\mathcal{A} = \{a_1, a_2, a_3\}$. Dashed lines denote stochastic transitions while solid lines are deterministic. The numbers on each dashed arrow depict the transition probabilities. We do not show observations for every state for brevity.

Here $\mathbb{P}_\pi[s]$ is the probability of visiting state s along the trajectory taken by π. As in Du et al. [16], our sample complexity guarantees scale polynomially with η_{min}^{-1}, so this quantity should be reasonably large. In contrast with prior work [16, 15], we do not require any further identifiability or determinism assumptions on the environment dynamics.

We call the policies that visit a particular state with maximum probability *homing policies*:

Definition 1 (Homing Policy)**.** *The homing policy for an observation $x \in \mathcal{X}$ is $\pi_x = \arg\max_{\pi \in \Pi_{NS}} \mathbb{P}_\pi[x]$. Similarly, the homing policy for a state $s \in \mathcal{S}$ is $\pi_s = \arg\max_{\pi \in \Pi_{NS}} \mathbb{P}_\pi[s]$.*

In Appendix A, we derive a number of interesting properties for these policies. One key property is their *non compositional* nature. We cannot extend homing policies for states in $\mathcal{S}_{h-1}$ to find homing policies for states in $\mathcal{S}_h$. For example, for the Block MDP in Figure 2b, the homing policy for s_5 takes action a_1 in s_1 but the homing policies for s_2, s_3, and s_4 do not take action a_1 in s_1. Non-compositionality implies that we must take a global policy optimization approach for learning homing policies, which we will do in the sequel.

Reward free learning. In addition to reward-sensitive learning, where the goal is to identify a policy with near-optimal value $V(\pi)$, we also consider a reward-free objective. In this setting, the goal is to find a small set of policies that can be used to visit the entire state space. If we had access to the set of homing policies for every state then this set would suffice. However, in practice we can only hope to learn an approximation. We capture this idea with an *α-policy cover*.

Definition 2 (Policy Cover)**.** *A finite set of non-stationary policies Ψ is called an α-policy cover if for every state $s \in \mathcal{S}$ we have $\sup_{\pi \in \Psi} \mathbb{P}_\pi[s] \geq \alpha\eta(s)$.*

Intuitively, we hope to find a policy cover Ψ of size $O(|\mathcal{S}|)$. By executing each policy in turn, we can collect a dataset of observations and rewards from all states at which point it is relatively straightforward to maximize any reward [28, 38, 6, 39, 4, 18, 12, 2]. Thus, constructing a policy cover can be viewed as an intermediate objective that facilitates reward sensitive learning.

Function classes. In the Block MDP setting we are considering, the agent may never see the same observation twice, so it must use function approximation to generalize across observations. Our algorithm uses two function classes. The first is simply the policy class $\Pi : \mathcal{X} \mapsto \Delta(\mathcal{A})$, which was used above to define the optimal value and the maximum visitation probabilities. For a simpler analysis, we assume Π is finite and measure statistical complexity via $\ln|\Pi|$. However, our results only involve standard uniform convergence arguments, so extensions to infinite classes with other statistical complexity notions is straightforward. Nevertheless, we emphasize that Π is typically not fully expressive.

We also use a family $\mathcal{F}_N$ of regression functions with a specific form. To define $\mathcal{F}_N$, first define $\Phi_N : \mathcal{X} \to [N]$ which maps observations into N discrete abstract states. Second, define $\mathcal{W}_N : [N] \times \mathcal{A} \times [N] \to [0,1]$ as another "tabular" regressor class which consists of *all* functions of the specified type. Then, we set $\mathcal{F}_N := \{(x,a,x') \mapsto w(\phi^{(\mathrm{F})}(x), a, \phi^{(\mathrm{B})}(x')) : w \in \mathcal{W}_N, \phi^{(\mathrm{F})}, \phi^{(\mathrm{B})} \in \Phi_N\}$ and $\mathcal{F} := \cup_{N\in\mathbb{N}}\mathcal{F}_N$. For the analysis, we assume that $|\Phi_N|$ is finite and our bounds scale with $\log|\Phi_N|$, which allows us to search over an exponentially large space of abstraction functions. As $\mathcal{W}_N$ is all functions over a discrete domain, it has pointwise entropy growth rate of $N^2|\mathcal{A}|$ (see Appendix F for a formal definition), and these two considerations determine the complexity of the regression class $\mathcal{F}_N$. As above, we remark that our results use standard uniform convergence arguments, so it is straightforward to extend to other notions.

Computational oracles. As we are working with large function classes, we consider an oracle model of computation where we assume that these classes support natural optimization primitives. This "reductions" approach abstracts away computational issues and addresses the desired situation where these classes are efficiently searchable. Note that the oracle model provides no statistical benefit as the oracles can always be implemented via enumeration; the model simply serves to guide the design of practical algorithms.

Specifically, for the policy class Π, we assume access to an *offline contextual bandit* optimization routine:

$$\texttt{CB}(D,\Pi) := \operatorname*{argmax}_{\pi\in\Pi} \sum_{(x,a,p,r)\in D} \mathbb{E}_{a'\sim\pi(.|x)}\left[\frac{r\mathbf{1}\{a'=a\}}{p}\right].$$

This is a one-step importance weighted reward maximization problem, which takes as input a dataset of (x,a,p,r) quads, where $x \in \mathcal{X}$, $a \in \mathcal{A}$, $p \in [0,1]$ and $r \in \mathbb{R}$ is the reward for the action a, which was chosen with probability p. This optimization arises in contextual bandit settings [46, 8, 49], and is routinely implemented via a further reduction to cost sensitive classification [1].[3]

For the regression class $\mathcal{F}_N$, we assume that we can solve *square loss minimization* problems:

$$\texttt{REG}(D,\mathcal{F}_N) := \operatorname*{argmin}_{f\in\mathcal{F}_N} \sum_{(x,a,x',y)\in D} (f(x,a,x') - y)^2.$$

Here, the dataset consists of (x,a,x',y) quads where $x, x' \in \mathcal{X}$, $a \in \mathcal{A}$ and $y \in \{0,1\}$ is a binary label. Square loss minimization is a standard optimization problem arising in supervised learning, but note that our function class $\mathcal{F}_N$ is somewhat non-standard. In particular, even though square loss regression is computationally tractable for convex classes, our class $\mathcal{F}$ is nonconvex as it involves quantization. This assumption is motivated by the empirical success of latent categorical models in the literature [22, 20].

We emphasize that these oracle assumptions are purely computational and simply guide the algorithm design. In our experiments, we instantiate both Π and $\mathcal{F}_N$ with neural networks, so both oracles solve non-convex problems. This non-convexity does not hinder the empirical effectiveness of the algorithm.

[3] We call this a contextual bandit oracle rather than a cost-sensitive classification oracle because the input dataset is specified in a standard contextual bandit format even though the oracle is formally solving a cost-sensitive classification problem. The advantage is that in practice, we can leverage statistical improvements developed for contextual bandits, such as doubly-robust estimators [17]. These may yield statistical improvements that are not captured by our theory.

For running time calculations, we assume that a single call to CB and REG with n examples can be solved in $\text{Time}_{\text{pol}}(n)$ and $\text{Time}_{\text{reg}}(n)$ time, respectively.

3 Kinematic Inseparability State Abstraction

The foundational concept for our approach is a new form of state abstraction, called *kinematic inseparability*. This abstraction has two key properties. First, it can be learned via a reduction to supervised learning, which we will discuss in detail in Section 4. Second, it enables reward-free exploration of the environment, studied in Section 3.1. In this section, we present the key definitions, some interesting properties, and some intuition.

For exploration, a coarser state abstraction, called *backward kinematic inseparability*, is sufficient.

Definition 3 (Backward Kinematic Inseparability)**.** *Two observations $x'_1, x'_2 \in \mathcal{X}$ are backward kinematically inseparable (KI) if for all distributions $u \in \Delta(\mathcal{X} \times \mathcal{A})$ supported on $\mathcal{X} \times \mathcal{A}$ and for every $x \in \mathcal{X}, a \in \mathcal{A}$ we have*

$$\mathbb{P}_u(x, a \mid x'_1) = \mathbb{P}_u(x, a \mid x'_2), \qquad \textit{where } \mathbb{P}_u(x, a \mid x') = \frac{T(x' \mid x, a)u(x, a)}{\sum_{\tilde{x},\tilde{a}} T(x' \mid \tilde{x}, \tilde{a})u(\tilde{x}, \tilde{a})}.$$

$\mathbb{P}_u(x, a \mid x')$ is the backward dynamics *measuring the probability that the previous observation and action was (x, a) given the current observation is x' and the roll-in distribution over observation and actions is u.*

The significance of backward KI is evident from the following lemma.

Lemma 1. *If x_1, x_2 are backward kinematic inseparable then for all $\pi_1, \pi_2 \in \Pi$ we have $\frac{\mathbb{P}_{\pi_1}(x_1)}{\mathbb{P}_{\pi_2}(x_1)} = \frac{\mathbb{P}_{\pi_1}(x_2)}{\mathbb{P}_{\pi_2}(x_2)}$.*

The proof of this lemma and all mathematical statements in this paper are deferred to the appendices. At a high level, the lemma shows that backward KI observations induce the same ordering over policies with respect to visitation probability. This property is useful for exploration, since a policy that maximizes the probability of visiting an abstract state, also maximizes the probability of visiting each individual observation in that abstract state *simultaneously*. Thus, if we train a policy to visit backward KI abstract state i for each i — which we can do in an inductive manner with synthetic reward functions, as we will see in the next subsection — we guarantee that *all* observations are visited with high probability, so there is no risk of cascading failure. Therefore, this lemma crucially helps establish that a backward KI abstraction enables sample-efficient reward-free exploration.

While backward KI is sufficient for exploration, it ignores the forward dynamics, which are useful for learning a model. This motivates the definition of forward kinematic inseparability:

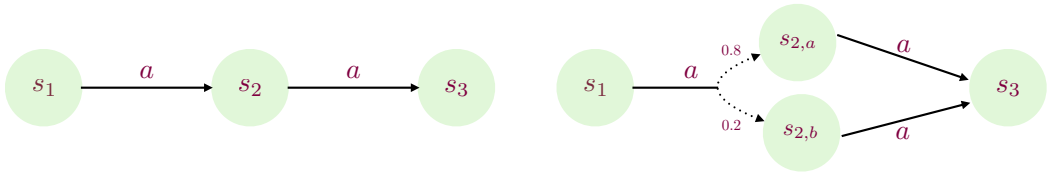
Definition 4 (Forward Kinematic Inseparability)**.** *Two observations $x_1, x_2 \in \mathcal{X}$ are forward kinematically inseparable (KI) if for every $x' \in \mathcal{X}$ and $a \in \mathcal{A}$ we have*

$$T(x' \mid x_1, a) = T(x' \mid x_2, a).$$

Finally, observations are *kinematic inseparable* if they satisfy both of these definitions:

Definition 5 (Kinematic Inseparability)**.** *Two observations x'_1, x'_2 are called kinematically inseparable if for every distribution $u \in \Delta(\mathcal{X} \times \mathcal{A})$ with support over $\mathcal{X} \times \mathcal{A}$ and for every $x \in \mathcal{X}$ and $a \in \mathcal{A}$ we have*

$$\mathbb{P}_u(x, a \mid x'_1) = \mathbb{P}_u(x, a \mid x'_2) \quad \textit{and} \quad T(x \mid x'_1, a) = T(x \mid x'_2, a).$$



(a) (b)

Figure 3: **Left**: A Block MDP with 3 states and 1 action (observations are not depicted). **Right**: We take the MDP on the left and treat s_2 as two states: $s_{2,a}$ and $s_{2,b}$ where $s_{2,a}$ contains observations that are emitted 80% of the time from s_2 and $s_{2,b}$ contains the rest. There is no way to distinguish between these two MDPs, and we call the left MDP the canonical form.

It is straightforward to verify that all three of these notions are equivalence relations, and hence they partition the observation space. The *backward kinematic inseparability dimension*, denoted $\mathrm{N_{BD}}$, is the size of the coarsest partition generated by the backward KI equivalence relation, with $\mathrm{N_{FD}}$ and $\mathrm{N_{KD}}$ defined similarly for the forward KI and KI relations. We also use mappings $\phi_B^\star, \phi_F^\star, \phi^\star : \mathcal{X} \to \mathbb{N}$ to denote these abstractions, for example $\phi_B^\star(x_1) = \phi_B^\star(x_2)$ if and only if x_1 and x_2 are backward KI.

In Appendix B, we collect and prove several useful properties of these state abstractions. Importantly, we show that for Block MDPs, observations emitted from the same state are kinematically inseparable and, hence, $\max\{\mathrm{N_{FD}}, \mathrm{N_{BD}}\} \leq \mathrm{N_{KD}} \leq |\mathcal{S}|$. Ideally, we would like $\mathrm{N_{KD}} = |\mathcal{S}|$ so that the abstract states are in correspondence with the real states of the environment, and we could recover the true model by learning the dynamics between abstract states. In general we may have $\mathrm{N_{KD}} < |\mathcal{S}|$, but only in cases where the true state space is unidentifiable from observations. Figure 3 depicts such an example. From the left panel, if we split s_2 into two states $s_{2,a}, s_{2,b}$ with the same forward dynamics and proportional backward dynamics, we obtain the structure in the right panel. Note that these two Block MDPs are indistinguishable from observations, so we say that the simpler one is the *canonical form*.

Definition 6 (Canonical Form)**.** *A Block MDP is in* canonical form *if there does not exist $x_1, x_2 \in \mathcal{X}$ such that $g^\star(x_1) \neq g^\star(x_2)$ and x_1 and x_2 are kinematically inseparable.*

Note that canonical form is simply a way to identify the notion of state for a Block MDP, it does not restrict this class of environments whatsoever.

3.1 Exploration with an Oracle Kinematic Inseparability Abstraction

We now show that backward KI enables reward-free strategic exploration and reward-sensitive learning, by developing an algorithm, `ExpOracle`, that assumes oracles access to a backward KI abstraction function $\phi^\star : \mathcal{X} \to [N]$. The pseudocode is displayed in Algorithm 1.

`ExpOracle` takes as input a policy class Π, a backward KI abstraction $\phi^\star$, and three hyperparameters $\epsilon, \delta, \eta \in (0, 1)$. The hyperparameter η is an estimate of the reachability parameter (we want $\eta \leq \eta_{min}$), while ϵ, δ are the standard PAC parameters. The algorithm operates in two phases: a reward-free phase in which it learns a policy cover (Algorithm 1, line 2-line 7) and a reward-sensitive phase where it learns a near-optimal policy for the given reward function (Algorithm 1, line 8). In the reward-free phase, the algorithm proceeds inductively, using a policy cover for time steps $1, \ldots, h-1$ to learn a policy cover for time step h. In the h^{th} iteration, we define N *internal reward functions* $\{R_{i,h}\}_{i=1}^N$ corresponding to each output of $\phi^\star$ The reward

Algorithm 1 `ExpOracle`$(\Pi, \phi^\star, \epsilon, \delta, \eta)$. Reinforcement learning in a Block MDP with oracle access to a Backward KI abstraction $\phi^\star : \mathcal{X} \to [N]$

1: Set $n_{\text{psdp}} = \tilde{\mathcal{O}}\left(\frac{N^4 H^2 |\mathcal{A}|}{\eta^2} \ln\left(\frac{|\Pi|}{\delta}\right)\right)$, $n_{\text{eval}} = \tilde{\mathcal{O}}\left(\frac{N^2 H^2 |\mathcal{A}|}{\epsilon^2} \ln\left(\frac{|\Pi|}{\delta}\right)\right)$ and $\Psi_1 = \emptyset$
2: **for** $h = 2, \ldots, H$ **do**
3: $\quad \Psi_h = \emptyset$
4: $\quad$ **for** $i = 1$ to N **do**
5: $\quad\quad$ Define $R_{i,h}(x, a, x') := \mathbf{1}\{\tau(x') = h \wedge \phi^\star(x') = i\}$ // Define internal reward functions
6: $\quad\quad \pi_{i,h} = \texttt{PSDP}(\Psi_{1:h-1}, R_{i,h}, h-1, \Pi, n_{\text{psdp}})$ // Learn an exploration policy using PSDP
7: $\quad\quad \Psi_h \leftarrow \Psi_h \cup \{\pi_{i,h}\}$
8: $\hat{\pi} = \texttt{PSDP}(\Psi_{1:H}, R, H, \Pi, n_{\text{eval}})$ // Reward Sensitive Learning
9: **return** $\hat{\pi}$, $\Psi_{1:H}$

Algorithm 2 `PSDP`$(\Psi_{1:h}, R', h, \Pi, n)$. Learning policy for reward function R' given a policy covers $\Psi_{1:h}$.

1: **for** $t = h, h-1, \cdots, 1$ **do**
2: $\quad D = \emptyset$
3: $\quad$ **for** n times **do**
4: $\quad\quad (x, a, p, r) \sim \texttt{Unf}(\Psi_t) \circ \texttt{Unf}(\mathcal{A}) \circ \hat{\pi}_{t+1} \circ \cdots \circ \hat{\pi}_h$ // PSDP sampling procedure (see text)
5: $\quad\quad D \leftarrow \{(x, a, p, r)\} \cup D$
6: $\quad \hat{\pi}_t = \texttt{CB}(D, \Pi)$ // Solve a contextual bandit problem given dataset D
7: **return** $(\hat{\pi}_1, \hat{\pi}_2, \cdots, \hat{\pi}_h)$

function $R_{i,h}$ gives a reward of 1 if the agent observes x' at time step h satisfying $\phi^\star(x') = i$ and 0 otherwise. The internal reward functions incentivize the agent to reach different backward KI abstract states.

We find a policy that optimizes this internal reward function using the subroutine `PSDP`, displayed in Algorithm 2. This subroutine is based on Policy Search by Dynamic Programming [6], which, using an exploratory data-collection policy, optimizes a reward function by solving a sequence of contextual bandit problems [31] in a dynamic programming fashion. In our case, we use the policy covers for time steps $1, \ldots, h-1$ to construct the exploratory policy (Algorithm 2, line 4).

Formally, at time step t of `PSDP`, we solve the following optimization problem:

$$\max_{\pi \in \Pi} \mathbb{E}_{x_t \sim D_t, a_t \sim \pi, a_{t+1} \sim \hat{\pi}_{t+1}, \cdots, a_h \sim \hat{\pi}_h} \left[\sum_{h'=t}^{h} R'(x_{h'}, a_{h'}, x_{h'+1}) \right],$$

using the previously computed solutions $(\hat{\pi}_{t+1}, \cdots, \hat{\pi}_h)$ for future time steps. The context distribution D_t is obtained by uniformly sampling a policy in Ψ_t and rolling-in with it until time step t. To solve this problem, we first collect a dataset D of tuples (x, a, p, r) of size n by (1) sampling x by rolling-in with a uniformly selected policy in Ψ_t until time step t, (2) taking action a uniformly at random, (3) setting $p = 1/|\mathcal{A}|$, and (4) executing $\hat{\pi}_{t+1:h}$, and (5) setting $r = \sum_{h'=t}^{h} r_{t'}$. Then we invoke the contextual bandit oracle `CB` with dataset D to obtain $\hat{\pi}_t$. Repeating this process we obtain the non-stationary policy $\hat{\pi}_{1:h}$ which is returned by `PSDP`.

Algorithm 3 HOMER$(\Pi, \mathcal{F}, N, \eta, \epsilon, \delta)$. Perform reinforcement learning on a given Block MDP by learning to simultaneously do strategic exploration and learn kinematic inseparability state abstraction.

1: Set $\Psi_1 = \emptyset$, $n_{\text{reg}} = \widetilde{\mathcal{O}}\left(\frac{N^6|\mathcal{A}|^3}{\eta^3}\left(N^2|\mathcal{A}| + \ln\left(\frac{|\Phi_N|H}{\delta}\right)\right)\right)$ and $n_{\text{psdp}} = \widetilde{\mathcal{O}}\left(\frac{N^4H^2|\mathcal{A}|}{\eta^2}\ln\left(\frac{|\Pi|}{\delta}\right)\right)$.
2: **for** $h = 2, \ldots, H$ **do**
3: $\quad D = \emptyset$, $\Psi_h = \emptyset$.
4: $\quad$ **for** n_{reg} times **do**
5: $\quad\quad (x_1, a_1, x_1'), (x_2, a_2, x_2') \sim \texttt{Unf}(\Psi_{h-1}) \circ \texttt{Unf}(\mathcal{A})$
6: $\quad\quad y \sim \text{Ber}(1/2)$
7: $\quad\quad$ **if** $y = 1$ **then**
8: $\quad\quad\quad D \leftarrow D \cup \{([x_1, a_1, x_1'], 1)\}$, // Add a real transition
9: $\quad\quad$ **else**
10: $\quad\quad\quad D \leftarrow D \cup \{([x_1, a_1, x_2'], 0)\}$. // Add an imposter transition
11: $\quad (w_h, \hat{\phi}_{h-1}^{(\text{F})}, \hat{\phi}_h^{(\text{B})}) = \texttt{REG}(\mathcal{F}_N, D)$ // Supervised learning on D
12: $\quad$ **for** $i = 1$ to N **do**
13: $\quad\quad$ Define $R_{i,h}(x, a, x') := \mathbf{1}\{\tau(x') = h \wedge \hat{\phi}_h^{(\text{B})}(x') = i\}$
14: $\quad\quad \pi_{i,h} = \texttt{PSDP}(\Psi_{1:h-1}, R_{i,h}, h-1, \Pi, n_{\text{psdp}})$
15: $\quad\quad \Psi_h \leftarrow \Psi_h \cup \{\pi_{i,h}\}$
16: Set $n_{\text{eval}} = \widetilde{\mathcal{O}}\left(\frac{N^2H^2|\mathcal{A}|}{\epsilon^2}\ln\left(\frac{|\Pi|}{\delta}\right)\right)$
17: $\hat{\pi} = \texttt{PSDP}(\Psi_{1:H}, R, H, \Pi, n_{\text{eval}})$ // Reward Sensitive Learning
18: **return** $\hat{\pi}$, $\Psi_{1:H}$, $\hat{\phi}_{1:H-1}^{(\text{F})}$, $\hat{\phi}_{2:H}^{(\text{B})}$

Learn State Abstraction

Learn Policy Cover

The learned policy cover Ψ_h for time step h is simply the policies identified by optimizing each of the N internal reward functions $\{R_{i,h}\}_{i=1}^N$. Once we find the policy covers $\Psi_1, \ldots, \Psi_H$, we perform reward-sensitive learning via a single invocation of PSDP using the *external* reward function R (Algorithm 1, line 8). Of course, in a purely reward free setting, we can omit this last step and simply return the policy covers $\Psi_{1:H}$.

4 Learning Kinematic Inseparability for Strategic Exploration

We now present our main algorithm HOMER, displayed in Algorithm 3, which learns a kinematic inseparability abstraction while performing reward-free strategic exploration. Given a Block MDP $\mathcal{M}$, hypothesis classes Π and $\mathcal{F}$, a positive integer N, and three hyperparameters $\eta, \epsilon, \delta \in (0, 1)$, HOMER learns a policy cover of size N and a state abstraction function for each time step. We assume that $N \geq \text{N}_{\text{KD}}$ and $\eta \leq \eta_{min}$. In Section 5 we use these conditions for our theoretical analysis and also discuss how to relax them via an adaptive procedure.

The overall structure of HOMER is similar to ExpOracle with a reward-free phase preceding the reward-sensitive one. As with ExpOracle, HOMER proceeds inductively, learning a policy cover for observations reachable at time step h given the learned policy covers $\Psi_{1:h-1}$ for previous steps (Algorithm 3, line 2-line 15).

The key difference with ExpOracle is that, for each iteration h, we first learn an abstraction function $\hat{\phi}_h^{(\text{B})}$ over $\mathcal{X}_h$. This is done using a form of contrastive estimation and our function class $\mathcal{F}_N$. Specifically

in the h^{th} iteration, HOMER collects a dataset D of size n_{reg} containing real and imposter transitions. We define a sampling procedure: $(x, a, x') \sim \texttt{Unf}(\Psi_{h-1}) \circ \texttt{Unf}(\mathcal{A})$ where x is observed after rolling-in with a uniformly sampled policy in Ψ_{h-1} until time step $h-1$, action a is taken uniform randomly, and x' is observed sampled from the transition $T(\cdot \mid x, a)$ (Algorithm 3, line 5). We sample two independent transitions $(x_1, a_1, x_1'), (x_2, a_2, x_2')$ using this procedure, and we also sample a Bernoulli random variable $y \sim \text{Ber}(1/2)$. If $y = 1$ then we add the observed transition $([x_1, a_1, x_1'], y)$ to D and otherwise we add the *imposter* transition $([x_1, a_1, x_2'], y)$ (Algorithm 3, line 6-line 10). Note that the imposter transition may or may not correspond feasible environment outcomes.

We call the subroutine REG to solve the supervised learning problem induced by D with model family $\mathcal{F}_N$ (Algorithm 3, line 11), and we obtain a predictor $\hat{f}_h = (w_h, \hat{\phi}_{h-1}^{(\text{F})}, \hat{\phi}_h^{(\text{B})})$. As we will show, $\hat{\phi}_h^{(\text{B})}$ is closely related to a backward KI abstraction, so we can use it to drive exploration at time step h, just as in ExpOracle. We will also see that $\hat{\phi}_{h-1}^{(\text{F})}$ is closely related to a forward KI abstraction, which is useful for auxiliary tasks such as learning the transition dynamics or visualization.

With $\hat{\phi}_h^{(\text{B})}$ the rest of the algorithm proceeds similarly to ExpOracle. We invoke PSDP with the N internal reward functions induced by $\hat{\phi}_h^{(\text{B})}$ to find the policy cover (Algorithm 3, line 14). Then once we have found policy covers for all time steps, we perform reward-sensitive optimization just as before (Algorithm 3, line 17). As with ExpOracle we can ignore the reward-sensitive phase and operate in a purely reward free setting by simply returning the policy covers $\Psi_{1:H}$.

We can view the combination of the two learned state abstractions $\overline{\phi}_h = (\hat{\phi}_h^{(\text{F})}, \hat{\phi}_h^{(\text{B})})$ as the learned KI abstraction where for any $x \in \mathcal{X}$, we have $\overline{\phi}_h(x) = (\hat{\phi}_h^{(\text{F})}(x), \hat{\phi}_h^{(\text{B})}(x))$ and for any $x_1, x_2 \in \mathcal{X}$, $\overline{\phi}_h(x_1) = \overline{\phi}_h(x_2)$ if and only if $\hat{\phi}_h^{(\text{F})}(x_1) = \hat{\phi}_h^{(\text{F})}(x_2)$ and $\hat{\phi}_h^{(\text{B})}(x_1) = \hat{\phi}_h^{(\text{B})}(x_2)$. We define $\hat{\phi}_1^{(\text{B})}(x) := 1$ and $\hat{\phi}_H^{(\text{F})} := 1$ to be identity functions as there is no backward and forward dynamics information available at these steps, respectively. We use $\overline{\phi}$ for learning the transition dynamics and visualization (see Section 8).

5 Theoretical Analysis

The main theoretical contribution of this paper is to show that our algorithm computes a near-optimal policy with high probability in a sample-efficient and computationally-tractable manner. The result requires an additional expressivity assumption on our model classes Π and $\mathcal{F}$, which we now state.

Assumption 2 (Realizability). *Let* $\mathcal{R} := \{R\} \cup \{(x, a, x') \mapsto \mathbf{1}\{\phi(x') = i \wedge \tau(x') = h\} \mid \phi \in \Phi_N, i \in [N], h \in [H]\}$ *be the set of external and internal reward functions. We assume that* Π *satisfies* policy completeness *for every* $R' \in \mathcal{R}$*: for every* $h \in [H]$ *and* $\pi' \in \Pi_{NS}$*, there exists* $\pi \in \Pi$ *such that*

$$\forall x \in \mathcal{X}_h : \quad \pi(x) = \operatorname*{argmax}_{a \in \mathcal{A}} \mathbb{E}\left[\sum_{h'=h}^{H} r_{h'} \mid x_h = x, a_h = a, a_{h'} \sim \pi'\right].$$

We also assume that $\mathcal{F}$ *is* realizable*: for any* $h \in [H]$*,* $N \geq N_{KD}$*, and any prior distribution* $\rho \in \Delta(\mathcal{S}_h)$ *with* $\mathit{supp}(\rho) = \mathcal{S}_h$*, there exists* $f_\rho \in \mathcal{F}_N$ *such that*

$$\forall x_{h-1}, a_{h-1}, x_h : \qquad f_\rho(x_{h-1}, a_{h-1}, x_h) = \frac{T(g^\star(x_h) | g^\star(x_{h-1}), a_{h-1})}{T(g^\star(x_h) | g^\star(x_{h-1}), a_{h-1}) + \rho(g^\star(x_h))}.$$

Completeness assumptions are common in the analysis of dynamic programming style algorithms for the function approximation setting [4] (see Chen and Jiang [12] for a detailed discussion). Our exact completeness

assumption appears in the work of Dann et al. [15], who use it to derive an efficient algorithm for a restricted version of our setting with deterministic latent state transitions.

The realizability assumption on $\mathcal{F}$ is adapted to our learning approach: as we use $\mathcal{F}$ to distinguish between real and imposter transitions, $\mathcal{F}$ should contain the Bayes optimal classifier for these problems. In HOMER, the sampling procedure $\texttt{Unf}(\Psi_{h-1}) \circ \texttt{Unf}(\mathcal{A})$ that is used to collect data for the learning problem in the h^{th} iteration induces a marginal distribution $\rho \in \Delta(\mathcal{S}_h)$ and the Bayes optimal predictor for this problem is precisely f_ρ (See Lemma 9 in Appendix E). It is not hard to see that if x_1, x_2 are kinematically inseparable then $f_\rho(x_1, a, x') = f_\rho(x_2, a, x')$ and the same claim holds for the third argument of f_ρ. Therefore, by the structure of $\mathcal{F}_N$, a sufficient condition to ensure realizability is that Φ_N contains a kinematic inseparability abstraction, which is reasonable as this is precisely what we are trying to learn.

Theoretical Guarantees. We first present the guarantee for ExpOracle.

Theorem 1 (ExpOracle Result)**.** *For any Block MDP, given a backward kinematic inseparability abstraction $\phi^\star_{\text{back}} : \mathcal{X} \to [N]$ such that $\phi^\star_{\text{back}} \in \Phi_N$, and parameters $(\epsilon, \eta, \delta) \in (0,1)^3$ with $\eta \leq \eta_{min}$, ExpOracle outputs policy covers $\{\Psi_h\}_{h=1}^H$ and a reward sensitive policy $\hat{\pi}$ such that the following guarantees hold, with probability at least $1-\delta$:*

1. *For each $h \in [H]$, Ψ_h is a $1/2$-policy cover for $\mathcal{S}_h$;*
2. *$V(\hat{\pi}) \geq \max_{\pi \in \Pi_{NS}} V(\pi) - \epsilon$.*

The sample complexity is $\widetilde{\mathcal{O}}\left(NH^2 n_{\text{psdp}} + H n_{\text{eval}}\right) = \widetilde{\mathcal{O}}\left(N^2H^3|\mathcal{A}|\log(|\Pi|/\delta)\left(NH/\eta^2_{min} + 1/\epsilon^2\right)\right)$, and the running time is $\widetilde{\mathcal{O}}\left(NH^3 n_{\text{psdp}} + H^2 n_{\text{eval}} + NH^2 \cdot \text{Time}_{\text{pol}}(n_{\text{psdp}}) + H \cdot \text{Time}_{\text{pol}}(n_{\text{eval}})\right)$, where n_{psdp} and n_{eval} are defined in Algorithm 1.

Theorem 1 shows that we can learn a policy cover and use it to learn a near-optimal policy, assuming access to a backward KI abstraction. The sample complexity is $\text{poly}(N, H, |\mathcal{A}|, \eta_{min}^{-1}, \epsilon^{-1}, \log|\Pi|/\delta)$, which at a coarse level is our desired scaling. We have not attempted to optimize the exponents in the sample complexity or running time.

We next present the result for HOMER. Here we show that HOMER achieves a similar guarantee to ExpOracle, *without* prior access to a backward KI abstraction. In other words, HOMER provably learns a backward KI abstraction and uses it for exploration and policy optimization.

Theorem 2 (Main Result)**.** *For any Block MDP and hyperparameters $\epsilon, \delta, \eta \in (0,1), N \in \mathbb{N}$, satisfying $\eta \leq \eta_{min}$ and $N \geq N_{KD}$, HOMER outputs a set of exploration policies $\{\Psi_h\}_{h=1}^H$ and a reward sensitive policy $\hat{\pi}$ satisfying:*

1. *Ψ_h is an $1/2$-policy cover of $\mathcal{S}_h$ for every $h \in [H]$*
2. *$V(\hat{\pi}) \geq \max_{\pi \in \Pi_{NS}} V(\pi) - \epsilon$*

with probability least $1-\delta$. The sample complexity of HOMER is $\mathcal{O}\left(n_{\text{psdp}} NH^3 + n_{\text{reg}} H + n_{\text{eval}} H\right)$ where $n_{\text{psdp}}, n_{\text{reg}}, n_{\text{eval}}$ are as specified in Algorithm 3. This evaluates to

$$\widetilde{\mathcal{O}}\left(\frac{N^8|\mathcal{A}|^4 H}{\eta^3_{min}} + \frac{N^6|\mathcal{A}|H}{\eta^3_{min}}\ln(|\Phi_N|/\delta) + \left(\frac{N^5H^4|\mathcal{A}|}{\eta^2_{min}} + \frac{N^2H^3|\mathcal{A}|}{\epsilon^2}\right)\ln(|\Pi|/\delta)\right).$$

The running time is

$$\mathcal{O}\left(n_{\text{psdp}} NH^3 + n_{\text{reg}} H^2 + n_{\text{eval}} H^2 + \text{Time}_{\text{pol}}(n_{\text{psdp}}) NH^2 + \text{Time}_{\text{reg}}(n_{\text{reg}}) H + \text{Time}_{\text{pol}}(n_{\text{eval}}) H\right).$$

Since $n_{\text{reg}}, n_{\text{psdp}}$, and n_{eval} are polynomial in the relevant parameters, HOMER also has polynomial sample complexity. In comparison with the guarantee for ExpOracle, the main qualitative difference is that the guarantee for HOMER also has a dependence on $\log|\Phi_N|$, which is natural as HOMER attempts to learn the backward KI function. Crucially, the logarithmic dependence on $|\Phi_N|$ implies that we can take Φ_N to be exponentially large and achieve a guarantee that is qualitatively comparable to that of ExpOracle. This demonstrates that we can learn a backward KI abstraction function from an exponentially large class and then use it for exploration and policy optimization.

In terms of computation, the running time is polynomial in our oracle model, where we assume we can solve contextual bandit problems over Π and regression problems over $\mathcal{F}_N$. In Section 8 we will see that these problems can be solved effectively in practice.

Why does HOMER learn kinematic inseparability? A detailed proof of both theorems is deferred to Appendix C-Appendix E, but for intuition, we provide a sketch of how HOMER learns a kinematic inseparability abstraction. For this discussion only, we focus on asymptotic behavior and ignore sampling issues.

Inductively, assume that Ψ_{h-1} is a policy cover of $\mathcal{S}_{h-1}$, let $D(x,a,x')$ be the marginal distribution over real and imposter transitions sampled by HOMER in the h^{th} iteration (line 4–line 10), and let ρ be the marginal distribution over $\mathcal{X}_h$. First observe that as Ψ_{h-1} is a policy cover, we have $\text{supp}(D) = \mathcal{X}_{h-1}\times\mathcal{A}\times\mathcal{X}_h$, which is crucial for our analysis. Let $\hat{f} = (\hat{w}, \hat{\phi}^{(\text{F})}_{h-1}, \hat{\phi}^{(\text{B})}_h)$ be the output of the regression oracle REG in this iteration. The first observation is that the Bayes optimal regressor for this problem is f_ρ defined in Assumption 2, and, with realizability, in this asymptotic discussion we have $\hat{f} \equiv f_\rho$.

Next, we show that for any two observations $x'_1, x'_2 \in \mathcal{X}_h$, if $\hat{\phi}^{(\text{B})}_h(x'_1) = \hat{\phi}^{(\text{B})}_h(x'_2)$ then x'_1 and x'_2 are backward kinematically inseparable. If this precondition holds, then

$$\forall x \in \mathcal{X}_{h-1}, a \in \mathcal{A}: \quad f_\rho(x,a,x'_1) = \hat{p}(\hat{\phi}^{(\text{F})}_{h-1}(x), a, \hat{\phi}^{(\text{B})}_h(x'_1)) = \hat{p}(\hat{\phi}^{(\text{F})}_{h-1}(x), a, \hat{\phi}^{(\text{B})}_h(x'_2)) = f_\rho(x,a,x'_2).$$

Then, by inspection of the form of f_ρ and elementary algebra, we have

$$f_\rho(x,a,x'_1) = f_\rho(x,a,x'_2) \Leftrightarrow \frac{T(x'_1 \mid x,a)}{\rho(x'_1)} = \frac{T(x'_2 \mid x,a)}{\rho(x'_2)}.$$

As this identity holds for all $x \in \mathcal{X}_{h-1}, a \in \mathcal{A}$ and trivially when $x \notin \mathcal{X}_{h-1}$, it is easy to see that x'_1, x'_2 satisfy Definition 3, so they are backward kinematically inseparable. In particular, taking expectation against any prior $u \in \Delta(\mathcal{X},\mathcal{A})$, we have

$$\begin{aligned}\mathbb{P}_u(x,a \mid x'_1) = \frac{T(x'_1 \mid x,a)u(x,a)}{\sum_{\tilde{x},\tilde{a}} T(x'_1 \mid \tilde{x},\tilde{a})u(\tilde{x},\tilde{a})} &= \frac{\frac{\rho(x'_1)}{\rho(x'_2)}T(x'_2 \mid x,a)u(x,a)}{\sum_{\tilde{x},\tilde{a}} \frac{\rho(x'_1)}{\rho(x'_2)}T(x'_2 \mid \tilde{x},\tilde{a})u(\tilde{x},\tilde{a})} \\ &= \frac{T(x'_2 \mid x,a)u(x,a)}{\sum_{\tilde{x},\tilde{a}} T(x'_2 \mid \tilde{x},\tilde{a})u(\tilde{x},\tilde{a})} = \mathbb{P}_u(x,a \mid x'_2).\end{aligned}$$

This implies that $\hat{\phi}^{(\text{B})}_h$ is a backward KI abstraction over $\mathcal{X}_h$.

While this suffices for the purpose of exploration, we also show that $\hat{\phi}^{(\text{F})}_{h-1}$ is a forward KI abstraction. As before, if $\hat{\phi}^{(\text{F})}_{h-1}(x_1) = \hat{\phi}^{(\text{F})}_{h-1}(x_2)$ then we must have $f_\rho(x_1,a,x') = f_\rho(x_2,a,x')$ for all $a \in \mathcal{A}, x' \in \mathcal{X}_h$. Using the structure of f_ρ and the fact that $\text{supp}(\rho) = \mathcal{X}_h$, this implies that $T(x' \mid x_1,a) = T(x' \mid x_2,a)$ for all x', a, so x_1, x_2 are forward kinematically inseparable. The KI abstraction is formed by combining the backward abstraction learned over $\mathcal{X}_{h-1}$ in the previous iteration of HOMER with this forward abstraction.

Setting hyperparameters η and N. In practice, we set η and N using a restarting approach and a doubling scheduling. Specifically, we run HOMER inside an outer loop where in the t^{th} iteration of this outerloop we execute HOMER with $\eta = 2^{-t}$ and $N = 2^t$. During the execution, we track the validation performance of all contextual bandit and regression problems and we terminate the outer loop when validation performance plateaus. After $t_0 := \max\{\lceil \log_2(\frac{1}{\eta_{min}}) \rceil, \lceil \log_2(\text{N}_{\text{KD}}) \rceil\}$ iterations of the outer loop, the requirements of Theorem 2 are satisfied. Moreover, as the Bayes optimal predictors are in our classes at round t_0 (using Assumption 2), the validation performance cannot decrease further. Therefore, we can hope to terminate the outer loop after $\mathcal{O}(t_0)$ rounds. This doubling schedule only affects our bounds in constant and logarithmic factors.

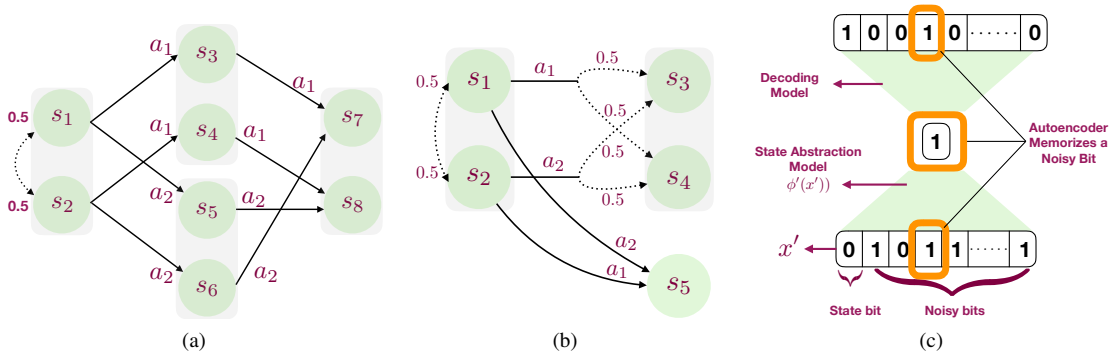
Efficient Implementation of HOMER. As stated, the most computationally expensive component of HOMER is the $\mathcal{O}(NH)$ calls to PSDP for learning the policy covers. This has a total computational cost of $\mathcal{O}(NH^3 n_{\text{psdp}} + \text{Time}_{\text{pol}}(n_{\text{psdp}})NH^2)$, but in practice, it can be significantly reduced. Empirically, we use two important optimizations: First, we parallelize the N calls to PSDP for optimizing the internal reward functions in each iteration of the algorithm (line 12–line 15). Second and perhaps more significantly, we can attempt to find compositional policies using a greedy search procedure (GPS). Here, rather than perform full dynamic programming to optimize reward $R_{i,h}$, we find the policy $\hat{\pi}_h$ for the last time step, and then we append this policy to the best one from our cover Ψ_{h-1}. Formally, we compute $\hat{\pi}_{1:h-1} = \text{argmax}_{\pi \in \Psi_{h-1}} V(\pi \circ \hat{\pi}_h; R_{i,h})$, where $V(\cdot; R)$ is the value function with respect to reward function R and $\circ$ denotes policy composition. Then, since the optimal value with respect to $R_{i,h}$ is at most 1, we check if $V(\hat{\pi}_{1:h-1} \circ \hat{\pi}_h; R_{i,h}) \geq 1 - \epsilon$. If it is we need not perform a full dynamic programming backup, otherwise we revert to PSDP. GPS may succeed even though the policies we are trying to find are non-compositional in general. In the favorable situation where GPS succeeds, actually no further samples are required, since we can re-use the real transitions from the regression step, and we need only solve one contextual bandit problem, instead of H. Empirically, both of these optimizations may yield significant statistical and computational savings.

6 Can We Use Existing State Abstraction Oracles?

Our analysis so far verifies the utility of the backward KI state abstraction: it enables efficient exploration via ExpOracle and it can be learned using contrastive estimation procedure with a bottleneck classifier as in HOMER. Do other, previously studied, state abstractions admit similar properties?

In this section, we discuss three prior approaches for learning state abstractions. In Block-MDPs, we show that these approaches fail to find a policy cover when interleaved with a PSDP-style routine used to find policies that visit the abstract states, following the structure of HOMER. Note that it may be possible to embed these approaches in other algorithmic frameworks and successfully explore.

Predicting Previous Action from Observations. Curiosity based approaches learn a representation by predicting the previous action from the previous and current observation [41]. When embedded in a PSDP-style routine, this approach fails to guarantee coverage of the latent state space, as can be seen in Figure 4a. A Bayes optimal predictor of previous action a given previous and current observations x, x' collapses the observations generated from $\{s_3, s_4\}$, $\{s_5, s_6\}$, and $\{s_7, s_8\}$ together. To see why, the agent can only transition to $\{s_3, s_4\}$ by taking action a_1, so we can perfectly predict the previous action a_1 even if all of the observations from these states have the same representation. This also happens with $\{s_5, s_6\}$ and $\{s_7, s_8\}$. Unfortunately, if we collapse observations from $\{s_7, s_8\}$ together, we create an unfavorable tie-breaking scenario in the policy optimization step when trying to find a policy to visit this representation. For example,



(a) (b) (c)

Figure 4: Counterexamples for prior work on abstraction/representation learning. We do not show observations for brevity. **Left:** A Block MDP where predicting the previous action from observations [41] or predicting the previous abstract state and action fails [16]. **Middle:** A Block-MDP where the model-based algorithm of Du et al. [16] fails. **Right:** Illustration of a failure mode for the autoencoding approach of Tang et al. [50], where optimal reconstruction loss is attained by memorizing noise. See text for more details.

the policy that takes action a_1 in s_1 and s_3 and a_2 in s_2 and s_6 deterministically reaches s_7, so it visits this representation maximally, but it never visits s_8. So this approach for representation learning, interleaved with policy optimization, does not yield a policy cover. HOMER in contrast will map observations from $\{s_7, s_8\}$ to different abstract states and therefore, will not fail.

Predicting Previous Action and Abstract State. Instead of predicting the previous action, Du et al. [16] learn a representation by predicting the previous action *and* previous abstract state. As they show, this approach can be used to provably learn a restricted class of Block-MDPs, but unfortunately this approach fails in the general case. For example in Figure 4a, a Bayes optimal predictor for the prediction problem collapses observations from $\{s_1, s_2\}$, $\{s_3, s_4\}$, $\{s_5, s_6\}$, and $\{s_7, s_8\}$, leading to the same failure for policy optimization as the curiosity-based approaches. This state collapses is caused by a stochastic start state; $\{s_1, s_2\}$ cannot be separated by this approach, and using the joint representation for $\{s_1, s_2\}$ as a prediction target causes a cascading failure. Note that Du et al. [16] assume a deterministic start state in their analysis.

Instead of a PSDP-style routine, Du et al. [16] use a model-based approach for building a policy cover, where the learned policies operate directly on the abstract states. Actually this approach avoids the tie-breaking issue in policy optimization and does succeed for the example in Figure 4a, but it fails in Figure 4b. If policies are defined over abstract states, we must take the same action in s_1 and s_2 (as this approach can never separate a stochastic start state), so we can reach $\{s_3, s_4\}$ with probability at most $1/2$, while a policy operating directly on observations could reach these states with probability 1. Chaining this construction together shows that this approach can at best find an α-policy cover where α is exponentially small in the horizon.

Training Autoencoders. The final approach we discuss uses an autoencoder to learn a representation, similar to the approach of Tang et al. [50]. Here we learn a representation ϕ and a decoder U by minimizing the reconstruction loss $\texttt{dist}(x, U(\phi(x)))$ over a training set of raw observations, where dist is some domain-specific distance function. Figure 4c shows that this approach may fail to learn a meaningful representation altogether. The RL problem contains just two states s_0 and s_1 and the observations for both

states are d-dimensional binary vectors, where the first bit encodes the state and the remaining bits are sampled from Ber($1/2$) (it is easy to see that this is a Block-MDP). For this problem, optimal reconstruction loss is achieved by a representation that ignores that state bit and memorizes (a subset of) the noise bits. For example, if the representation ϕ has a single output bit (which is sufficient as there are only two states), it is never worse to preserve the first noise bit of the input than to preserve the state bit (there is a tie if s_0 and s_1 are equally probable). In fact, if one state is more probable than other then preserving a noise bit and reconstructing it along with outputting the most likely state, performs better than preserving the state bit. So an optimal representation using this autoencoding approach will ignore all state information and is not useful for exploration.

Summary. The arguments and counter-examples in this section show that previously studied state-abstraction or representation learning approaches cannot be used for provably efficient exploration in general Block-MDPs, at least when used in a similar algorithmic framework to `HOMER`. We emphasize that our analysis does not preclude the value of these approaches in other RL settings (e.g., outside of Block-MDPs) or when used in other algorithms. Moreover, the remarks here are of a worst-case nature and do not necessarily imply that the approaches are empirically ineffective.

7 Related Work

Sample efficient exploration of Markov Decision Processes with a small number of observed states has been studied in a number of papers [10, 48, 21], initiated by the breakthrough result of Kearns and Singh [29]. While state-of-the-art theoretical results provide near-optimal sample complexity bounds for these small-state MDPs, the algorithms do not exploit latent structure in the environment, and therefore, cannot scale to the rich-observation environments that are popular in modern empirical RL.

A recent line of theoretical work [30, 23] focusing on rich observation reinforcement learning has shown that it is information-theoretically possible to explore these environments and has provided computationally efficient algorithms for some special settings. In particular, Dann et al. [15] considers deterministic latent-state dynamics while Du et al. [16] allows for limited stochasticity. The present work continues in this line, further expanding the scope for tractable rich observation reinforcement learning. In particular, compared to Du et al. [16], `HOMER` can accommodate a stochastic starting state and does not require margin conditions on the dynamics. Our algorithm also does not rely on abstract states for defining policies or future prediction problems, therefore, avoiding cascading errors due to inaccurate predictions (See discussion in Section 6). On the other hand, we rely on different computational assumptions, which we show to be empirically reasonable.

On the empirical side, a number of approaches have been proposed for exploration with large observation spaces using pseudo-counts [50], optimism-driven exploration [13], intrinsic motivation [7] and prediction errors [41]. While these algorithms can perform well on certain RL benchmarks, we lack a deep understanding of their behavior and failure modes. As the examples in Section 6 show, using the representations learned by these methods for provably efficient exploration is challenging, and may not be possible in some cases.

Most closely related to our work, Nachum et al. [40] use a supervised learning objective similar to ours for learning state abstractions. However, they do not address the problem of exploration and do not provide any sample complexity guarantees. Importantly, we arrive at our supervised learning objective from the need to learn kinematic inseparability which itself is motivated by the `ExpOracle` algorithm.

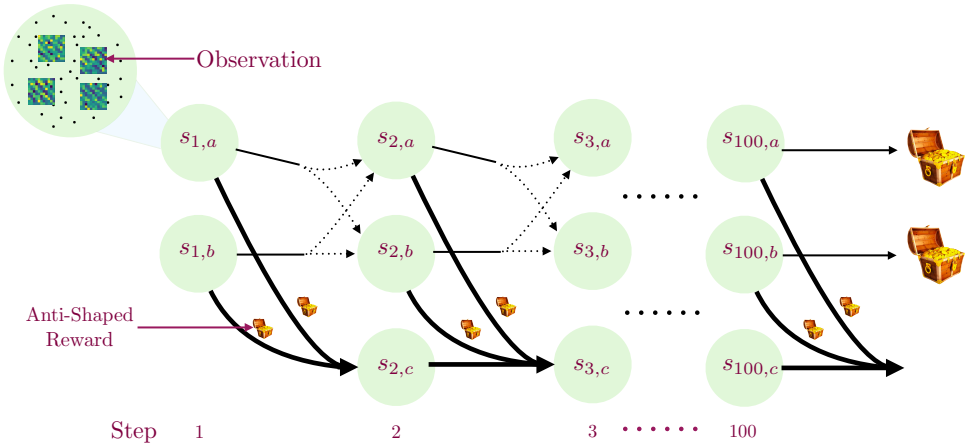


Figure 5: Illustrates the diabolical combination lock problem which contains multiple challenges including sparse anti-shaped rewards, rich-observations, long horizons and extremely sparse good rewards.

8 Proof of Concept Experiments

We evaluate HOMER on a challenging problem called a *diabolical combination lock* that contains high-dimensional observations, precarious dynamics, anti-shaped rewards where greedy optimization leads to sub-optimal policies, and googal-sparse good rewards.

Problem Setup. The diabolical combination lock is a class of rich observation MDPs, with an example shown in Figure 5. For a fixed horizon H and action space size K, the state space is given by $\mathcal{S} = \{s_{1,a}, s_{1,b}\} \cup \{s_{h,a}, s_{h,b}, s_{h,c}\}_{h=2}^{H}$ and the action space by $\mathcal{A} = \{a_1, ..., a_K\}$. The agent starts in either $s_{1,a}$ or $s_{1,b}$ with equal probability. After taking h actions the agent is in $s_{h+1,a}, s_{h+1,b}$ or $s_{h+1,c}$.

Informally, the states $\{s_{h,a}\}_{h=2}^{H}$ and $\{s_{h,b}\}_{h=2}^{H}$ are "good" states which may lead to the optimal return, and the states $\{s_{h,c}\}_{h=2}^{H}$ are "bad" states from which it it impossible to achieve the optimal return. Each good state $s_{h,a}$ and is associated to a good action u_h which will lead the agent to one of the two good states at step $h+1$, and similarly each good state $s_{h,b}$ is associated with a good action v_h. All other actions lead the agent to the bad state $s_{h+1,c}$ from which it is impossible to recover. Formally, the transition dynamics are given by:

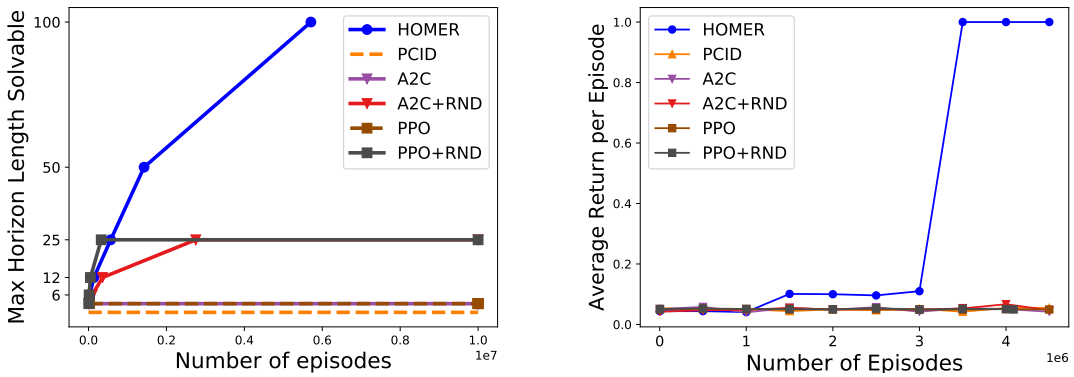
$$
\begin{aligned}
&T(s_{h+1,a} \mid s_{h,a}, u_h) = T(s_{h+1,b} \mid s_{h,a}, u_h) = \frac{1}{2}, \\
&T(s_{h+1,a} \mid s_{h,b}, v_h) = T(s_{h+1,b} \mid s_{t,b}, v_h) = \frac{1}{2}, \\
&T(s_{h+1,c} \mid s_{h,a}, a') = 1, \qquad \forall a' \neq u_h, \\
&T(s_{h+1,c} \mid s_{h,b}, a') = 1, \qquad \forall a' \neq v_h, \\
&T(s_{h+1,c} \mid s_{h,c}, a') = 1, \qquad \forall a'
\end{aligned}
$$

The agent receives a reward of 1 on taking action u_H in $s_{H,a}$ or action v_H in $s_{H,b}$. Upon transitioning from $s_{h,a}, s_{h,b}$ to $s_{h+1,c}$ the agent receives an anti-shaped reward of 0.1 with probability 0.5. This anti-shaped reward is easy for the agent to achieve but is negatively correlated with long term performance. The agent

receives a reward of 0 for all other transitions. The reachability parameter for this problem is $\frac{1}{2}$ and $V(\pi^\star)$ is given by 1 where for all $h \in [H]$, $\pi^\star$ takes action u_h in $s_{h,a}$ and action v_h in $s_{h,b}$.

The agent never directly observes the true state it is in, and instead receives an observation $x \in \mathbb{R}^d$ which is generated stochastically as follows. First, the current state information and time step are encoded into one-hot vectors which are concatenated, and isotropic Gaussian noise with mean 0 and variance 0.1 is added. This vector is then zero-padded and multiplied by a fixed Hadamard matrix. [4]

Our main experiments are on the setting with horizon (H) of 100 and an action space of size (K) 10. In this case, the probability of a uniform random walk getting a reward of 1 is $\frac{1}{10^{100}}$.[5] In fact for any fixed sequence of actions the probability of getting a reward of 1 is at most $\frac{1}{2^\tau}$ where $\tau = \sum_{h=1}^{100} 1\{u_h \neq v_h\}$. When creating an instance of the problem for a fixed horizon and action space, we choose u and v randomly. This gives us an expected value of τ as 90 for the above setting.



(a) (b)

Figure 6: Results on the diabolical combination lock problem with action space (K) of size 10. **Left:** Horizon of the problem against episodes needed to learn a policy with value at least half of optimal policy. **Right:** Empirical return of policy against number of episodes for horizon of 100. The value of the optimal policy is 1.

HOMER Implementation. We use non-stationary deterministic policies, where each policy is represented as a tuple of H linear models $\pi = (W_1, W_2, \cdots, W_H)$. Here $W_h \in \mathbb{R}^{|\mathcal{A}| \times d}$ for each $h \in [H]$. Given an observation $x \in \mathbb{R}^d$ at time step h, the policy takes the action $\pi(x) = \arg\max_{a \in \mathcal{A}} (W_h x)_a$.

We represent the state abstraction function $\phi : \mathcal{X} \to [N]$ using a linear model $A \in \mathbb{R}^{N \times d}$. We next describe the model class $\mathcal{F}_N$ in the context of an input transition (x, a, x'). Instead of using a single model class with bottlenecks on both x and x', we consider two different model classes based on where we place the bottlenecks to simplify the learning problem. We first describe the model class with bottleneck only on x': we map x to a vector x using a linear model $B \in \mathbb{R}^{M \times d}$, we map each action a to its one hot encoding

[4] A Hadamard matrix is a square matrix with entries as either +1 or -1 and whose rows are mutually orthogonal. Sylvester's method is an iterative procedure for creating Hadamard matrix H_k of order 2^k. Sylvester's approach defines $H_1 = [1]$ and $H_{2k} = \begin{bmatrix} H_k & H_k \\ H_k & -H_k \end{bmatrix}$ for $k \in \mathbb{N}$. In our setting, we used a Hadamard matrix of size 2^q with $q = \lceil \log_2(H+4) \rceil$ and pad the state vector with $2^q - H - 4$ zeros.

[5] For comparison, 10^{100} is more than the current estimate of the total number of elementary particles in the universe.

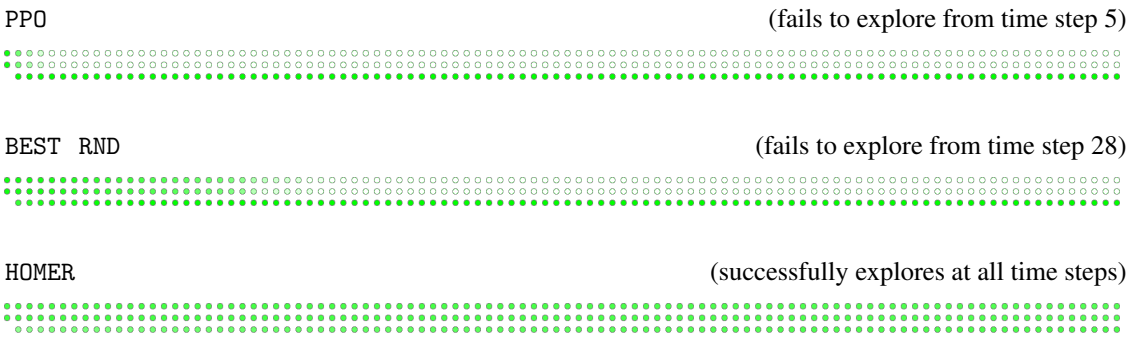


Figure 7: Visualization of the visitation probabilities for algorithms on the diabolical combination lock problem. The top row, middle row and the bottom row represent states in $\{s_{h,a}\}_{h=1}^{100}$, $\{s_{h,b}\}_{h=1}^{100}$ and $\{s_{h,c}\}_{h=1}^{100}$ respectively. The h^{th} column represents states reachable at time step h. We do not show observations or transition edges for brevity. We sample $100,000$ episodes uniformly through the execution of the algorithm and compute the number of time $\texttt{count}[s]$ the agent visits a state s. The count statistics is shown using the opacity of the fill of each state. Formally, we set opacity of s as $\propto \ln(\texttt{count}[s]+1)$. The more opaque the circles are the more frequently the agent visits them. HOMER is able to explore well for all time steps unlike baselines. The BEST RND baseline is the best algorithm (using PPO) with random distillation bonus.

$\mathbf{1}_a$, and we map x' to Ax'. We use a Gumbel softmax to convert Ax' to a probability distribution [22]. Formally, we compute $z_i \propto \exp((Ax')_i + g_i)$ where g_i's are independent Gumbel noise. We define $\phi(x') = \arg\max_{i\in[N]} z_i$. We concatenate $\mathtt{x}, \mathbf{1}_a, z$ and map it to a probability distribution using a two layer feed-forward neural network with leaky rectified linear units and a softmax final layer [35].[6] We learn all linear models for representing policies (W_i), weight matrices A, B and parameters of the feed-forward network. For defining the model class with bottleneck on x, we do the same operations as above but switch the roles of x and x' i.e., we apply Gumbel softmax on x instead of x'. We allow the capacity of the forward abstraction model (M) and the backward abstraction model (N) to be different.

We make three empirically motivated deviations from the theoretical analysis besides the model changes. First, we optimize cross-entropy loss instead of squared loss in Algorithm 3, line 11. Second, we solve the contextual bandit problem in Algorithm 2, line 6 by training the policy model to predict the reward for taking an action using mean squared loss. Lastly, we modify our data collection procedure for training the state abstraction model to make it more sample efficient. We first collect a set of observed transitions $\{(x_i, a_i, x'_i)\}_{i=1}^n$ using our sampling procedure (Algorithm 2, line 5). When we are training the model with bottleneck on x', we create a set of imposter transition candidates $\{(x_i, a_i, \tilde{x}_i)\}_{i=1}^n$ using this set, where $\tilde{x}_i$ is independently and randomly uniformly chosen from $\{x'_1, \cdots, x'_n\}$ for every $i \in [n]$. When we are training the model with bottleneck on x we create a set of imposter transitions candidates $\{(\tilde{x}_i, \tilde{a}_i, x_i)\}_{i=1}^n$ where $(\tilde{x}_i, \tilde{a}_i)$ is independently and randomly uniformly chosen from $\{(x_1, a_1), \ldots, (x_n, a_n)\}$ for every $i \in [n]$. We also found that the learning is more stable if we initialize the state abstraction model by first training without the discretization step. This is achieved by removing the Gumbel-Softmax step. For full details on optimization and hyperparameter values see Appendix G.

[6] We use a softmax final layer since we empirically train our model using cross-entropy loss instead of squared loss. See Appendix G for details.

Baselines. Two baselines which we compare our method against are Advantage Actor Critic (A2C) [37] and Proximal Policy Optimization (PPO) [44]. By default, these algorithms use naive exploration strategies based on entropy bonuses which are often insufficient for very sparse exploration problems. Therefore, we also augment them with an exploration bonus based on Random Network Distillation (RND) [11]. These are denoted A2C + RND and PPO + RND. The RND exploration bonus consists of randomly initializing a neural network mapping observations to an embedding space, which is kept fixed, and training a second network to predict the outputs of the first network on the observations observed by the agent. The prediction error of this network on the current observation is then used as an auxiliary reward signal. The motivation is that the prediction error will tend to be low on observations which have been experienced many times and high on rare observations, thus rewarding the agent for reaching novel states.

We also consider the model-based algorithm (PCID) of Du et al. [16] which provides PAC RL guarantees on Block MDPs satisfying a certain margin condition. This margin condition is violated by the *diabolical combination lock* and, therefore, the theoretical guarantees of Du et al. [16] do not apply. However, experimentally we want to check whether this is an empirical limitation as well as a theoretical one. We use publicly available code for running these baselines. For details see Appendix G.

Results. Figure 6a reports results on different horizons. For an given horizon (H), we report the number of episodes needed to achieve 0.5 times the optimal value. We run each algorithm for a maximum of 10 million episodes. A2C and PPO fail to solve the problem for $H > 3$, and do not escape the local minimum of simply collecting the anti-shaped reward. This is expected since these algorithms do not address the problem of strategic exploration. PCID fails to solve the problem for any horizon length as it cannot learn to separate observations from $s_{h,a}$ and $s_{h,b}$ for any value of h. As PCID defines the policy class over abstract states, the agent is forced to take a single action in both $s_{h,a}$ and $s_{h,b}$ which leads to exploration failure. Our experiments thus confirm that the margin condition for PCID is both an empirical and theoretical limitation. A2C + RND and PPO + RND both obtain respectable performance and are able to solve problems with horizon up to 25. However, they fail for longer horizons of 50 and 100. HOMER is the only algorithm able to solve the problem for all horizon lengths tested.

Figure 6b shows the average return per episode throughout training for $H = 100$. None of the baselines learn the optimal policy which has a value of 1.0. The plot for HOMER shows three plateaus corresponding to three different phases of the algorithm: (i) learning a policy cover, (ii) learning the reward-sensitive policy $\hat{\pi}$ and (iii) sampling actions from $\hat{\pi}$.

We visualize the visitation probabilities for the best performing algorithms in Figure 7. The result demonstrates that the visitation probabilities for HOMER don't decay significantly with time steps as they do for baselines.

We plot the recovered state abstraction and dynamics in Figure 8 for the first 10 time steps for the setting with horizon of 100. We use the learned forward and backward abstractions at each time step to recover the state abstraction. We use the learned policy cover for generating trajectories on which we do count-based estimation to learn the transition dynamics.[7] As Figure 8 shows, HOMER is able to recover the canonical latent transition dynamics.[8] For all 100 time steps, we found that HOMER only made mistake once by merging observations from two kinematically separable states into one abstract state. Note that we can only hope to recover a model that has the same canonical form as the actual model since we can always treat a state in a

[7] Formally, let $\{\Psi_h\}_{h=1}^H$ be the learned policy cover and $\{\overline{\phi}_h\}_{h=1}^H$ be the learned state abstraction which is a composition of both forward and backward abstractions. For time step h, we draw n transitions $x_i, a_i, x' \sim \texttt{Unf}(\Psi_{h-1}) \circ \texttt{Unf}(\mathcal{A})$. The model $\hat{T}_h(j \mid i, a)$ for time step h is given by: $\hat{T}(j \mid i, a) = \frac{\sum_{k=1}^n \mathbf{1}\{\overline{\phi}_h(x'_k)=j, a_k=a, \overline{\phi}_{h-1}(x_k)=i\}}{\sum_{k=1}^n \mathbf{1}\{a_k=a, \overline{\phi}_{h-1}(x_k)=i\}}$ for any learned abstract states j, i and action a.

[8] Recall that canonical form of a Block MDP is created by merging kinematically inseparable states into one state.

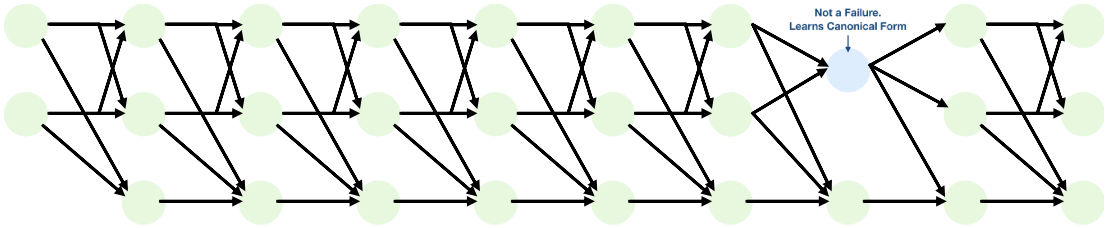


Figure 8: Recovered transition dynamics for the first 10 time steps of the diabolical combination lock problem with horizon of 100. The abstract states in the first two rows correspond to $\{s_{h,a} \mid h \in [10]\}$ and $\{s_{h,b} \mid h \in [10]\}$ while those in the last row correspond to $\{s_{h,c} \mid h \in [10]\}$. At time step 8, the model maps observations from $\{s_{8,a}, s_{8,b}\}$ to the same abstract state since actions u_8 and v_8 are the same, which makes these states kinematically inseparable.

Block MDP as multiple kinematically inseparable states. HOMER does not use the learned model for planning to avoid cascading errors arising due to inaccurate state abstraction functions. However, utilizing the learned dynamics for model-based planning is a natural alternative direction.

9 Conclusion

We present HOMER, a model-free reinforcement learning algorithm for rich observation environments. Theoretically, we prove that HOMER learns both a policy cover and a near optimal policy for any Block MDP using only polynomially many trajectories and in polynomial time, under certain computational and expressivity assumptions. Empirically, we show that HOMER can solve a difficult exploration problem in a tractable manner, and even recover the latent transition dynamics. Experimenting with HOMER on real-world reinforcement learning scenarios is a future work direction.

References

[1] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert E Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, 2014.

[2] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. *arXiv:1908.00261*, 2019.

[3] Noga Alon, Shai Ben-David, Nicolo Cesa-Bianchi, and David Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM*, 1997.

[4] András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 2008.

[5] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, 2017.

[6] J Andrew Bagnell, Sham M Kakade, Jeff G Schneider, and Andrew Y Ng. Policy search by dynamic programming. In *Advances in Neural Information Processing Systems*, 2004.

[7] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, 2016.

[8] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 2013.

[9] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford University Press, 2013.

[10] Ronen I. Brafman and Moshe Tennenholtz. R-MAX - A general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 2002.

[11] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019.

[12] Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, 2019.

[13] Richard Y Chen, Szymon Sidor, Pieter Abbeel, and John Schulman. UCB exploration via Q-Ensembles. *arXiv:1706.01502*, 2017.

[14] Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

[15] Christoph Dann, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. On oracle-efficient PAC RL with rich observations. In *Advances in Neural Information Processing Systems*, 2018.

[16] Simon S Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudík, and John Langford. Provably efficient RL with rich observations via latent state decoding. In *International Conference on Machine Learning*, 2019.

[17] Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 2014.

[18] Amir-massoud Farahmand, Csaba Szepesvári, and Rémi Munos. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems*, 2010.

[19] Elad Hazan, Sham M Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. *arXiv:1812.02690*, 2018.

[20] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *International Conference on Computer Vision*, 2017.

[21] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 2010.

[22] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2016.

[23] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low Bellman rank are PAC-learnable. In *International Conference on Machine Learning*, 2017.

[24] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, 2018.

[25] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. The malmo platform for artificial intelligence experimentation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

[26] Nicholas K. Jong and Peter Stone. State abstraction discovery from irrelevant state variables. In *International Joint Conference on Artificial Intelligence*, 2005.

[27] Sham M Kakade. *On the sample complexity of reinforcement learning*. PhD thesis, University College London, 2003.

[28] Sham M Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002.

[29] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 2002.

[30] Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, 2016.

[31] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems*, 2008.

[32] Tor Lattimore and Marcus Hutter. PAC bounds for discounted MDPs. In *Conference on Algorithmic Learning Theory*, 2012.

[33] Lihong Li, Thomas J. Walsh, and Michael L. Littman. Towards a unified theory of state abstraction for MDPs. In *International Symposium on Artificial Intelligence and Mathematics*, 2006.

[34] Tengyuan Liang, Alexander Rakhlin, and Karthik Sridharan. Learning with square loss: Localization through offset Rademacher complexity. In *Conference on Learning Theory*, 2015.

[35] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[36] Andrew Mccallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, The University of Rochester, 1996.

[37] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016.

[38] Rémi Munos. Error bounds for approximate policy iteration. In *International Conference on Machine Learning*, 2003.

[39] Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *The Journal of Machine Learning Research*, 2008.

[40] Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Near-optimal representation learning for hierarchical reinforcement learning. In *International Conference on Learning Representations*, 2019.

[41] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017.

[42] Balaraman Ravindran. *An Algebraic Approach to Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2004.

[43] Stephane Ross and J Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv:1406.5979*, 2014.

[44] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[45] Zhang Shangtong. Modularized implementation of deep RL algorithms in PyTorch. https://github.com/ShangtongZhang/DeepRL, 2018.

[46] Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from logged implicit exploration data. In *Advances in Neural Information Processing Systems*, 2010.

[47] Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 2008.

[48] Alexander L. Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L. Littman. PAC model-free reinforcement learning. In *International Conference on Machine Learning*, 2006.

[49] Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, 2015.

[50] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. #Exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

[51] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning, 2012.

Appendices

Symbol	Definition
$[N]$	Defines the set $\{1, 2, \cdots, N\}$ for any $N \in \mathbb{N}$.
$\Delta(\mathcal{U})$	The space of probability distribution over the set $\mathcal{U}$.
$\texttt{Unf}(\mathcal{U})$	A uniform distribution over the set $\mathcal{U}$.
$\text{supp}(p)$	Support of a distribution. For any $p \in \Delta(\mathcal{U})$, we have $\text{supp}(p) := \{u \in \mathcal{U} \mid p(u) > 0\}$.
$\mathcal{M}$	Denotes a Block Markov Decision Process (Block MDP).
H	Number of actions the agent takes for any episode.
$\mathcal{S}$	A finite state space of $\mathcal{M}$. The process is layered, so states also encode the time step.
$\mathcal{S}_h$	The set of states reachable at time step h. $\mathcal{S} := \cup_{h\in[H]}\mathcal{S}_h$.
$\mathcal{X}$	The observation space. May be infinitely large, but is assumed to be countable.
$\mathcal{X}_h$	Set of observations reachable at time step h. $\mathcal{X} := \cup_{h\in[H]}\mathcal{X}_h$.
$\mathcal{A}$	The finite discrete action space of $\mathcal{M}$
q	An emission function $q : s \to \Delta(\mathcal{X})$. $q(x \mid s)$ denotes the probability of observing x in state s. Note that $\text{supp}(q(\cdot\|s)) \cap \text{supp}(q(\cdot\|s')) = \emptyset$ when $s \neq s'$.
$g^\star$	A decoder function $g^\star : \mathcal{X} \to \mathcal{S}$. $g^\star(x) = s$ iff $q(x\|s) > 0$.
$\mu(s)$	Probability of starting in state s at the beginning of any episode.
$T(s' \mid s, a)$	The probability of transitioning to $s' \in \mathcal{S}$ when taking action $a \in \mathcal{A}$ in $s \in \mathcal{S}$.
π	A policy $\pi : \mathcal{X} \to \Delta(\mathcal{A})$, which may or may not be stationary.
$(\pi_1, \ldots, \pi_h)$	A h-step policy where the t^{th} action ($1 \le t \le h$) is taken according to π_t.
Π	The policy class, $\Pi : \mathcal{X} \to \Delta(\mathcal{A})$.
Π_{NS}	The set of non-stationary policies: $\Pi_{\text{NS}} := \{(\pi_1, \ldots, \pi_H) : \pi_t \in \Pi\}$.
$\mathbb{P}_\pi(s)$	Probability of visiting s when following π, from the starting distribution μ.
$\pi_s^\star$	Homing policy of the state s, $\pi_s^\star := \arg\max_{\pi\in\Pi_{\text{NS}}} \mathbb{P}_\pi(s)$. Due to Lemma 2, we take $\pi_s^\star$ to be deterministic and non-stationary.
$\pi_x^\star$	Analogous homing policy of the observation x, $\pi_x^\star := \arg\max_{\pi\in\Pi_{\text{NS}}} \mathbb{P}_\pi(x)$ It is easy to see from the Block MDP assumption that $\pi_x^\star = \pi_s^\star$ where $s = g^\star(x)$.
$V(\pi; R)$	Value for (non-stationary) policy π on reward function R from starting distribution μ. R may have type $R : \mathcal{X} \to \mathbb{R}$ or $R : (\mathcal{X} \times \mathcal{A}) \to \mathbb{R}$ and may also be stochastic.
$V(s; \pi, R)$ and $V(x; \pi, R)$	Value functions for π on R, defined over states or observations.
$\eta(s)$	Maximum visitation probability for state s, $\eta(s) := \sup_{\pi\in\Pi} \mathbb{P}_\pi(s)$.
η_{min}	Reachability parameter, $\eta_{min} := \min_{s\in\mathcal{S}} \eta(s)$.
$\mathcal{F}$	Regressor class containing functions f from $\mathcal{X} \times \mathcal{A} \times \mathcal{X} \to [0, 1]$.
Φ_N	Decoder function class $\Phi_N : \mathcal{X} \to [N]$, for a fixed N.
τ	Time function that maps states/observations to the time step where they are reachable, which is well-defined due to the layered assumption.

Table 1: List of notations and brief definitions. Operators defined on states are lifted to observations in the natural way, e.g., $\mu(x) := \sum_s \mu(s)q(x \mid s)$.

Notation and Overview. See Table 1 for an overview of the notation and definitions used in this appendix. The appendix is structured as follows:

- Appendix A: Properties of homing policies;

- Appendix B: Properties of kinematic inseparability;
- Appendix C: Basic results for Policy Search from Dynamic Programming (PSDP);
- Appendix D: Analysis of the `ExpOracle` algorithm;
- Appendix E: Analysis of the `HOMER` algorithm;
- Appendix F: Supporting results;
- Appendix G: Experimental details.

A Properties of Homing Policies

In this section we prove some basic properties of homing policies. For this section only, we consider a fully expressive policy set $\Upsilon := (\mathcal{X} \to \Delta(\mathcal{A}))$. We further define the set of *all* deterministic policies $\Upsilon_{\text{det}} := (\mathcal{X} \to \mathcal{A})$. Clearly we have $\Upsilon_{\text{det}} \subset \Upsilon$. Note that both of these classes contain non-stationary policies due to the layered structure of the environments we consider. In particular, we have $(\pi_1, \ldots, \pi_H) \in \Upsilon$ whenever $\pi_h \in \Upsilon$ for all h, with a similar statement for Υ_{det}.

The first result is that for every state, there exists a *deterministic non-stationary* policy $\pi \in \Upsilon_{\text{det}}$ that is a homing policy for that state. This motivates our decision to restrict our search to only these policies in experiments. The result also appears in [6], but we provide a proof for completeness.

Lemma 2. *For any, possibly stochastic, reward function R, we have*

$$\max_{\pi \in \Upsilon_{det}} V(\pi; R) = \max_{\pi \in \Upsilon} V(\pi; R),$$

where $V(\pi; R)$ is the value for policy π under reward function R. In particular, the result holds for $R(x) = \mathbf{1}\{g^\star(x) = s\}$, for any $s \in \mathcal{S}$, which yields:

$$\max_{\pi \in \Upsilon_{det}} \mathbb{P}_\pi(s) = \max_{\pi \in \Upsilon} \mathbb{P}_\pi(s).$$

Proof. As $\Upsilon_{\text{det}} \subset \Upsilon$, that the LHS is at most the RHS is obvious. We are left to establish the other direction.

The proof is a simple application of dynamic programming. Assume inductively that there exists a policy $\tilde{\pi}_{h:H} \in \Upsilon_{\text{det}}$ such that, for any distribution $Q \in \Delta(\mathcal{X}_h)$, we have

$$\mathbb{E}_{x_h \sim Q}\left[V(\tilde{\pi}_{h:H}; R, x_h)\right] \geq \max_{\pi_{h:H} \in \Upsilon} \mathbb{E}_{x_h \sim Q}\left[V(\pi_{h:H}; R, x_h)\right],$$

where the value function here denotes the future reward, according to R, when executing the policy from the starting observation x_h. The base case is when $h = H$, in which case, it is easy to verify that the claim holds for the policy $\tilde{\pi}_H(x_H) := \text{argmax}_{a \in \mathcal{A}} \mathbb{E}[R \mid x_H]$.

Define the policy component for time step $h-1$ as

$$\forall x_{h-1} \in \mathcal{X}_{h-1} : \tilde{\pi}_{h-1}(x_{h-1}) := \operatorname*{argmax}_{a \in \mathcal{A}} \mathbb{E}[R + V(\tilde{\pi}_{h:H}; R, x_h) \mid x_{h-1}].$$

Now for any potentially stochastic policy $\pi_{h-1:H}$, and any distribution $Q \in \Delta(\mathcal{X}_{h-1})$ we have

$$\begin{aligned}
\mathbb{E}_{x_{h-1} \sim Q}\left[V(\tilde{\pi}_{h-1:H}; R, x_{h-1})\right] &\geq \mathbb{E}_{x_{h-1} \sim Q}\left[R + V(\tilde{\pi}_{h:H}; R, x_h) \mid a \sim \pi_{h-1}\right] \\
&\geq \mathbb{E}_{x_{h-1} \sim Q}\left[R + V(\pi_{h:H}; R, x_h) \mid a \sim \pi_{h-1}\right] \\
&= \mathbb{E}_{x_{h-1} \sim Q}\left[V(\pi_{h:H}; R, x_{h-1})\right].
\end{aligned}$$

This proves the inductive step. We conclude the proof by noting that $V(\pi; R) = \mathbb{E}_{x_1 \sim \mu} V(\pi; R, x_1)$, for which we have established the optimality guarantee for $\tilde{\pi}_{1:H}$. □

We also observe that homing policies do not grow compositionally. In other words, we may not be able to construct homing policies for states $\mathcal{S}_h$, by appending a one-step policy to the homing policies for $\mathcal{S}_{h-1}$. Note that this holds even when working with the unrestricted policy class Υ. This observation justifies the global policy search procedure PSDP for finding the homing policies.

For the statement, for a policy subset Π', we use the notation $\Delta(\Pi')$ to denote the set of *mixture policies* that, on each episode samples a policy $\pi \in \Pi'$ from a distribution and executes that policy. Note that this is not the same as choosing a new policy from the distribution on a per time-step basis.

Lemma 3. *There exists a Block MDP $\mathcal{M}$ a time step $h \in [H]$ and a state $s \in \mathcal{S}_h$ such that*

$$\eta(s) > \sup_{\pi_{mix} \in \Delta(\{\pi^\star_s\}_{s \in \mathcal{S}_{h-1}}), \pi_h \in \Upsilon} \mathbb{P}_{\pi_{mix} \circ_h \pi_h}(s).$$

Here π_{mix} is a mixture policy *over the homing policies $\{\pi^\star_s\}_{s \in \mathcal{S}_{h-1}}$ for the states at time step $h-1$, and $\circ_h$ denotes policy composition at time h.*

Proof. See Figure 2b. The homing policy for s_5 takes action a_1 in s_1, which yields a visitation probability for s_5 of 1. However, the homing policies for states s_2, s_3, and s_4 do not take action a_1 in s_1. □

B Properties of Kinematic Inseparability

In this section, we establish several useful properties for kinematically inseparable (KI) state abstractions. Recall Definition 3, Definition 4, and Definition 5.

Fact 3. *Forward kinematic inseparability (KI), backward KI and KI defines an equivalence relation on $\mathcal{X}$.*

Proof. That these relations are reflexive, symmetric, and transitive, all follow trivially from the definitions, in particular using the fact that equality itself is symmetric and transitive. □

Lemma 4. *Let $x_1, x_2 \in \mathcal{X}$. If $g^\star(x_1) = g^\star(x_2)$ then x_1 and x_2 are KI. This implies that they are also forward KI and backward KI.*

Proof. Fix any $x \in \mathcal{X}$, $a \in \mathcal{A}$ and $u \in \Delta(\mathcal{X} \times \mathcal{A})$ with $\text{supp}(u) = \mathcal{X} \times \mathcal{A}$. We show below that x_1 and x_2 are forward KI and backward KI which together establishes that desired claim.

Forward KI: By the Block MDP structure, we have

$$T(x \mid x_1, a) = T(x \mid g^\star(x_1), a) = T(x \mid g^\star(x_2), a) = T(x \mid x_2, a)$$

Backward KI: Again, using the Block MDP structure:

$$\begin{aligned}
\mathbb{P}_u(x, a \mid x_1) &= \frac{T(x_1 \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[T(x_1 \mid \tilde{x}, \tilde{a})\right]} = \frac{q(x_1 \mid g^\star(x_1))T(g^\star(x_1) \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[q(x_1 \mid g^\star(x_1))T(g^\star(x_1) \mid \tilde{x}, \tilde{a})\right]} \\
&= \frac{T(g^\star(x_1) \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[T(g^\star(x_1) \mid \tilde{x}, \tilde{a})\right]} = \frac{T(g^\star(x_2) \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[T(g^\star(x_2) \mid \tilde{x}, \tilde{a})\right]} \\
&= \frac{q(x_2 \mid g^\star(x_2))T(g^\star(x_1) \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[q(x_2 \mid g^\star(x_2))T(g^\star(x_1) \mid \tilde{x}, \tilde{a})\right]} = \frac{T(x_2 \mid x, a)u(x, a)}{\mathbb{E}_{(\tilde{x}, \tilde{a}) \sim u}\left[T(x_2 \mid \tilde{x}, \tilde{a})\right]} = \mathbb{P}_u(x, a \mid x_2).
\end{aligned}$$

□

The next simple fact shows that observations that appear at different time points are always separable.

Fact 4. *If x, x' are forward or backward KI, then $\tau(x) = \tau(x')$, where recall that $\tau(x)$ denotes the time step where x is reachable.*

Proof. If $h := \tau(x) \neq \tau(x') := h'$ then $T(\cdot \mid x, a) \in \Delta(\mathcal{X}_{h+1})$ while $T(\cdot \mid x', a) \in \Delta(\mathcal{X}_{h'+1})$, so these distributions cannot be equal. A similar argument holds for Backward KI. □

Using the transitivity property for backward KI, we can consider sets of observations that are all pairwise backward KI. The next lemma provides a convenient characterization for backward KI sets.

Lemma 5. *Let $\mathcal{X}' \subseteq \mathcal{X}$ be a set of backward KI observations. Then $\exists u \in \Delta(\mathcal{X})$ with supp$(u) = \mathcal{X}$ such that for all $x', x'' \in \mathcal{X}'$ we have:*

$$\forall x \in \mathcal{X}, a \in \mathcal{A}, \qquad \frac{T(x' \mid x, a)}{u(x')} = \frac{T(x'' \mid x, a)}{u(x'')}. \tag{1}$$

The converse is also true: if (1) holds for some $u \in \Delta(\mathcal{X})$ with full support and all $x', x'' \in \mathcal{X}' \subset \mathcal{X}$ then $\mathcal{X}'$ are is a backward KI set.

Proof. Fix $\tilde{u} \in \Delta(\mathcal{X} \times \mathcal{A})$ with supp$(\tilde{u}) = \mathcal{X} \times \mathcal{A}$. Define $u(x) := \mathbb{E}_{(\tilde{x},\tilde{a})\sim\tilde{u}}\left[T(x \mid \tilde{x}, \tilde{a})\right]$. Observe that by construction $u(x) > 0$ for all $x \in \mathcal{X}$. Let $x', x'' \in \mathcal{X}'$ then as x' and x'' are backward KI, we have that for all $x \in \mathcal{X}, a \in \mathcal{A}$:

$$\begin{aligned}\mathbb{P}_{\tilde{u}}(x, a \mid x'') = \mathbb{P}_{\tilde{u}}(x, a \mid x') &\Rightarrow \frac{T(x'' \mid x, a)\tilde{u}(x, a)}{\mathbb{E}_{(\tilde{x},\tilde{a})\sim\tilde{u}}\left[T(x'' \mid \tilde{x}, \tilde{a})\right]} = \frac{T(x' \mid x, a)\tilde{u}(x, a)}{\mathbb{E}_{(\tilde{x},\tilde{a})\sim\tilde{u}}\left[T(x' \mid \tilde{x}, \tilde{a})\right]} \\ &\Rightarrow \frac{T(x'' \mid x, a)}{u(x'')} = \frac{T(x' \mid x, a)}{u(x')}.\end{aligned}$$

For the converse, let $\tilde{u} \in \Delta(\mathcal{X} \times \mathcal{A})$ have full support. Then we have

$$\mathbb{P}_{\tilde{u}}(x, a \mid x'_1) = \frac{T(x'_1 \mid x, a)\tilde{u}(x, a)}{\sum_{\tilde{x},\tilde{a}} T(x'_1 \mid \tilde{x}, \tilde{a})\tilde{u}(\tilde{x}, \tilde{a})} = \frac{\frac{u(x'_1)}{u(x'_2)}T(x'_2 \mid x, a)\tilde{u}(x, a)}{\sum_{\tilde{x},\tilde{a}} \frac{u(x'_1)}{u(x'_2)}T(x'_2 \mid \tilde{x}, \tilde{a})\tilde{u}(\tilde{x}, \tilde{a})} = \mathbb{P}_{\tilde{u}}(x, a \mid x'_2),$$

and so x_1, x_2 are backward KI. □

We next show that an ordering relation between policy visitation probabilities is preserved through backward KI. This key structural property allows us to use the backward KI relationship to find a policy cover.

Lemma 6. *Let $\mathcal{X}' \subseteq \mathcal{X}$ be a set of backward KI observations and let $\mathcal{X}'_1, \mathcal{X}'_2 \subset \mathcal{X}'$. For any $\pi_1, \pi_2 \in \Upsilon$, we have $\frac{\mathbb{P}_{\pi_1}(\mathcal{X}'_1)}{\mathbb{P}_{\pi_2}(\mathcal{X}'_1)} = \frac{\mathbb{P}_{\pi_1}(\mathcal{X}'_2)}{\mathbb{P}_{\pi_2}(\mathcal{X}'_2)}$.*

Proof. Assume that $\mathcal{X}' \subseteq \mathcal{X}_h$ for some h, which is without loss of generality, since if they are observable at different time steps, then they are trivially separable. From Lemma 5, there exists a $u \in \Delta(\mathcal{X})$ supported everywhere such that for any $x'_1, x'_2 \in \mathcal{X}'$ we have: $\frac{T(x'_1|x,a)}{u(x'_1)} = \frac{T(x'_2|x,a)}{u(x'_2)}$ for any $x \in \mathcal{X}$ and $a \in \mathcal{A}$. Let π be any policy and define its occupancy measure at time $h-1$, $\xi_{h-1} \in \Delta(\mathcal{X}_{h-1} \times \mathcal{A})$, as $\xi_{h-1}(x, a) := \mathbb{E}_\pi\left[\mathbf{1}\{x_{h-1} = x, a_{h-1} = a\}\right]$. Then for any fixed $\tilde{x} \in \mathcal{X}'$ and $j \in \{1, 2\}$ we have

$$\mathbb{P}_\pi(\mathcal{X}'_j) = \mathbb{E}_{(x,a)\sim\xi}\left[\sum_{x' \in \mathcal{X}'_j} T(x' \mid x, a)\right] = \mathbb{E}_{x,a\sim\xi}\left[\sum_{x' \in \mathcal{X}'_j} \frac{u(x')}{u(\tilde{x})}T(\tilde{x} \mid x, a)\right] = \frac{u(\mathcal{X}'_j)}{u(\tilde{x})}\mathbb{P}_\pi(\tilde{x}),$$

where the second inequality follows from Lemma 5. Re-arranging, we have that $\frac{\mathbb{P}_\pi(\mathcal{X}'_1)}{\mathbb{P}_\pi(\mathcal{X}'_2)} = \frac{u(\mathcal{X}'_1)}{u(\mathcal{X}'_2)}$, and as the right hand side does not depend on π, the result follows. □

Lastly, we show that a set of observations are backward KI, then a single policy simultaneously maximizes the visitation probability to these observations. Moreover, we can construct a reward function for which this common policy is the reward maximizer. Recall that Υ is the (unrestricted) set of *all* policies.

Lemma 7. *Let $\mathcal{X}' \subseteq \mathcal{X}$ be a set of backward kinematically inseparable observations. Then there exists a policy $\pi \in \Upsilon$ that maximizes $\mathbb{P}_\pi(x')$ simultaneously for all $x' \in \mathcal{X}'$. Further, this policy is the optimal policy for the internal reward function $R'(x,a) := 1\{x \in \mathcal{X}'\}$.*

Proof. Let $x_1, x_2 \in \mathcal{X}'$ and define $\pi := \operatorname{argmax}_{\pi \in \Upsilon} \mathbb{P}_\pi(x_1)$. Let $\pi_2 \in \Upsilon$ be any other policy. Then, by Lemma 6, we have

$$\mathbb{P}_\pi(x_1) \geq \mathbb{P}_{\pi_2}(x_1) \Leftrightarrow \mathbb{P}_\pi(x_2) \geq \mathbb{P}_{\pi_2}(x_2).$$

As the left hand side is true by definition of π, we see that π also maximizes the visitation probability for x_2. As this is true for any x_2, we have that π simultaneously maximizes the visitation probability for all $x \in \mathcal{X}'$.

Clearly, for this policy and the specified reward function R', we have

$$\mathbb{E}_\pi\left[\sum_h R'(x_h, a_h)\right] = \mathbb{P}_\pi(\mathcal{X}') = \sum_{x' \in \mathcal{X}'} \mathbb{P}_\pi(x') \geq \max_{\pi' \in \Upsilon} \sum_{x' \in \mathcal{X}'} \mathbb{P}_{\pi'}(x') = \max_{\pi' \in \Upsilon} \mathbb{E}_{\pi'}\left[\sum_h R'(x_h, a_h)\right].$$

Here we are assuming $\mathcal{X}$ is countable, as we have mentioned. □

C Analysis of Policy Search by Dynamic Programming

This section provides a detailed statistical and computation analysis of the Policy Search by Dynamic Programming (PSDP) algorithm, with pseudocode in Algorithm 2. The main guarantee is as follows:

Theorem 5. *Let $(\hat{\pi}_1, \hat{\pi}_2, \cdots, \hat{\pi}_h) = \texttt{PSDP}(\Psi, R, h, \Pi, n)$ be the policy returned by Algorithm 2 using policy covers $\Psi = \{\Psi_t\}_{t=1}^h$ where Ψ_t is an α-policy cover for $\mathcal{S}_t$ and $|\Psi_t| \leq N$ for all $t \in [h]$. Assume that either R is an internal reward function corresponding to time $h+1$, or that R is the external reward function and $h = H$, and that Assumption 2 holds. Then for any $\delta \in (0,1)$, with probability at least $1 - h\delta$ we have:*

$$V(\hat{\pi}_{1:h}; R) \geq \max_{\pi_1, \ldots, \pi_h \in \Pi} V(\pi_{1:h}; R) - \frac{Nh\Delta_{csc}}{\alpha} \qquad \textit{where} \qquad \Delta_{csc} := 4\sqrt{\frac{|\mathcal{A}|}{n} \ln\left(\frac{2|\Pi|}{\delta}\right)}.$$

The algorithm runs in polynomial time with h calls to the contextual bandit oracle.

Before turning to the proof, we state a standard generalization bound for the contextual bandit problems created by the algorithm. These problems are induced by an underlying distribution Q over tuples $(x, \vec{r})$ where $x \in \mathcal{X}$ and $\vec{r} \in [0,1]^{|\mathcal{A}|}$, and a *logging policy* $\pi_{\log}$. Formally, we obtain tuples $(x, a, p, r) \sim Q_{\log}$ where $(x, \vec{r}) \sim Q$, $a \sim \pi_{\log}(x)$, $p := \pi_{\log}(a \mid x)$ is the probability of choosing the action for the current observation, and $r := \vec{r}(a)$. In our application, we always have $\pi_{\log} := \texttt{Unf}(\mathcal{A})$ so that $p = 1/|\mathcal{A}|$. Given a dataset of n tuples $D := \{(x_i, a_i, p_i, r_i)\}_{i=1}^n \overset{iid}{\sim} Q_{\log}$, we invoke the contextual bandit oracle, $\texttt{CB}(D, \Pi)$, to find a policy $\hat{\pi}$. The following proposition provides a performance guarantee for $\hat{\pi}$.

Proposition 6. *Let $D := \{(x_i, a_i, p_i, r_i)\}_{i=1}^n \overset{iid}{\sim} Q_{log}$ be a dataset of n samples from a contextual bandit distribution Q_{log} induced by the uniform logging policy interacting with an underlying distribution Q. Let $\hat{\pi} = \texttt{CB}(D, \Pi)$. Then for any $\delta \in (0,1)$, with probability at least $1-\delta$, we have*

$$\mathbb{E}_{(x,\vec{r})\sim Q}[\vec{r}(\hat{\pi}(x)] \geq \max_{\pi\in\Pi} \mathbb{E}_{(x,\vec{r})\sim Q}[\vec{r}(\pi(x)] - \Delta_{csc}.$$

Proof. The proof is a standard generalization bound for contextual bandits [c.f., 31]. We provide a short proof for completeness.

For policy π, define $R(\pi) := \mathbb{E}_Q\left[\vec{r}(\pi(x))\right]$, $\hat{r}_i(\pi) = \frac{\mathbf{1}\{a_i=\pi(x_i)\}r_i}{p_i} = |\mathcal{A}|\mathbf{1}\{a_i = \pi(x_i)\}r_i$, and observe that the contextual bandit oracle finds the policy that maximizes $\hat{R}(\pi) := \frac{1}{n}\sum_{i=1}^n \hat{r}_i(\pi)$. The random variables $\hat{r}_i(\pi)$ satisfy the following useful properties:

$$\text{Unbiased: } \mathbb{E}_{Q_{\log}}\left[\hat{r}(\pi)\right] = \mathbb{E}_Q\left[\sum_a \pi_{\log}(a \mid x)\frac{\mathbf{1}\{a=\pi(x)\}\vec{r}(a)}{\pi_{\log}(a \mid x)}\right] = \mathbb{E}_Q\left[\vec{r}(\pi(x))\right].$$

$$\text{Low variance: } \mathrm{Var}[\hat{r}(\pi)] \leq \mathbb{E}_{Q_{\log}}\left[\hat{r}^2(\pi)\right] \leq \mathbb{E}_{Q_{\log}}\left[\frac{\mathbf{1}\{a=\pi(x)\}}{p^2}\right] = |\mathcal{A}| \cdot \mathbb{P}_{Q_{\log}}[a=\pi(x)] = |\mathcal{A}|$$

$$\text{Range: } |\hat{r}(\pi)| \leq |\mathcal{A}|.$$

Therefore, using Bernstein's inequality (Proposition 11) and union bound we have that for any $\delta \in (0,1)$, with probability at least $1-\delta$:

$$\forall \pi \in \Pi, \qquad \left|\hat{R}(\pi) - R(\pi)\right| \leq \frac{2|\mathcal{A}|}{3n}\ln\left(\frac{2|\Pi|}{\delta}\right) + \sqrt{\frac{2|\mathcal{A}|}{n}\ln\left(\frac{2|\Pi|}{\delta}\right)} =: \Delta.$$

The contextual bandit oracle finds $\hat{\pi}$ that maximizes the empirical quantity $\hat{R}(\pi)$, so, by a standard generalization argument, we have

$$R(\hat{\pi}) \geq \hat{R}(\pi) - \Delta \geq \max_{\pi\in\Pi}\hat{R}(\pi) - \Delta \geq \max_{\pi\in\Pi} R(\pi) - 2\Delta.$$

Of course as the reward vector is bounded in $[0,1]$ we always have $R(\hat{\pi}) \geq \max_{\pi\in\Pi} R(\pi) - 1$, which means that with probability at least $1-\delta$, we have

$$R(\hat{\pi}) \geq \max_{\pi\in\Pi} R(\pi) - \min\left\{1, 2\Delta\right\}.$$

Finally, if $2\Delta \leq 1$ then $\frac{4|\mathcal{A}|\ln(2|\Pi|/\delta)}{3n} \leq \sqrt{\frac{4|\mathcal{A}|\ln(2|\Pi|/\delta)}{3n}}$. This observation leads to the definition Δ_{csc}. □

Proof of Theorem 5. Let $\pi_1^\star, \pi_2^\star, \cdots \pi_h^\star = \mathrm{argmax}_{\pi_1,\pi_2,\cdots\pi_h\in\Pi} V(\pi_{1:h}; R)$ be the optimal non-stationary policy, for reward function R with time horizon h. PSDP solves a sequence of h contextual bandit problems to learn policies $\hat{\pi}_t$ for $t = h, \ldots, 1$. The t^{th} problem is induced by a distribution Q_t supported over $\mathcal{X}_t \times [0,1]^{|\mathcal{A}|}$, which is defined inductively as follows: The observations are induced by choosing a $\pi_t \sim \texttt{Unf}(\Psi_t)$ and executing π_t for $t-1$ steps to visit x_t. The reward given x_t and an action $a \in \mathcal{A}$ is $R(x_{h+1})$ where the trajectory is completed by first executing a from x_t and then following $\hat{\pi}_{t+1:h}$. As in Proposition 6, the contextual bandit dataset is induced by this distribution Q_t the uniform logging policy.

By Proposition 6 with probability at least $1-h\delta$, we have that for all t, $\hat{\pi}_t$ satisfies

$$\mathbb{E}_{(x,\vec{r})\sim Q_t}\left[\vec{r}(\hat{\pi}_t(x))\right] \geq \max_{\pi\in\Pi}\mathbb{E}_{(x,\vec{r})\sim Q_t}\left[\vec{r}(\pi_t(x))\right] - \Delta_{csc},$$

where Q_t is as defined above. Using the definition of Q_t, this guarantee may be written as

$$\forall t \in [h] : \mathbb{E}_{x\sim Q_t}\left[V(x;\hat{\pi}_{t:h},R)\right] \geq \max_{\pi\in\Pi} \mathbb{E}_{x\sim Q_t}\left[V(x;\pi\circ\hat{\pi}_{t+1:h},R)\right] - \Delta_{csc}.$$

Define $Q_t^\star \in \Delta(\mathcal{X}_t)$ to be the distribution of observations visited by executing $\pi^\star_{1:t-1}$. By the performance difference lemma (Lemma 13) [6, 27, 43, c.f.,], we have

$$\begin{aligned}
V(\hat{\pi}_{1:h};R) - V(\pi^\star_{1:h};R) &= \sum_{t=1}^{h} \mathbb{E}_{x_t\sim Q_t^\star}\left[V(x_t;\pi_t^\star\circ\hat{\pi}_{t+1:h},R) - V(x_t;\hat{\pi}_{t:h},R)\right] \\
&\leq \sum_{t=1}^{h} \mathbb{E}_{x_t\sim Q_t^\star}\left[V(x_t;\tilde{\pi}_t^\star\circ\hat{\pi}_{t+1:h},R) - V(x_t;\hat{\pi}_{t:h},R)\right],
\end{aligned}$$

where $\tilde{\pi}_t^\star(x) := \operatorname{argmax}_{a\in\mathcal{A}} \mathbb{E}\left[R(x,a) + V(x_{t+1};\hat{\pi}_{t+1:h},R) \mid x_t = x, a_t = a\right]$, for all $x \in \mathcal{X}_t$. With this definition, the inequality here is immediate, by definition of the value function.

Assumption 2 implies that $\tilde{\pi}_t^\star \in \Pi$ for each t, which is immediate for the external reward function. If we are using the internal reward function with some $h < H$, then by construction the internal reward function is defined only at time $h+1$, so we may simply append arbitrary policies $\hat{\pi}_{h+1:H}$ without affecting the reward or the value function. Formally, we have

$$\begin{aligned}
V(\hat{\pi}_{1:h};R) - V(\pi^\star_{1:h};R) &\leq \sum_{t=1}^{h} \mathbb{E}_{x_t\sim Q_t^\star}\left[V(x_t;\tilde{\pi}_t^\star\circ\hat{\pi}_{t+1:H},R) - V(x_t;\hat{\pi}_{t:H},R)\right] \\
&= \sum_{t=1}^{h} \mathbb{E}_{x_t\sim Q_t}\left[\frac{Q_t^\star(x_t)}{Q_t(x_t)}\left(V(x_t;\tilde{\pi}_t^\star\circ\hat{\pi}_{t+1:H},R) - V(x_t;\hat{\pi}_{t:H},R)\right)\right] \\
&\leq \sum_{t=1}^{h} \sup_{x_t}\left|\frac{Q_t^\star(x_t)}{Q_t(x_t)}\right| \cdot \mathbb{E}_{x_t\sim Q_t}\left[\left|V(x_t;\tilde{\pi}_t^\star\circ\hat{\pi}_{t+1:H},R) - V(x_t;\hat{\pi}_{t:H},R)\right|\right] \\
&\leq \sum_{t=1}^{h} \sup_{x_t}\left|\frac{Q_t^\star(x_t)}{Q_t(x_t)}\right| \cdot \Delta_{csc}.
\end{aligned}$$

The first line appends $\hat{\pi}_{h+1:H}$ to the roll-out policy, which as we argued does not affect the value function for any policy. The second line simply introduces the distribution Q_t that we used for learning $\hat{\pi}_t$. The third line is Holder's inequality, and in the fourth line, we use the fact that $\tilde{\pi}_t^\star$ is *pointwise* better than $\hat{\pi}_t$, so we can remove the absolute values. Then we simply use our guarantee from Proposition 6.

We finish the proof by using the policy cover property (Definition 2), namely that

$$\sup_{x_t}\left|\frac{Q_t^\star(x_t)}{Q_t(x_t)}\right| = \sup_{s_t}\left|\frac{\mathbb{P}_{\pi^\star_{1:t-1}}[s_t]}{\frac{1}{|\Psi_t|}\sum_{\pi\in\Psi_t}\mathbb{P}_\pi[s_t]}\right| \leq \sup_{s_t}\frac{\eta(s_t)}{\frac{1}{N}\alpha\eta(s_t)} = \frac{N}{\alpha}.$$

Combining terms proves the theorem. □

D Analysis of the `ExpOracle` Algorithm

In this section we study the `ExpOracle` algorithm, which we assume is given a backward KI abstraction $\phi^\star_{\text{back}} : \mathcal{X} \to [N]$. We establish that with this abstraction, both exploration and policy optimization are relatively straightforward. Formally, we prove the following theorem:

Theorem 7 (Main theorem for `ExpOracle`). *For any Block MDP, given a backward kinematic inseparability abstraction $\phi^\star_{\text{back}} : \mathcal{X} \to [N]$ such that $\phi^\star_{\text{back}} \in \Phi_N$, and parameters $(\epsilon, \eta, \delta) \in (0,1)^3$ with $\eta \leq \eta_{min}$, `ExpOracle` outputs policy covers $\{\Psi_h\}_{h=1}^H$ and a reward sensitive policy $\hat{\pi}$ such that the following guarantees hold, with probability at least $1-\delta$:*

1. *For each $h \in [H]$, Ψ_h is a $1/2$-policy cover for $\mathcal{S}_h$;*
2. $V(\hat{\pi}) \geq \max_{\pi \in \Pi_{NS}} V(\pi) - \epsilon$.

The sample complexity is $\widetilde{\mathcal{O}}\left(NH^2 n_{\text{psdp}} + H n_{\text{eval}}\right) = \widetilde{\mathcal{O}}\left(N^2 H^3 |\mathcal{A}| \log(|\Pi|/\delta)\left(NH/\eta^2_{min} + 1/\epsilon^2\right)\right)$, and the running time is $\widetilde{\mathcal{O}}\left(NH^3 n_{\text{psdp}} + H^2 n_{\text{eval}} + NH^2 \cdot \text{Time}_{\text{pol}}(n_{\text{psdp}}) + H \cdot \text{Time}_{\text{pol}}(n_{\text{eval}})\right)$, where n_{psdp} and n_{eval} are defined in Algorithm 1.

The intuition for the proof is as follows: Consider the time step h. We know from Lemma 7 that for any $i \in [N]$ the policy $\pi \in \Upsilon$ (that is, an unrestricted policy) maximizing the internal reward $R_i(x,a) := \mathbf{1}\{\phi^\star_{\text{back}}(x) = i \wedge \tau(x) = h\}$ simultaneously maximizes $\mathbb{P}_\pi(x)$ for each $x \in \mathcal{X}_h$ such that $\phi^\star_{\text{back}}(x) = i$. In other words, if index i corresponds to a set of observations $\mathcal{X}_h^{(i)} \subseteq \mathcal{X}_h$, then this policy is a homing policy for every observation $x \in \mathcal{X}_h^{(i)}$, *simultaneously*. As this reward function is in our class of internal rewards (which we will verify), we have realizability and completeness by Assumption 2, so Theorem 5 reveals that we learn a policy that approximately maximizes this reward function. Naturally, this policy is an approximate homing policy for all observations $x \in \mathcal{X}_h^{(i)}$. By repeating this argument for all indices $i \in [N]$ and working inductively from $h = 1, \ldots, H$, we build a policy cover for all $\mathcal{X}_h$. This also gives a policy cover for $\mathcal{S}_h$. Once we have this cover, learning the reward sensitive policy is a straightforward application of Theorem 5.

Proof. We first establish the policy cover guarantee, for which we inductively build up $\Psi_1, \ldots, \Psi_H$. As the starting distribution is fixed, we may simply take $\Psi_1 := \emptyset$, which establishes the base case.

Now fix h and let us condition on the event that $\Psi_1, \ldots, \Psi_{h-1}$ are $1/2$-policy covers for $\mathcal{S}_1, \ldots, \mathcal{S}_{h-1}$, respectively. Fix $i \in [N]$. Define the reward function $R_{i,h}(x,a) := \mathbf{1}\{\tau(x) = h \wedge \phi^\star_{\text{back}}(x) = i\}$ and $\mathcal{X}_h^{(i)} := \{x \in \mathcal{X}_h \mid \phi^\star_{\text{back}}(x) = i\}$. We will assume $\mathcal{X}_h^{(i)} \neq \emptyset$ otherwise $R_{i,h}$ is 0 everywhere so we can ignore this case. As $\phi^\star_{\text{back}} \in \Phi_N$, it follows that $R_{i,h}$ is in our class of internal reward functions, so Assumption 2 holds, and we may apply Theorem 5. As such, the call to `PSDP` in this loop of `ExpOracle` finds a policy $\pi_{i,h}$ with the following guarantee: with probability $1 - h\delta$

$$V(\pi_{i,h}; R_{i,h}) \geq \max_{\pi' \in \Pi^h} V(\pi'; R_{i,h}) - 2Nh\Delta_{csc},$$

where at the end of the proof we will instantiate Δ_{csc} with our choice of n and our failure probability. In particular it should be understood that Δ_{csc} is defined in terms of n_{psdp}. It is straightforward to see from the definition of $R_{i,h}$ that $V(\pi'; R_{i,h}) = \mathbb{P}_{\pi'}(\mathcal{X}_h^{(i)})$.

Let $s \in \mathcal{S}_h$ be such that $\mathcal{X}_s \cap \mathcal{X}_h^{(i)} \neq \emptyset$ where recall $\mathcal{X}_s$ is the set of observations emitted from s. (Such a state must exist as $\mathcal{X}_h^{(i)}$ is non-empty.) By transitivity of backward KI, we have that $\mathcal{X}_s \cup \mathcal{X}_h^{(i)}$ is a backward KI set, and so Lemma 6 yields that for any policy π, we have $\mathbb{P}_\pi(\mathcal{X}_h^{(i)})/\mathbb{P}_{\pi_s}(\mathcal{X}_h^{(i)}) = \mathbb{P}_\pi(s)/\mathbb{P}_{\pi_s}(s)$. Recall $\pi_s := \text{argmax}_\pi \mathbb{P}_\pi(s)$ is the homing policy of s and $\eta(s) = \max_\pi \mathbb{P}_\pi(s)$. Using this with our guarantee for `PSDP` we have:

$$\mathbb{P}_{\pi_{i,h}}(s) \geq \eta(s) - 2Nh\Delta_{csc}\frac{\mathbb{P}_{\pi_s}(s)}{\mathbb{P}_{\pi_s}(\mathcal{X}_h^{(i)})}.$$

This holds for any $i \in [N]$ and $s \in \mathcal{S}$ satisfying $\mathcal{X}_s \cap \mathcal{X}_h^{(i)} \neq \emptyset$. The failure probability for any single call to PSDP is $h\delta \leq H\delta$ and so the by union bound the probability of failure for N calls is $NH\delta$.

For any $s \in \mathcal{S}_h$ we define $j(s) := \arg\max_{i \in [N]} \mathbb{P}_{\pi_s}(\mathcal{X}_h^{(i)} \cap \mathcal{X}_s)$. We can then bound $\mathbb{P}_{\pi_s}(\mathcal{X}_h^{(j(s))})$ as:

$$\mathbb{P}_{\pi_s}(\mathcal{X}_h^{(j(s))}) \geq \mathbb{P}_{\pi_s}(\mathcal{X}_h^{(j(s))} \cap \mathcal{X}_s) \geq \frac{1}{N}\sum_{i=1}^{N} \mathbb{P}_{\pi_s}(\mathcal{X}_h^{(i)} \cap \mathcal{X}_s) = \frac{1}{N}\mathbb{P}_{\pi_s}(\cup_{i \in [N]} \mathcal{X}_h^{(i)} \cap \mathcal{X}_s) = \frac{1}{N}\mathbb{P}_{\pi_s}(s),$$

where we use the definition of $j(s)$, the fact that the maximum of a set of values is greater than the mean, and the fact that $\mathcal{X}_h^{(i)}$ form a partition of $\mathcal{X}_h$. Applying this bound, we obtain

$$\forall s \in \mathcal{S}_h,\ \exists j(s) \in [N], \qquad \mathbb{P}_{\pi_{j(s),h}}(s) \geq \eta(s) - 2N^2 h \Delta_{csc},$$

with probability of at least $1 - NH\delta$. It is easy to see that if $2N^2 h\Delta_{csc}$ is at most $\frac{\eta_{min}}{2}$ then $\Psi_h := \{\pi_{i,h}\}$ is a $\frac{1}{2}$-policy cover of $\mathcal{S}_h$. Thus we require

$$n_{\text{psdp}}(\delta) \geq \frac{256N^4H^2|\mathcal{A}|\log(2|\Pi|/\delta)}{\eta_{min}^2}.$$

We apply this argument inductively for every $h \in [H]$. Each step in this induction fails with probability of at most $NH\delta$ so by union bound the failure probability is at most $NH^2\delta$.

The reward sensitive learning is similar, we simply apply Theorem 5 and observe that we have $1/2$-policy covers $\Psi_1, \ldots, \Psi_H$, so the error guarantee is $2NH\Delta_{csc}$ with failure probability at most $H\delta$. To bound this error by ϵ we want to set $n_{\text{eval}}(\delta)$ as:

$$n_{\text{eval}}(\delta) \geq \frac{64N^2H^2|\mathcal{A}|\log(2|\Pi|/\delta)}{\epsilon^2}.$$

The total failure probability is $NH^2\delta + H\delta \leq 2NH^2\delta$. Therefore, we rebind $\delta \mapsto \frac{\delta}{2NH^2}$ and set our hyperparameters to $n_{\text{psdp}}\left(\frac{\delta}{2NH^2}\right)$ and $n_{\text{eval}}\left(\frac{\delta}{2NH^2}\right)$.

We conclude the proof by calculating the overall sample and computational complexity. The sample compleixty is simply $\mathcal{O}\left(n_{\text{psdp}}H^2N + n_{\text{eval}}H\right)$ which evaluates to

$$\widetilde{\mathcal{O}}\left(N^2H^3|\mathcal{A}|\log(|\Pi|/\delta)\left(\frac{N^3H}{\eta_{min}^2} + \frac{1}{\epsilon^2}\right)\right).$$

The running time is dominated by collecting the trajectories and invoking the contextual bandit oracle. For the former, we collect $\mathcal{O}\left(NH^2n_{\text{psdp}} + Hn_{\text{eval}}\right)$ trajectories in total, but each trajectory takes $O(H)$ time. For the latter, we make NH^2 calls to the oracle with n_{psdp} samples and H calls to the oracle with n_{eval} samples. □

Theorem 7 assumes access to a backward KI abstraction in our decoding class Φ_N. But does Φ_N contain a backward KI abstraction? The next proposition shows that our realizability assumption guarantees that we can construct a backward KI abstraction function using members of Φ_N for $N \geq \mathrm{N}_{\mathrm{KD}}$.

Proposition 8. *For any $h \in [H]$ and $N \geq N_{KD}$, there exists a $\phi_h^{(\mathrm{B})} \in \Phi_N$ such that for any $x'_1, x'_2 \in \mathcal{X}_h$ if $\phi_h^{(\mathrm{B})}(x'_1) = \phi_h^{(\mathrm{B})}(x'_2)$ then x'_1 and x'_2 are backward kinematically inseparable.*

Proof. Fix $N \geq \mathrm{N_{KD}}$ and $h \in [H]$. Let $\rho \in \Delta(\mathcal{S}_h)$ with $\mathrm{supp}(\rho) = \mathcal{S}_h$. From Assumption 2, there exists $f_\rho \in \mathcal{F}_N$ such that for any $x \in \mathcal{X}_{h-1}, a \in \mathcal{A}, x' \in \mathcal{X}_h$:

$$f_\rho(x, a, x') = \frac{T(g^\star(x') \mid g^\star(x), a)}{T(g^\star(x') \mid g^\star(x), a) + \rho(g^\star(x'))} = \frac{T(x' \mid x, a)}{T(x'|x, a) + \rho(x')}.$$

From our construction of the function class $\mathcal{F}_N$, we can decompose f_ρ into two state abstractions $\phi^{(\mathrm{F})}_{h-1}, \phi^{(\mathrm{B})}_h$, and a lookup table p such that $f_\rho(x, a, x') = p(\phi^{(\mathrm{F})}_{h-1}(x), a, \phi^{(\mathrm{B})}_h(x'))$. Then, for any $x'_1, x'_2 \in \mathcal{X}_h$ with $\phi^{(\mathrm{B})}_h(x'_1) = \phi^{(\mathrm{B})}_h(x'_2)$, we have $f_\rho(x, a, x'_1) = f_\rho(x, a, x'_2)$, which implies

$$\frac{T(x'_1 \mid x, a)}{T(x'_1 \mid x, a) + \rho(x'_1)} = \frac{T(x'_2 \mid x, a)}{T(x'_2 \mid x, a) + \rho(x'_2)} \Rightarrow \frac{T(x'_1 \mid x, a)}{\rho(x'_1)} = \frac{T(x'_2 \mid x, a)}{\rho(x'_2)}.$$

If $x \notin \mathcal{X}_{h-1}$ then $\frac{T(x'_1|x,a)}{\rho(x'_1)} = \frac{T(x'_2|x,a)}{\rho(x'_2)} = 0$ trivially due to layering. Therefore, Lemma 5 then reveals that x'_1, x'_2 are backward KI, which proves the claim. □

Proposition 8 enables us to define an abstraction $\phi^{(\mathrm{B})} : \mathcal{X} \to [N]$ such that $\phi^{(\mathrm{B})}(x) = \phi^{(\mathrm{B})}_{\tau(x)}(x)$ for any $x \in \mathcal{X}$ and where $\{\phi^{(\mathrm{B})}_1, \ldots, \phi^{(\mathrm{B})}_H\}$ are defined as in Proposition 8 for a fixed $N \geq \mathrm{N_{KD}}$. Theorem 7 then holds using $\phi^{(\mathrm{B})}$ as input, even though it may not be in Φ_N, as long as each element $\phi^{(\mathrm{B})}_h \in \Phi_N$. This can be verified by noting that the proof of Theorem 7 uses only two properties of the input state abstraction: (i) it maps backward maps backward KI observations from *the same time step* together and (ii) we have policy completeness with respect to the internal reward functions defined by the abstraction functions. The former is already accomplished by our construction of $\phi^{(\mathrm{B})}$. The latter can be verified by observing that any internal reward function R_{ih} defined using $\phi^{(\mathrm{B})}$ can be equivalently expressed as $R_{ih}(x, a) = \mathbf{1}\{\tau(x) = h \wedge \phi^{(\mathrm{B})}_h(x) = i\}$ for all $x \in \mathcal{X}, a \in \mathcal{A}$. Since $\phi^{(\mathrm{B})}_h \in \Phi_N$, we have policy completeness for this reward function.

E Analysis for the HOMER algorithm

In this section we present the analysis for HOMER. The proof is inductive in nature, proceeding from time point $h = 1$ to $h = H$, where for the h^{th} step, we use the policy covers from time points $h' = 1, \ldots, h-1$ to construct the policy cover at time h. Formally, the inductive claim is that for each h, given α-policy covers $\Psi_1, \ldots, \Psi_{h-1}$ over $\mathcal{S}_1, \ldots, \mathcal{S}_{h-1}$, the h^{th} iteration of HOMER finds an α-policy cover Ψ_h for $\mathcal{S}_h$. We will verify the base case shortly, and we break down the inductive argument into two main components: In the first part, we analyze the contrastive estimation problem and introduce a coupling to show that the supervised learning problem relates to backward kinematic inseparability. In this second part, we use this coupling to show that invoking PSDP with the learned representation yields a policy cover for time point h.

The base case. The base case is that Ψ_1 found by the algorithm is a policy cover over $\mathcal{S}_1$. This is easy to see, since for any states $s \in \mathcal{S}_1$, we have $\eta(s) = \mu(s)$, where recall that μ is the starting stat distribution. We can define Ψ_1 to be any finite set of policies, which immediately is a 1-policy cover, but since we never actually execute these policies, we simply set $\Psi_1 = \emptyset$ in Algorithm 3 (line 1).

E.1 The supervised learning problem and a coupling

In this subsection we analyze the supervised learning problem induced by HOMER, which is a form of contrastive estimation (line 11 in Algorithm 3). We reason about the Bayes optimal predictor for this problem, obtain a finite-sample generalization bound, and introduce a coupling to elucidate the connection to backward kinematic inseparability. Fix a time point $h \in \{2, \dots, H\}$ and inductively assume that we have α policy covers $\Psi_1, \dots, \Psi_{h-1}$ over $\mathcal{S}_1, \dots, \mathcal{S}_{h-1}$ respectively. For the rest of this subsection, we suppress dependence on h in observations, that is we will always take $x \in \mathcal{X}_{h-1}$, and $x' \in \mathcal{X}_h$.

The supervised learning problem at time h is induced by a distribution $D \in \Delta(\mathcal{X}_{h-1}, \mathcal{A}, \mathcal{X}_h, \{0,1\})$, which is defined as follows: Two tuples are obtained $(x_1, a_1, x_1'), (x_2, a_2, x_2')$ are obtained by sampling $\pi_1, \pi_2 \sim \texttt{Unf}(\Psi_{h-1})$, and executing the policies $\pi_1 \circ \texttt{Unf}(\mathcal{A})$ and $\pi_2 \circ \texttt{Unf}(\mathcal{A})$, respectively. Then with probability $1/2$ the sample from D is $(x_1, a_1, x_1, 1)$ and with the remaining probability, the sample is $(x_1, a_1, x_2', 0)$. Let $D(x, a, x' \mid y)$ be the conditional probability of the triple, conditional on the label y. Let $\rho_h \in \Delta(\mathcal{X}_h)$ denote the marginal probability distribution over x', that is $\rho_h(x') := \mathbb{P}_{\pi \circ \texttt{Unf}:\pi \sim \texttt{Unf}(\Psi_{h-1})}[x']$, and let $\mu_{h-1} \in \Delta(\mathcal{X}_{h-1})$ be the marginal distribution over x, that is $\mu_{h-1}(x) := \mathbb{P}_{\pi \sim \texttt{Unf}(\Psi_{h-1})}[x]$. These definitions are lifted to the state spaces $\mathcal{S}_{h-1}$ and $\mathcal{S}_h$ in the natural way.

With these definition, we have

$$D(x, a, x' \mid y = 1) = \frac{\mu_{h-1}(x)}{|\mathcal{A}|} \cdot T(x' \mid x, a), \qquad D(x, a, x' \mid y = 0) = \frac{\mu_{h-1}(x)}{|\mathcal{A}|} \cdot \rho_h(x').$$

The first lemma uses the fact that Ψ_{h-1} is an α-policy cover to lower bound the marginal probability $\rho_h(s_h)$, which ensure we have adequate coverage in our supervised learning problem.

Lemma 8. *If Ψ_{h-1} is an α-policy cover over $\mathcal{S}_{h-1}$, then for any $s \in \mathcal{S}_h$, we have $\rho_h(s) \geq \frac{\alpha \eta(s)}{N|\mathcal{A}|}$.*

Proof. For any $s \in \mathcal{S}_h$, we first upper bound $\eta(s)$ by

$$\begin{aligned}
\eta(s) &= \sup_{\pi \in \Pi_{\mathrm{NS}}} \mathbb{P}_\pi(s) = \sup_{\pi \in \Pi_{\mathrm{NS}}} \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \mathbb{P}_\pi[s_{h-1}] \mathbb{E}_{x \sim q(\cdot \mid s_{h-1})} \left[\sum_{a \in \mathcal{A}} \pi(a \mid x) T(s \mid s_{h-1}, a) \right] \\
&\leq \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \sup_{\pi \in \Pi_{\mathrm{NS}}} \mathbb{P}_\pi[s_{h-1}] \sum_{a \in \mathcal{A}} T(s \mid s_{h-1}, a) = \sum_{s_{h-1} \in \mathcal{S}_{h-1}} \eta(s_{h-1}) \sum_{a \in \mathcal{A}} T(s \mid s_{h-1}, a).
\end{aligned}$$

We can also lower bound ρ_h as

$$\rho_h(s) = \sum_{\substack{s_{h-1} \in \mathcal{S}_{h-1} \\ a \in \mathcal{A}}} \frac{\mathbb{P}_{\pi \sim \texttt{Unf}(\Psi_{h-1})}[s_{h-1}]}{|\mathcal{A}|} T(s \mid s_{h-1}, a) \geq \frac{\alpha}{N|\mathcal{A}|} \sum_{\substack{s_{h-1} \in \mathcal{S}_{h-1} \\ a \in \mathcal{A}}} \eta(s_{h-1}) T(s \mid s_{h-1}, a) \geq \frac{\alpha \eta(s)}{N|\mathcal{A}|}.$$

Here the first identity expands the definition, and in the first inequality we use the fact that Ψ_{h-1} is an α-policy cover. The last inequality uses our upper bound on $\eta(s)$. □

The next lemma characterizes the Bayes optimal predictor for square loss minimization with respect to D. Recall that the Bayes optimal classifier is defined as

$$f^\star := \operatorname*{argmin}_f \mathbb{E}_{(x,a,x',y) \sim D} \left[\left(f(x, a, x') - y \right)^2 \right]$$

where the minimization is over *all* measurable functions.

Lemma 9. *The Bayes optimal predictor for square loss minimization over D is*

$$f^\star(x,a,x') := \frac{T(g^\star(x') \mid g^\star(x), a)}{T(g^\star(x') \mid g^\star(x), a) + \rho_h(g^\star(x'))}$$

Under Assumption 2, we have that $f^\star \in \mathcal{F}_N$ for any $N \geq N_{KD}$.

Proof. As we are using the square loss, the Bayes optimal predictor is the conditional mean, so $f^\star(x,a,x') = \mathbb{E}_D[y \mid (x,a,x')] = D(y=1 \mid x,a,x')$. By Bayes rule and the fact that the marginal probability for both labels is $1/2$, we have

$$\begin{aligned} D(y=1 \mid x,a,x') &= \frac{D(x,a,x' \mid y=1)}{D(x,a,x' \mid y=1) + D(x,a,x' \mid y=0)} = \frac{T(x' \mid x,a)}{T(x' \mid x,a) + \rho_h(x')} \\ &= \frac{T(g^\star(x') \mid g^\star(x), a)}{T(g^\star(x') \mid g^\star(x), a) + \rho_h(g^\star(x'))}. \end{aligned}$$

□

Now that we have characterized the Bayes optimal predictor, we turn to the learning rule. We perform empirical risk minimization over n iid samples from D to learn a predictor $\hat{f} \in \mathcal{F}_N$ (We will bind $n = n_{\text{reg}}$ toward the end of the proof). As $\mathcal{F}_N$ has pointwise metric entropy growth rate $\ln \mathcal{N}(\mathcal{F}_N, \varepsilon) \leq c_0 d_N \ln(1/\varepsilon)$, a standard square loss generalization analysis (see Proposition 12) yields the following corollary, which follows easily from Proposition 12.

Corollary 9. *For any $\delta \in (0,1)$ with probability at least $1-\delta$, the empirical risk minimizer, $\hat{f}$ based n iid samples from D satisfies*[9]

$$\mathbb{E}_D\left[\left(\hat{f}(x,a,x') - f^\star(x,a,x')\right)^2\right] \leq \Delta_{reg} \text{ with } \Delta_{reg} := \frac{16\left(\ln|\Phi_N| + N^2|\mathcal{A}|\ln(n) + \ln(2/\delta)\right)}{n}.$$

Proof. The proof follows from a bound on the pointwise covering number of the class $\mathcal{F}_N$. For any $\varepsilon > 0$ we first form a cover of the class $\mathcal{W}_N$ by discretizing the output space to $Z := \{\epsilon, \ldots, \lfloor 1/\epsilon \rfloor \epsilon\}$, and letting W_N be all functions from $[N] \times \mathcal{A} \times [N] \to Z$. Clearly we have $|W_N| \leq (1/\epsilon)^{N^2|\mathcal{A}|}$, and it is easy to see that W_N is a pointwise cover for $\mathcal{W}_N$. Then we form $F_N = \{(x,a,x') \mapsto w(\phi(x), a, \phi'(x')) : w \in W_N, \phi, \phi' \in \Phi_N\}$, which is clearly a pointwise cover for $\mathcal{F}_N$ and has size $|\Phi_N|^2|W_N|$. In other words, the pointwise log-covering number is $N^2|\mathcal{A}|\ln(1/\varepsilon) + 2\ln|\Phi_N|$, which we plug into the bound in Proposition 12. Taking $\varepsilon = 1/n$ there the bound from Proposition 12 is at most

$$\frac{6}{n} + \frac{16\ln|\Phi_N| + 8N^2|\mathcal{A}|\ln(n) + 8\ln(2/\delta)}{n} \leq \Delta_{reg}.$$

□

The coupling. For the next step of the proof, we introduce a useful coupling distribution based on D. Let $D_{\text{coup}} \in \Delta(\mathcal{X}_{h-1} \times \mathcal{A} \times \mathcal{X}_h \times \mathcal{X}_h)$ have density $D_{\text{coup}}(x,a,x'_1,x'_2) = D(x,a)\rho_h(x'_1)\rho_h(x'_2)$. That is, we sample x, a by choosing $\pi \sim \texttt{Unf}(\Psi_{h-1})$, rolling in, and then taking a uniform action $a_{h-1} \sim \texttt{Unf}(\mathcal{A})$. Then, we obtain x'_1, x'_2 independently by sampling from the marginal distribution ρ_h induced by D.

It is also helpful to define the shorthand notation $V : \mathcal{X}_h \times \mathcal{X}_h \times \mathcal{X}_{h-1} \times \mathcal{A} \to \mathbb{R}$ by

$$V(x'_1, x'_2, x, a) := \frac{T(g^\star(x'_1) \mid g^\star(x), a)}{\rho_h(g^\star(x'_1))} - \frac{T(g^\star(x'_2) \mid g^\star(x), a)}{\rho_h(g^\star(x'_2))}.$$

[9] As we remark in Appendix F, sharper generalization analyses are possible, with more refined notions of statistical complexity. Such results are entirely composable with our analyses.

This function is lifted to operate on states $\mathcal{S}_h$ in the natural way. Note also that, as $\rho_h(\cdot) > 0$ everywhere, V is well-defined. Observe that V is related to the notion of backward kinematic inseparability. Finally, define

$$b_i := \mathbb{E}_{x' \sim \rho_h}\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x') = i\}\right],$$

which is the prior probability over the learned abstract state i, where $\hat{\phi}_h^{(\mathrm{B})}$ is the learned abstraction function for time h implicit in the predictor $\hat{f}$. In the next lemma, we show that $\hat{\phi}_h^{(\mathrm{B})}$ approximately learns a backward KI abstraction by relating the excess risk of $\hat{f}$ to the performance of the decoder via the V function.

Lemma 10. *Let $\hat{f} =: (\hat{w}, \hat{\phi}_{h-1}^{(\mathrm{F})}, \hat{\phi}_h^{(\mathrm{B})})$ be the empirical risk minimizer on n iid samples from D, that is the output of $\texttt{REG}(\mathcal{F}_N, \mathcal{D})$. Under the $1-\delta$ event of Corollary 9, for each $i \in [N]$ we have*

$$\mathbb{E}_{D_{\text{coup}}}\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i = \hat{\phi}_h^{(\mathrm{B})}(x_2')\}\,|V(x_1', x_2', x, a)|\right] \le 8\sqrt{b_i \Delta_{reg}}.$$

Proof. For the proof, it is helpful to introduce a second coupled distribution D'_{coup} in which x, a are sampled as before, but now $x_1', x_2' \overset{iid}{\sim} D(\cdot \mid x, a)$, instead of from the prior. Note that this condition probability is $D(x' \mid x, a) = 1/2 q(x' \mid g^\star(x))\,(T(g^\star(x') \mid g^\star(x), a) + \rho_h(g^\star(x')))$. To translate from D_{coup} to D'_{coup} we expand the definition of V and introduce $f^\star$. The main observation here is that V is normalized by $\rho_h(\cdot)$ but $f^\star$ is normalized, essentially by $D(\cdot \mid x, a)$.

$$\begin{aligned} V(x_1', x_2', x, a) &= \frac{\rho_h(x_2')T(x_1' \mid x, a) - \rho_h(x_1')T(x_2' \mid x, a)}{\rho_h(x_1')\rho_h(x_2')} \\ &= \frac{4D(x_1' \mid x, a)D(x_2' \mid x, a)}{\rho_h(x_1')\rho_h(x_2')} \cdot \left(f^\star(x, a, x_1') - f^\star(x, a, x_2')\right). \end{aligned}$$

The last step follows since, in the first term the emission distributions cancel, while the cross terms cancel when we introduce the least common multiple in the term $f^\star(x, a, x_1') - f^\star(x, a, x_2')$. Specifically

$$\begin{aligned} f^\star(x, a, x_1') - f^\star(x, a, x_2') &= \frac{T(g^\star(x_1') \mid g^\star(x), a)}{T(g^\star(x_1') \mid g^\star(x), a) + \rho_h(g^\star(x_1'))} - \frac{T(g^\star(x_2') \mid g^\star(x), a)}{T(g^\star(x_2') \mid g^\star(x), a) + \rho_h(g^\star(x_2')} \\ &= \frac{\rho_h(g^\star(x_2'))T(g^\star(x_1') \mid g^\star(x), a) - \rho_h(g^\star(x_1'))T(g^\star(x_2') \mid g^\star(x), a)}{(T(g^\star(x_1') \mid g^\star(x), a) + \rho_h(g^\star(x_1')))\,T(g^\star(x_2') \mid g^\star(x), a) + \rho_h(g^\star(x_2'))}. \end{aligned}$$

Nevertheless, this calculation lets us translate from D_{coup} to D'_{coup} while moving from V to $f^\star$. For shorthand, let $\mathcal{E}_i := \mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i = \hat{\phi}_h^{(\mathrm{B})}(x_2')\}$, so the above derivation yields

$$\mathbb{E}_{D_{\text{coup}}}\left[\mathcal{E}_i \cdot |V(x_1', x_2', x, a)|\right] = 4\mathbb{E}_{D'_{\text{coup}}}\left[\mathcal{E}_i \cdot |f^\star(x, a, x_1') - f^\star(x, a, x_2')|\right]. \tag{2}$$

Now that we have introduce $f^\star$, we can introduce $\hat{f}$ and relate to the excess risk

$$\begin{aligned} &\mathbb{E}_{D'_{\text{coup}}}\left[\mathcal{E}_i \cdot |f^\star(x, a, x_1') - f^\star(x, a, x_2')|\right] \\ &\quad\le \mathbb{E}_{D'_{\text{coup}}}\left[\mathcal{E}_i \cdot \left(\left|f^\star(x, a, x_1') - \hat{f}(x, a, x_1')\right| + \left|f^\star(x, a, x_2') - \hat{f}(x, a, x_1')\right|\right)\right] \\ &\quad= \mathbb{E}_{D'_{\text{coup}}}\left[\mathcal{E}_i \cdot \left(\left|f^\star(x, a, x_1') - \hat{f}(x, a, x_1')\right| + \left|f^\star(x, a, x_2') - \hat{f}(x, a, x_2')\right|\right)\right] \\ &\quad\le 2\mathbb{E}_D\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x') = i\}\left|f^\star(x, a, x') - \hat{f}(x, a, x')\right|\right] \le 2\sqrt{b_i \mathbb{E}_D\left[\left(f^\star(x, a, x') - \hat{f}(x, a, x')\right)^2\right]} \\ &\quad\le 2\sqrt{b_i \Delta_{reg}}. \end{aligned}$$

The first step is the triangle inequality. The key step is the second one: the identity uses the fact that under event $\mathcal{E}_i$, we know that $\hat{\phi}_h^{(\mathrm{B})}(x_1') = \hat{\phi}_h^{(\mathrm{B})}(x_2')$, which, by the bottleneck structure of $\hat{f}$, yields that $\hat{f}(x, a, x_1') = \hat{f}(x, a, x_2')$. In the third step we combine terms, drop the dependence on the other observaiton, and use the fact that D'_{coup} shares marginal distributions with D. Finally, we use the Cauchy-Schwarz inequality, the fact that $\mathbb{E}_D \mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x') = i\} = b_i$, and Corollary 9. Combining with (2) proves the lemma. □

An aside on realizability. Given the bottleneck structure of our function class $\mathcal{F}_N$, it is important to ask whether realizability is even feasible. In particular for a bottleneck capacity of N, each $f \in \mathcal{F}_N$ has a range of at most $N^2|\mathcal{A}|$ discrete values. If we choose N to be too small, we may not have enough degrees of freedom to express $f^\star$. By inspection $f^\star$ has a range of at most $|\mathcal{S}|^2|\mathcal{A}|$, so certainly a bottleneck capacity of $N \geq |\mathcal{S}|$ suffices. In the next proposition, we show that in fact $N \geq \mathrm{N}_{\mathrm{KD}}$ suffices, which motivates the condition in Assumption 2.

Proposition 10. *Fix $h \in \{2, \ldots, H\}$. If $x_1, x_1 \in \mathcal{X}_{h-1}$ are kinematically inseparable observations, then for all $x' \in \mathcal{X}_h$ and $a \in \mathcal{A}$, we have $f^\star(x_1, a, x') = f^\star(x_2, a, x')$. Analogously, if $x_1', x_2' \in \mathcal{X}_h$ are kinematically inseparable, then for all $x \in \mathcal{X}_{h-1}$ and $a \in \mathcal{A}$, we have $f^\star(x, a, x_1') = f^\star(x, a, x_2')$.*

Proof. We prove the forward direction first. As x_1, x_2 are forward KI, we have

$$f^\star(x_1, a, \tilde{x}) = \frac{T(\tilde{x} \mid x_1, a)}{T(\tilde{x} \mid x_1, a) + \rho_h(\tilde{x})} = \frac{T(\tilde{x} \mid x_2, a)}{T(\tilde{x} \mid x_2, a) + \rho_h(\tilde{x})} = f^\star(x_2, a, \tilde{x}).$$

For the backward direction, as x_1', x_2' are backward KI, from Lemma 5 there exists $u \in \Delta(\mathcal{X})$ with $\mathrm{supp}(u) = \mathcal{X}$ such that $\frac{T(x_1' \mid x, a)}{u(x_1')} = \frac{T(x_2' \mid x, a)}{u(x_2')}$. Further, ρ_h satisfies $\rho_h(x_1') = \frac{u(x_1')}{u(x_2')}\rho_h(x_2')$. Thus, we obtain

$$f^\star(x, a, x_1') = \frac{T(x_1' \mid x, a)}{T(x_1' \mid x, a) + \rho_h(x_1')} = \frac{\frac{u(x_1')}{u(x_2')}T(x_2' \mid x, a)}{\frac{u(x_1')}{u(x_2')}T(x_2' \mid x, a) + \frac{u(x_1')}{u(x_2')}\rho_h(x_2')} = f^\star(x, a, x_2'). \qquad \square$$

E.2 Building the policy cover

Lemma 10 relates our learned decoder function $\hat{\phi}_h^{(\mathrm{B})}$ to backward KI. For some intuition as to why, it is helpful to consider the asymptotic regime, where $n \to \infty$, so that $\Delta_{reg} \to 0$. In this regime, Lemma 10 shows that whenever $\hat{\phi}_h^{(\mathrm{B})}$ maps two observations to the same abstract state, these observations must have $V = 0$ for all x, a. As our distribution has full support, by Lemma 5, these observations must be backward KI. Of course, this argument only applies to the asymptotic regime. In this section, we establish a finite-sample analog, and we show how using the internal reward functions induced by $\hat{\phi}_h^{(\mathrm{B})}$ in PSDP yields a policy cover for $\mathcal{S}_h$.

The first lemma is a comparison lemma, which lets us compare visitation probabilities for two policies.

Lemma 11. *Assume Ψ_{h-1} is an α policy cover for $\mathcal{S}_{h-1}$. Then for any two policies π_1, π_2 and any state $s \in \mathcal{S}_h$, we have*

$$\mathbb{P}[s \mid \pi_1] - \mathbb{P}[s \mid \pi_2] \leq \min_{i \in [N]} \frac{\rho_h(s)}{b_i} \left(\mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i \mid \pi_1] - \mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i \mid \pi_2] \right) + \frac{16N|\mathcal{A}|\rho_h(s)}{\alpha b_i^{3/2}} \sqrt{\Delta_{reg}}.$$

Proof. The key step is to observe that by the definition of V

$$\forall x_2', x, a : \mathbb{E}_{x_1' \sim \rho_h}\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}V(x_1', x_2', x, a)\right] = \sum_{x_1'} \mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}T(x_1' \mid x, a) - \frac{b_i T(x_2' \mid x, a)}{\rho_h(x_2')}.$$

Using this identity, we may express the visitation probability for a policy as

$$\begin{aligned}
&\mathbb{P}[s \mid \pi_1] = \mathbb{E}_{(x,a)\sim\pi_1} \sum_{x_2'} \mathbf{1}\{g^\star(x_2') = s\}T(x_2' \mid x, a) \\
&= \frac{1}{b_i}\mathbb{E}_{(x,a)\sim\pi_1, x_2'\sim\rho_h} \sum_{x_1'} \mathbf{1}\{g^\star(x_2') = s \wedge \hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}\left(T(x_1' \mid x, a) - \rho_h(x_1')V(x_1', x_2', x, a)\right) \\
&= \frac{\rho_h(s)}{b_i}\mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i \mid \pi] - \frac{1}{b_i}\mathbb{E}_{D_{\text{coup}}}\left[\frac{\pi(x,a)}{D(x,a)}\mathbf{1}\{g^\star(x_2') = s \wedge \hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}V(x_1', x_2', x, a)\right].
\end{aligned}$$

Here we are using the shorthand $\pi(x,a) = \mathbb{P}[x \mid \pi]\pi(a \mid x)$ for the policy occupancy measure, with a similar notation the distribution D induced by our policy cover Ψ_{h-1}. Using the inductive hypothesis that Ψ_{h-1} is a α-policy cover (essentially Lemma 8), we have

$$\left|\frac{\pi(x,a)}{D(x,a)}\right| = \left|\frac{\mathbb{P}[x \mid \pi]\cdot\pi(a \mid x)}{\mathbb{E}_{\pi'\sim\mathtt{Unf}(\Psi_{h-1})}\mathbb{P}[x \mid \pi']\cdot 1/|\mathcal{A}|}\right| \leq |\mathcal{A}|\left|\frac{\mathbb{P}[g^\star(x) \mid \pi]}{\mathbb{E}_{\pi'\sim\mathtt{Unf}(\Psi_{h-1})}\mathbb{P}[g^\star(x) \mid \pi']}\right| \leq \frac{|\mathcal{A}|N}{\alpha}.$$

Combining, the second term above is at most

$$\frac{N|\mathcal{A}|}{\alpha b_i}\mathbb{E}_{D_{\text{coup}}}\left[\mathbf{1}\{g^\star(x_2') = s \wedge \hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}\left|V(x_1', x_2', x, a)\right|\right].$$

Let us now work with just the expectation. Recall that we can lift the definition of V to operate on states $s \in \mathcal{S}_h$ in lieu of observations. Using this fact, we have that under the probability $1-\delta$ event of Lemma 10

$$\begin{aligned}
&\mathbb{E}_{D_{\text{coup}}}\left[\mathbf{1}\{g^\star(x_2') = s \wedge \hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}\left|V(x_1', x_2', x, a)\right|\right] = \rho_h(s)\mathbb{E}_D\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}\left|V(x_1', s, x, a)\right|\right] \\
&= \rho_h(s)\mathbb{E}_D\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i\}\left|V(x_1', s, x, a)\right|\frac{\mathbb{E}_{x_2'\sim\rho_h}\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_2') = i\}}{b_i}\right] \\
&= \frac{\rho_h(s)}{b_i}\mathbb{E}_{D_{\text{coup}}}\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i = \hat{\phi}_h^{(\mathrm{B})}(x_2')\}\left|V(x_1', x_2', x, a)\right|\right] \\
&\leq \frac{\rho_h(s)}{b_i}\mathbb{E}_{D_{\text{coup}}}\left[\mathbf{1}\{\hat{\phi}_h^{(\mathrm{B})}(x_1') = i = \hat{\phi}_h^{(\mathrm{B})}(x_2')\}\left|V(x_1', x_2', x, a)\right|\right] \leq \rho_h(s)\cdot 8\sqrt{\Delta_{reg}/b_i},
\end{aligned}$$

Putting things together, and using the same bound for the second policy, we have the following comparison inequality, which holds in the $1-\delta$ event of Corollary 9

$$\mathbb{P}[s \mid \pi_1] - \mathbb{P}[s \mid \pi_2] \leq \frac{\rho_h(s)}{b_i}\left(\mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i \mid \pi_1] - \mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i \mid \pi_2]\right) + \frac{16N|\mathcal{A}|\rho_h(s)}{\alpha b_i^{3/2}}\sqrt{\Delta_{reg}}.$$

As this calculation applies for each i, we obtain the result. □

In the next lemma, we introduce our policy cover.

Lemma 12. *Assume that $\Psi_1, \ldots, \Psi_{h-1}$ are α policy covers for $\mathcal{S}_1, \ldots, \mathcal{S}_{h-1}$, each of size at most N. Let $\Psi_h := \{\hat{\pi}_{i,h} : i \in [N]\}$ be the policy cover learned at step h of* HOMER*. Then in the $\geq 1 - (1 + NH)\delta$ probability event that the N calls to* PSDP *succeed and Corollary 9 holds, we have that for any state $s \in \mathcal{S}_h$, there exists an index $i \in [N]$ such that*

$$\mathbb{P}[s \mid \hat{\pi}_{i,h}] \geq \eta(s) - \frac{N^2 h \Delta_{csc}}{\alpha} - \frac{16N^3|\mathcal{A}|^{3/2}}{\alpha^{3/2}}\sqrt{\Delta_{reg}/\eta(s)}$$

Proof. Let us condition on the success of Corollary 9, as well as the success of the N calls to PSDP. As the former fails with probability at most δ, and each call to PSDP fails with probability at most $H\delta$, the total failure probability here is $(1 + NH)\delta$.

In this event, by Theorem 5, and the definition of the internal reward function R_i, we know that

$$\mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i | \hat{\pi}_{i,h}] \geq \max_{\pi \in \Pi_{\mathrm{NS}}} \mathbb{P}[\hat{\phi}_h^{(\mathrm{B})}(x') = i | \pi] - \frac{Nh\Delta_{csc}}{\alpha}.$$

Plugging this bound into Lemma 11, we get

$$\mathbb{P}[s \mid \pi] \leq \mathbb{P}[s \mid \hat{\pi}_{i,h}] + \frac{Nh\rho_h(s)\Delta_{csc}}{\alpha b_i} + \frac{16N|\mathcal{A}|\rho_h(s)}{\alpha b_i^{3/2}}\sqrt{\Delta_{reg}}.$$

This bound also holds for all $i \in [N]$. To optimize the bound, we should choose the index i that is maximally correlated with the state s. To do so, define $P_{s,i} := \mathbb{E}_{x' \sim \rho_h} \mathbf{1}\{g^\star(x') = s, \hat{\phi}_h^{(\mathrm{B})}(x') = i\}$, and let us choose $i(s) = \max_i P_{s,i}$. This index satisfies

$$b_{i(s)} = \sum_{s'} P_{s',i(s)} \geq P_{s,i(s)} = \max_i P_{s,i} \geq \frac{1}{N}\sum_{i=1}^{N} P_{s,i} = \frac{\rho_h(s)}{N}$$

Plugging in this bound, we see that for every s, there exists $i \in [N]$ such that

$$\eta(s) = \max_{\pi \in \Pi_{\mathrm{NS}}} \mathbb{P}[s \mid \pi] \leq \mathbb{P}[s \mid \hat{\pi}_{i,h}] + \frac{N^2 h \Delta_{csc}}{\alpha} + \frac{16N^{5/2}|\mathcal{A}|}{\alpha}\sqrt{\Delta_{reg}/\rho_h(s)}$$

We conclude the proof by introducing the lower bound on $\rho_h(s) \geq \frac{\alpha\eta(s)}{N|\mathcal{A}|}$ from Lemma 8 and re-arranging. □

E.3 Wrapping up the proof

Lemma 12 is the core technical result, which certifies that our learned policy cover at time h yields good coverage. We are basically done with the proof; all that remains is to complete the induction, set all of the parameters, and take a union bound.

Union bound. For each $h \in [H]$ we must invoke Corollary 9 once, and we invoke Theorem 5 N times. We also invoke Theorem 5 once more to learn the reward sensitive policy. Thus the total failure probability is $H(\delta_1 + NH\delta_2) + H\delta_3$ where δ_1 appears in Δ_{reg}, δ_2 appears in Δ_{csc} for the internal reward functions, and δ_3 appears in Δ_{csc} for the external reward functions. We therefore take $\delta_1 = \delta/(3H)$ and $\delta_2 = \frac{\delta}{3NH^2}$ and $\delta_3 = \frac{\delta}{3H}$, which gives us the settings

$$\Delta_{csc} = 4\sqrt{\frac{|\mathcal{A}|}{n_{\mathrm{psdp}}}\ln\left(\frac{4NH^2|\Pi|}{\delta}\right)}, \qquad \Delta_{reg} = \frac{16\left(\ln(|\Phi_N|) + N^2|\mathcal{A}|\ln(n_{\mathrm{reg}}) + \ln(6H/\delta)\right)}{n_{\mathrm{reg}}},$$

for the inductive steps. With these choices, the total failure probability for the algorithm is δ.

The policy covers. Fix $h \in [H]$ and inductively assume that $\Psi_1, \ldots, \Psi_{h-1}$ are $1/2$-policy covers for $\mathcal{S}_1, \ldots, \mathcal{S}_{h-1}$. Then by Lemma 12, for each $s \in \mathcal{S}_h$ there exists $i \in [N]$ such that

$$\mathbb{P}[s \mid \hat{\pi}_{i,h}] \geq \eta(s) - 2N^2 H \Delta_{csc} - 32\sqrt{2}N^3|\mathcal{A}|^{3/2}\sqrt{\Delta_{reg}/\eta(s)}.$$

We simply must set n_{psdp} and n_{reg} so that the right hand side here is at least $\eta(s)/2$. By inspection, sufficient conditions for both parameters are:

$$n_{\text{psdp}} \geq \frac{32^2 N^4 H^2 |\mathcal{A}|}{\eta_{min}^2} \ln\left(\frac{4NH^2|\Pi|}{\delta}\right), \qquad 2\underbrace{n_{\text{reg}}}_{:=v} \geq \underbrace{\frac{512^2 N^6 |\mathcal{A}|^3}{\eta_{min}^3}}_{=:a}\left(\underbrace{N^2|\mathcal{A}|}_{=:c}\ln(n_{\text{reg}}) + \underbrace{\ln|\Phi_N| + \ln(6H/\delta)}_{=:b}\right).$$

To simplify the condition for n_{reg} we use the following transcendental inequality: For any $a > e$ and any b if $v \geq a\max\{c\ln(ac) + b, 0\}$ then $2v \geq ac\ln(v) + ab$. To see why, observe that

$$ac\ln(v) + ab = ac\ln(v/(ac)) + ac\ln(ac) + ab \leq v - ac + ac\ln(ac) + ab \leq 2v,$$

where the first inequality is simply that $\ln(x) \leq x - 1$ for $x > 0$, and the second inequality uses the lower bound on v. Using the highlighted definitions, a sufficient condition for n_{reg} is

$$n_{\text{reg}} \geq \frac{512^2 N^6 |\mathcal{A}|^3}{\eta_{min}^3}\left(N^2|\mathcal{A}|\ln\left(\frac{512^2 N^8|\mathcal{A}|^4}{\eta_{min}^3}\right) + \ln|\Phi_N| + \ln(6H/\delta)\right).$$

Note that the algorithm sets these quantities in terms of a parameter η instead of η_{min}, which may not be known. As long as $\eta \leq \eta_{min}$ our settings of n_{psdp} and n_{reg} certify that Ψ_h is a $1/2$-policy cover for $\mathcal{S}_h$. Appealing to the induction, this establishes the policy cover guarantee.

The reward sensitive step. Equipped with the policy covers, a single call to PSDP with the external reward R and an application of Theorem 5 yield the PAC guarantee. We have already accounted for the failure probability, so we must simply set n_{eval}. Applying Theorem 5 with the definition of $\delta_3 = \delta/(3H)$, we get

$$n_{\text{eval}} \geq \frac{64N^2H^2|\mathcal{A}|}{\epsilon^2}\ln\left(\frac{3H|\Pi|}{\delta}\right).$$

Sample complexity. As we solve H supervised learning problem, make NH calls to PSDP with parameter n_{psdp}, and make 1 call to PSDP with parameter n_{eval}, the sample complexity, measured in trajectories, is

$$H \cdot n_{\text{reg}} + NH^2 n_{\text{psdp}} + H n_{\text{eval}} =$$
$$\widetilde{\mathcal{O}}\left(\frac{N^8|\mathcal{A}|^4 H}{\eta_{min}^3} + \frac{N^6|\mathcal{A}|H}{\eta_{min}^3}\ln(|\Phi_N|/\delta) + \left(\frac{N^5H^4|\mathcal{A}|}{\eta_{min}^2} + \frac{N^2H^3|\mathcal{A}|}{\epsilon^2}\right)\ln(|\Pi|/\delta)\right).$$

Computational complexity. The running time is simply the time required to collect this many trajectories, plus the time required for all of the calls to the oracle. If T is the number of trajectories, the running time is

$$\mathcal{O}\left(HT + H\text{Time}_{\text{reg}}(n_{\text{reg}}) + NH^2\text{Time}_{\text{pol}}(n_{\text{psdp}}) + H\text{Time}_{\text{pol}}(n_{\text{eval}})\right).$$

F Supporting results

The next lemma is the well-known performance difference lemma, which has appeared in much prior work [6, 27, 43, 14]. Our version, which is adapted to episodic problems, is most closely related to Lemma 4.3 of [43]. We provide a short proof for completeness.

Lemma 13 (Performance difference lemma)**.** *For any episodic decision process with any reward function R, and any two non-stationary policies $\pi^{(1)}_{1:H}$ and $\pi^{(2)}_{1:H}$, let $Q^{(1)}_h \in \Delta(\mathcal{X}_h)$ be the distribution at time h induced by policy $\pi^{(1)}_{1:H}$. Then we have*

$$V(\pi^{(1)}_{1:H}; R) - V(\pi^{(2)}_{1:H}) = \sum_{h=1}^{H} \mathbb{E}_{x_h \sim Q^{(1)}_h} \left[V(x_h; \pi^{(1)}_h \circ \pi^{(2)}_{h+1:H}) - V(x_h; \pi^{(2)}_{h:H}) \right].$$

Proof. The proof is a standard telescoping argument.

$$\begin{aligned} V(\pi^{(1)}_{1:H}; R) - V(\pi^{(2)}_{1:H}) &= V(\pi^{(1)}_{1:H}; R) - V(\pi^{(1)}_1 \circ \pi^{(2)}_{2:H}; R) + V(\pi^{(1)}_1 \circ \pi^{(2)}_{2:H}; R) - V(\pi^{(2)}_{1:H}) \\ &= V(\pi^{(1)}_1 \circ \pi^{(2)}_{2:H}; R) - V(\pi^{(2)}_{1:H}) + \mathbb{E}_{x_2 \sim Q^{(1)}_2} \left[V(\pi^{(1)}_{2:H}; R) - V(\pi^{(2)}_{2:H}; R) \right]. \end{aligned}$$

The result follows by repeating this argument on the second term. □

The next result is Bernstein's inequality. The proof can be found in a number of textbooks [c.f., 9].

Proposition 11 (Bernstein's inequality)**.** *If $U_1, \ldots, U_n$ are independent zero-mean random variables with $|U_t| \le R$ a.s., and $\frac{1}{n}\sum_{t=1}^{n} \mathrm{Var}(U_t) \le \sigma^2$, then, for any $\delta \in (0,1)$, with probability at least $1-\delta$ we have*

$$\frac{1}{n}\sum_{t=1}^{n} U_t \le \sqrt{\frac{2\sigma^2 \ln(1/\delta)}{n}} + \frac{2R\ln(1/\delta)}{3n}.$$

The next proposition concerns learning with square loss, using a function class $\mathcal{G}$ with parametric metric entropy growth rate. Let D be a distribution over (x, y) pairs, where $x \in \mathcal{X}$ is an example space and $y \in [0,1]$. With a sample $\{(x_i, y_i)\}_{i=1}^n \overset{iid}{\sim} D$ and a function class $\mathcal{G} : \mathcal{X} \to [0,1]$, we may perform empirical risk minimization to find

$$\hat{g} := \operatorname*{argmin}_{g \in \mathcal{G}} \hat{R}_n(g) := \operatorname*{argmin}_{g \in \mathcal{G}} \frac{1}{n}\sum_{i=1}^{n} (g(x_i) - y_i)^2. \tag{3}$$

The population risk and minimizer are defined as

$$g^\star := \operatorname*{argmin}_{g \in \mathcal{G}} R(g) := \operatorname*{argmin}_{g \in \mathcal{G}} \mathbb{E}_{(x,y)\sim D} (g(x) - y)^2.$$

We assume *realizability*, which means that the Bayes optimal classifier $x \mapsto \mathbb{E}_D[y \mid x]$ is in our class, and as this minimizes the risk over all functions we know that $g^\star(x)$ is precisely this classifier.

We assume that $\mathcal{G}$ has "parametric" pointwise metric entropy growth rate, which means that the pointwise covering number at scale ε, which we denote $\mathcal{N}(\mathcal{G}, \varepsilon)$ scales as $\mathcal{N}(\mathcal{G}, \varepsilon) \le c_0 d \ln(1/\varepsilon)$, for a universal constant $c_0 > 0$. Recall that for a function class $\mathcal{G} : \mathcal{X} \to \mathbb{R}$ the pointwise covering number at scale ε is the size of the smallest set $V : \mathcal{X} \to \mathbb{R}$ such that

$$\forall g \in \mathcal{G}, \exists v \in V : \sup_x |g(x) - v(x)| \le \varepsilon.$$

With the above definitions, we can state the main guarantee for the empirical risk minimizer.

Proposition 12. *Fix $\delta \in (0,1)$. Let $\hat{g}$ be the empirical risk minimizer in (3) based on n samples from a distribution D. If $\mathcal{G}$ is realizable for D and has parametric entropy growth rate, then with probability at least $1-\delta$ we have*

$$\mathbb{E}_{(x,y)\sim D}\left[(\hat{g}(x)-g^\star(x))^2\right] \le \Delta_{reg}, \text{ with } \Delta_{reg} := \inf_{\varepsilon>0}\left\{6\varepsilon + \frac{8\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}\right\},$$

where $C>0$ is a universal constant.

The result here is a standard square loss excess risk bound, and it is perhaps the simplest such bound for well-specified infinite function classes. Sharper guarantees based on using empirical covering numbers, combinatorial parameters [3], or localization [34], are possible, and completely composable with the rest of our arguments. In other words, if a bound similar to the one in Proposition 12 is achievable under different assumptions (e.g., different measure of statistical complexity), we can incorporate it into the proof of Theorem 2. We state, prove, and use this simple bound to keep the arguments self contained.

Proof. Define

$$Z_i(g) = (g(x_i)-y_i)^2 - (g^\star(x_i)-y_i)^2.$$

Using the realizability assumption that $\mathbb{E}[y \mid x] = g^\star(x)$, it is easy to verify that

$$\mathbb{E}[Z_i(g)] = \mathbb{E}[g(x)^2 - g^\star(x)^2 - 2y(g(x)-g^\star(x))] = \mathbb{E}[(g(x)-g^\star(x))^2].$$

The variance is similarly controlled:

$$\begin{aligned}\mathrm{Var}[Z_i(g)] &\le \mathbb{E}[Z_i(g)^2] = \mathbb{E}[(g(x)+g^\star(x)-2y)^2(g(x)-g^\star(x)^2] \\ &\le 4\mathbb{E}[(g(x)-g^\star(x)^2] = 4\mathbb{E}[Z_i(g)],\end{aligned}$$

where we use that $y, g(x), g^\star(x)$ are in $[0,1]$. Therefore, via Bernstein's inequality (Proposition 11), with probability at least $1-\delta$ we have

$$\left|\frac{1}{n}\sum_i Z_i(g) - \mathbb{E}[Z(g)]\right| \le 2\sqrt{\frac{\mathbb{E}[Z(g)]\ln(2/\delta)}{n}} + \frac{2\ln(2/\delta)}{n}. \tag{4}$$

For the uniform convergence step, we show that $Z_i(g)$ is a Lipschitz function in g:

$$|Z_i(g) - Z_i(g')| = \left|(g(x_i)-g'(x_i))^2(g(x_i)+g'(x_i)-2y_i)^2\right| \le 4\left|g(x_i)-g'(x_i)\right|,$$

where we again use that $y_i, g(x_i)$ and $g'(x_i)$ are in $[0,1]$.

Now let V be a pointwise cover of $\mathcal{G}$ at scale ε, so that for any $g\in\mathcal{G}$ there exits $v_g \in V$ such that:

$$\sup_x |v_g(x) - g(x)| \le \varepsilon.$$

By our metric entropy assumptions, we have that $|V| \le \mathcal{N}(\mathcal{G},\varepsilon) \le \varepsilon^{-c_0 d}$. For any $g\in\mathcal{G}$ we have

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n Z_i(g) &\le \varepsilon + \frac{1}{n}\sum_{i=1}^n Z_i(v_g) \le \varepsilon + \mathbb{E}[Z(v_g)] + 2\sqrt{\frac{\mathbb{E}[Z(v_g)]\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}} + \frac{2\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n} \\ &\le \varepsilon + 2\mathbb{E}[Z(v_g)] + \frac{3\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n} \le 3\varepsilon + 2\mathbb{E}[Z(v_g)] + \frac{3\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}.\end{aligned}$$

Here we are applying (4) uniformly over the pointwise cover, using the fact that $2\sqrt{ab} \leq a + b$, and using the pointwise covering property. Similarly we can control the other tail

$$\mathbb{E}[Z(g)] \leq \varepsilon + \mathbb{E}[Z(v_g)] \leq \varepsilon + \frac{1}{n}\sum_{i=1}^{n} Z_i(v_g) + 2\sqrt{\frac{\mathbb{E}[Z(v_g)]\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}} + \frac{2\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}$$
$$\leq \frac{5}{2}\varepsilon + \frac{1}{n}\sum_{i=1}^{n} Z_i(g) + \frac{1}{2}\mathbb{E}[Z(g)] + \frac{4\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}$$

Re-arranging and putting the two bounds together, the following holds simultaneously for all $g \in \mathcal{G}$, with probability at least $1 - \delta$

$$\frac{1}{2}\mathbb{E}[Z(g)] - 3\varepsilon - \frac{4\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n} \leq \frac{1}{n}\sum_{i=1}^{n} Z_i(g) \leq 2\mathbb{E}[Z(g)] + 3\varepsilon + \frac{4\ln(2\mathcal{N}(\mathcal{G},\varepsilon)/\delta)}{n}. \qquad \square$$

G Experimental Setup and Optimization Details

Learning Details for HOMER. We describe the details of the oracle and hyperparameter below:
Oracle Implementation: We implement the REG subroutine by performing supervised binary classification instead of regression. Formally, we train the model $p_w(. \mid x', a, x)$ on a training data $\{(x_i, a_i, x'_i, y_i)\}_{i=1}^n$ as shown below:

$$\max_w \frac{1}{n}\sum_{i=1}^{n} \ln p_w\left(y_i \mid x_i, a_i, x'_i\right)$$

We use Adam optimization with mini batches. We separate a validation set from the training data. We train the model for a maximum number of epochs and use the model with the best validation performance. We first train the models without bottleneck and use it for initialization. The two training procedures are identical barring the difference in the models.

We learn policies for the CB subroutine by training a model to predict the immediate reward using mean squared loss error. This is equivalent to one-step Q-learning. Formally, we train a model $Q_\theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ on training data $\{(x_i, a_i, r_i)\}_{i=1}^m$ as shown below:

$$\max_\theta \frac{1}{m}\sum_{i=1}^{m} \left(Q_\theta(x_i, a_i) - r_i\right)^2.$$

The policy corresponding to Q_θ deterministically takes the action $\arg\max_{a\in\mathcal{A}} Q_\theta(x, a)$. We use Adam optimization with mini batches and terminate after a fixed number of epochs. We do not use a validation dataset for CB and use the policy model learned after the final epoch.

We use the two empirical optimizations described in Section 5 which provide significant computational and statistical advantage. Hyperparameter values for the diabolical combination lock problem can be found in Table 2. We use the PyTorch library (version 1.1.0) for implementing HOMER.[10] We use the standard mechanism provided by PyTorch for initializing the parameters.

[10] https://pytorch.org/

Hyperparameter	Value
Learning Rate	0.001 (for both REG and CB)
Batch size	32 (for both REG and CB)
n_{reg}	We sample $10,000$ observed transitions from which we generate additional $10,000$ imposter transitions.
n_{psdp}	$20,000$
N (capacity of a backward state abstraction model)	2
M (capacity of a forward state abstraction model)	3
Maximum epochs for REG	200
Maximum epochs for CB	50
Validation data size (REG)	20% of the size of training data for REG.
Hidden layer size for $\mathcal{F}_N$	56
Gumbel-Softmax temperature	1

Table 2: HOMER Hyperparameters

Learning Details for PCID. We use the code made publicly available by the authors.[11] PCID uses a model for predicting the previous state and action and performs k-means clustering on the predicted probabilities. We used a linear model which is representationally sufficient for the diabolical combination lock problem. We used the hyperparameter setting recommended by the authors who evaluated on a combination lock problem similar to ours. One key exception was the the data collection hyperparameter (n) used by their state decoding algorithm. We tried a value of n in $\{200, 1000, 2000\}$. We list the hyperparameter choice in Table 3.

Hyperparameter	Value
Learning Rate	0.03
n	200
Number of clusters for k-means	3

Table 3: PCID Hyperparameters

Learning Details for A2C, PPO, A2C + RND and PPO + RND. We train each baseline for a maximum of 10 million episodes. All baseline models use fully-connected, 2-layer MLPs with 64 hidden units and ReLU non-linearities. For each baseline, we used the RMSProp optimizer [51] and tuned learning rates over $\{0.01, 0.001, 0.0001\}$. For the methods which use the RND bonus, the random networks were 2-layer MLPs with 100 hidden units and ReLU non-linearities. We found that tuning the intrinsic reward coefficient λ_I was important to obtain good performance, and performed an initial search over $\lambda_I \in \{1, 10, 100, 500\}$ for $H \in \{6, 12, 25\}$ for A2C + RND and found that $\lambda_I = 100$ worked best across different horizon lengths. We then used this value for all subsequent experiments. We also experimented with higher entropy bonuses for A2C for $H = 6$, but this did not yield any improvement so we kept the default value of 0.01 for subsequent experiments. We used the PPO and A2C implementations provided in [45] and kept other hyperparameters fixed at their default values. We list the hyperparameter values for A2C and A2C + RND in Table 4 and for PPO and PPO + RND in Table 5.

[11] https://github.com/Microsoft/StateDecoding

Hyperparameter	Value
Learning Rate	0.001
Rollout length	H
γ	0.99
τ_{GAE}	0.95
Gradient clipping	0.5
Entropy Bonus	0.01
Extrinsic Reward coefficient λ_I	1.0
Intrinsic Reward coefficient λ_E (for A2C + RND)	100

Table 4: A2C and A2C + RND Hyperparameters

Hyperparameter	Value
Learning Rate	0.001
Rollout length	H
γ	0.99
τ_{GAE}	0.95
Gradient clipping	5
Entropy Bonus	0.01
Optimization Epochs	10
Minibatch size	160
Ratio clip	0.2
Extrinsic Reward coefficient λ_I	1.0
Intrinsic Reward coefficient λ_E (for PPO + RND)	100

Table 5: PPO and PPO + RND Hyperparameters