Confident Approximate Policy Iteration for Efficient Local Planning in q^π-realizable MDPs

Gellért Weisz
DeepMind, London, UK
University College London, London, UK

András György
DeepMind, London, UK

Tadashi Kozuno
University of Alberta, Edmonton, Canada
Omron Sinic X, Tokyo, Japan

Csaba Szepesvári
DeepMind, London, UK
University of Alberta, Edmonton, Canada

Abstract

We consider approximate dynamic programming in γ-discounted Markov decision processes and apply it to approximate planning with linear value-function approximation. Our first contribution is a new variant of APPROXIMATE POLICY ITERATION (API), called CONFIDENT APPROXIMATE POLICY ITERATION (CAPI), which computes a deterministic stationary policy with an optimal error bound scaling linearly with the product of the effective horizon H and the worst-case approximation error ε of the action-value functions of stationary policies. This improvement over API (whose error scales with H^2) comes at the price of an H-fold increase in memory cost. Unlike Scherrer and Lesner [2012], who recommended computing a non-stationary policy to achieve a similar improvement (with the same memory overhead), we are able to stick to stationary policies. This allows for our second contribution, the application of CAPI to planning with local access to a simulator and d-dimensional linear function approximation. As such, we design a planning algorithm that applies CAPI to obtain a sequence of policies with successively refined accuracies on a dynamically evolving set of states. The algorithm outputs an $\tilde{O}(\sqrt{d}H\varepsilon)$-optimal policy after issuing $\tilde{O}(dH^4/\varepsilon^2)$ queries to the simulator, simultaneously achieving the optimal accuracy bound and the best known query complexity bound, while earlier algorithms in the literature achieve only one of them. This query complexity is shown to be tight in all parameters except H. These improvements come at the expense of a mild (polynomial) increase in memory and computational costs of both the algorithm and its output policy.

1 Introduction

A key question in reinforcement learning is how to use value-function approximation to arrive at scaleable algorithms that can find near-optimal policies in Markov decision processes (MDPs). A flurry of recent results aims at solving this problem efficiently with varying models of interaction with the MDP. In this paper we focus on the problem of *planning with a simulator* when using linear function approximation. A simulator is a "device" that, given a state-action pair as a query, returns a next state and reward generated from the transition kernel of the MDP that is simulated. Depending on the application, such a simulator is often readily available (e.g., in chess, go, Atari). Planning with simulator access comes with great benefits: for example, in a recent work, Wang et al. [2021] showed that under some conditions it is exponentially more efficient to find a near-optimal policy if a simulator of the MDP (that can reset to a state) is available compared to the online case where a learner interacts with its environment by following trajectories but without the help of a simulator.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Our setting of *offline, local planning* considers the problem of finding a policy with near-optimal value at a given initial state s_0 in the MDP. The planner can issue queries to the simulator, and has to find and output a near-optimal policy with high probability. The efficiency of a planner is measured in four ways: the *suboptimality* of the policy found, that is, how far its value is from that of the optimal policy; the *query cost*, that is, the number of queries issued to the simulator; the *computational cost*, which is the number of operations used; and the *memory cost*, which is the amount of memory used (we adopt the real computation model for these costs). There are several interaction models between the planner and the simulator [Yin et al., 2022]. The most permissive one is called the *generative model*, or *random access*. Here, the planner receives the set of all states and is allowed to issue queries for any state and action. Coding a simulator that supports this model can be challenging, as oftentimes the set of states is computationally difficult to describe. Instead of *random access*, we consider the more practical and more challenging *local access* setting, where the planner only sees the initial state and the set of states received as a result to a query to the simulator. Consequently, the queries issued have to be for a state that has already been encountered this way (and any available action), while the simulator needs to support the ability to reset the MDP state *only* to previously seen states. A simple approach in practice to support this model is saving and reloading checkpoints during the operation of the simulator.

To handle large, possibly infinite state spaces, we use linear function approximation to approximate the action-value functions q^π of stationary, deterministic policies π (for background on MDPs, see the next section). A feature-map is a good fit to an MDP if the worst-case error of using the feature-map to approximate value functions of policies of the MDP is small:

Definition 1.1 (q^π-realizability: uniform policy value-function approximation error)**.** *Given an MDP, the uniform policy value-function approximation error induced by a feature map φ, which maps state-action pairs (s, a) to the Euclidean ball of radius L centered at zero in $\mathbb{R}^d$, over a set of parameters belonging to the d-dimensional centered Euclidean ball of radius B is*

$$\varepsilon = \sup_{\pi} \inf_{\theta:\|\theta\|_2 \le B} \sup_{(s,a)} |q^\pi(s,a) - \langle \varphi(s,a), \theta \rangle| ,$$

where the outermost supremum is over all possible stationary deterministic memoryless policies (i.e., maps from states to actions) of the MDP.

Our goal is to design algorithms that scale gracefully with the uniform approximation error ε at the expense of controlled computational cost. To achieve nontrivial guarantees, the uniform approximation error needs to be "small". This (implicit) assumption is stronger than the $q^\star$-realizability assumption (where the approximation error is only considered for optimal policies), which Weisz et al. [2021] showed an exponential query complexity lower bound for. At the same time, it is (strictly) weaker than the linear MDP assumption [Zanette et al., 2020], for which there are efficient algorithms to find a near-optimal policy in the online setting (without a simulator) [Jin et al., 2020], even in the more challenging reward-free setting where the rewards are only revealed after exploration [Wagenmaker et al., 2022].

In the *local access* setting, the planner learns the features $\varphi(s, a)$ of a state-action pair *only* for those states s that have already been encountered. In contrast, in the *random access* setting, the whole feature map $\varphi(\cdot, \cdot)$, of (possibly infinite) size $d|\mathcal{S}||\mathcal{A}|$ (where $\mathcal{S}$ and $\mathcal{A}$ are the state and action sets, resp.), is given to the planner as input. In the latter setting, when only the query cost is counted, Du et al. [2019] and Lattimore et al. [2020] proposed algorithms (the latter working in the misspecified, $\varepsilon > 0$ regime) that issue a number of queries that is polynomial in the relevant parameters, but require a barycentric spanner or near-optimal design of the input features. In the worst case, computing any of these sets scales polynomially in $|\mathcal{S}|$ and $|\mathcal{A}|$, which can be prohibitive.

In the case of *local access*, considered in this paper, the best known bound on the suboptimality of the computed policy is achieved by CONFIDENT MC-POLITEX [Yin et al., 2022]. In the more permissive *random access* setting, the best known query cost is achieved by Lattimore et al. [2020]. Our algorithm, CAPI-QPI-PLAN (given in Algorithm 3), achieves the *best of both* while only assuming *local access*. This is shown in the next theorem; in the theorem ε is as defined in Definition 1.1, γ is the discount factor, and $v^\star$ and v^π are the state value functions associated with the optimal policy and policy π, respectively (precise definitions of these quantities are given in the next section). A comparison to other algorithms in the literature is given in Table 1; there the accuracy parameter ω of the algorithms is set to ε, but a larger ω can be used to trade off suboptimality guarantees for an improved query cost.

Theorem 1.2. *For any confidence parameter $\delta \in (0, 1]$, accuracy parameter $\omega > 0$, and initial state $s_0 \in \mathcal{S}$, with probability at least $1 - \delta$,* CAPI-QPI-PLAN *(Algorithm 3) finds a policy π with*

$$v^\star(s_0) - v^\pi(s_0) = \tilde{O}\left((\varepsilon + \omega)\sqrt{d}(1-\gamma)^{-1}\right), \tag{1}$$

while executing at most $\tilde{O}\left(d(1-\gamma)^{-4}\omega^{-2}\right)$ queries in the local access *setting.*

CAPI-QPI-PLAN is based on CONFIDENT MC-LSPI, another algorithm of Yin et al. [2022], which relies on policy iteration from a *core set* of informative state-action pairs, but achieves inferior performance both in terms of suboptimality and query complexity. However, CAPI-QPI-PLAN's improvements come at the expense of increased memory and computational costs, as shown in the next theorem: compared to CONFIDENT MC-LSPI, the memory and computational costs of our algorithm increase by a factor of the effective horizon $H = \tilde{O}(1/(1-\gamma))$, and the policy computed by CAPI-QPI-PLAN uses a dH factor more memory for storage and a d^2H factor more computation to evaluate.

Theorem 1.3 (Memory and computational cost)**.** *The memory and computational cost of running* CAPI-QPI-PLAN *(Algorithm 3) are $\tilde{O}\left(d^2/(1-\gamma)\right)$ and $\tilde{O}\left(d^4|\mathcal{A}|(1-\gamma)^{-5}\omega^{-2}\right)$, respectively, while the memory and computational costs of storing and evaluating the final policy outputted by* CAPI-QPI-PLAN*, respectively, are $\tilde{O}\left(d^2/(1-\gamma)\right)$ and $\tilde{O}\left(d^3|\mathcal{A}|/(1-\gamma)\right)$.*

Next we present a lower bound corresponding to Theorem 1.2 that holds even in the more permissive *random access* setting, and shows that CAPI-QPI-PLAN trades of the query cost and the suboptimality of the returned policy asymptotically optimally up to its dependence on $1/(1-\gamma)$:

Theorem 1.4 (Query cost lower bound)**.** *Let $\alpha \in (0, \frac{0.1\gamma}{(1-\gamma)(1+\gamma)^2})$, $\delta \in (0, 0.2)$, $\gamma \in [\frac{7}{12}, 1]$, $d \geq 3$, and $\varepsilon \geq 0$. Then there is a class $\mathcal{M}$ of MDPs with uniform policy value-function approximation error at most ε such that any planner that guarantees to find an α-optimal policy π (i.e., $v^\star(s_0) - v^\pi(s_0) \leq \alpha$) with probability at least $1 - \delta$ for all $M \in \mathcal{M}$ when used with a simulator for M with* random access*, the worst-case (over $\mathcal{M}$) expected number of queries issued by the planner is at least*

$$\max\left(\exp\left(\Omega\left(\frac{d\varepsilon^2}{\alpha^2(1-\gamma)^2}\right)\right), \Omega\left(\frac{d^2}{\alpha^2(1-\gamma)^3}\right)\right). \tag{2}$$

If ω is set to ε for CAPI-QPI-PLAN, the first term of Eq. (2) implies that any planner with an asymptotically smaller (apart from logarithmic factors) suboptimality guarantee than Eq. (1) executes exponentially many queries in expectation. The second term of Eq. (2), which is shown to be a lower bound in Theorem H.3 even in the more general setting of linear MDPs with zero misspecification ($\varepsilon = 0$), matches the query complexity of Theorem 1.2 up to an $\tilde{O}((1-\gamma)^2)$ factor. Thus, the lower bound implies that the suboptimality and query cost bounds of Theorem 1.2 are tight up to logarithmic factors in all parameters except the $1/(1-\gamma)$-dependence of the query cost bound.

At the heart of our method is a new algorithm, which we call CONFIDENT APPROXIMATE POLICY ITERATION (CAPI). This algorithm, which belongs to the family of approximate dynamic programming algorithms [Bertsekas, 2012, Munos, 2003, 2005], is a novel variant of APPROXIMATE POLICY ITERATION (API) [Bertsekas and Tsitsiklis, 1996]: in the policy improvement step, CAPI only updates the policy in states where it is confident that the update will improve the performance. This simple modification allows CAPI to avoid the problem of "classical" approximate dynamic programming algorithms (approximate policy and value iteration) of inflating the value function evaluation error by a factor of H^2 where $H = \tilde{O}(1/(1-\gamma))$ (for discussions of this problem, see also the papers by Scherrer and Lesner, 2012 and Russo, 2020), and reduce this inflation factor to H. A similar result has already been achieved by Scherrer and Lesner [2012], who proposed to construct a non-stationary policy that strings together all policies obtained while running either approximate value or policy iteration. However, applying this result to our planning problem is problematic, since the policies to be evaluated are non-stationary, and hence including them in the policy set we aim to approximate may drastically increase the error ε as compared to Definition 1.1, which only considers stationary memoryless policies.

While the improvements provided by CAPI allows CAPI-QPI-PLAN to match the performance of CONFIDENT MC-POLITEX in terms of suboptimality, it is unlikely that a simple modification of CONFIDENT MC-POLITEX would lead to an algorithm which matches CAPI-QPI-PLAN's

Table 1: Comparison of suboptimality and query complexity guarantees of various planners (with the approximation accuracy parameter ω set to ε). Drawbacks are highlighted with red, the best bounds with blue.

Algorithm [Publication]	Query cost	Suboptimality	Access model
MC-LSPI [Lattimore et al., 2020]	$\tilde{O}\left(\frac{d}{\varepsilon^2(1-\gamma)^4}\right)$	$\tilde{O}\left(\frac{\varepsilon\sqrt{d}}{(1-\gamma)^2}\right)$	random access
CONFIDENT MC-LSPI [Yin et al., 2022]	$\tilde{O}\left(\frac{d^2}{\varepsilon^2(1-\gamma)^4}\right)$	$\tilde{O}\left(\frac{\varepsilon\sqrt{d}}{(1-\gamma)^2}\right)$	local access
CONFIDENT MC-POLITEX [Yin et al., 2022]	$\tilde{O}\left(\frac{d}{\varepsilon^4(1-\gamma)^5}\right)$	$\tilde{O}\left(\frac{\varepsilon\sqrt{d}}{1-\gamma}\right)$	local access
CAPI-QPI-PLAN [This work]	$\tilde{O}\left(\frac{d}{\varepsilon^2(1-\gamma)^4}\right)$	$\tilde{O}\left(\frac{\varepsilon\sqrt{d}}{1-\gamma}\right)$	local access

performance in terms of query cost (see Table 1): Both methods evaluate a sequence of policies at an $\tilde{O}(\varepsilon)$ accuracy each (requiring $\tilde{O}(1/\varepsilon^2)$ queries, omitting the dependence on other parameters). However, while CAPI-QPI-PLAN (and CONFIDENT MC-LSPI) evaluates $O(\log(1/\varepsilon))$ (again in terms of ε only) policies to find one which is $\tilde{O}(\varepsilon)$-optimal, CONFIDENT MC-POLITEX needs to compute $\tilde{O}(1/\varepsilon^2)$ policies to achieve the same. As a consequence, CONFIDENT MC-POLITEX only achieves $\tilde{O}(1/\varepsilon^4)$ query complexity, and to match CAPI-QPI-PLAN's $\tilde{O}(1/\varepsilon^2)$ complexity, one would need to come up with either significantly better policy evaluation methods (potentially using the similarity in the subsequent policies) or a much faster (exponential vs. square-root) convergence rate in the suboptimality of the policy sequence produced by CONFIDENT MC-POLITEX.

The rest of the paper is organized as follows: The model and notation are introduced in Section 2. CAPI is introduced and analyzed in Section 3. Planning with q^π-realizability is introduced in Section 4, with CAPI-QPI-PLAN being built-up and analyzed in Sections 4.1 and 4.2. In particular, the proof of Theorem 1.2 is given in Section 4.2. Several proofs are relegated to appendices, in particular, Theorem 1.3 is proved and implementation details of CAPI-QPI-PLAN are discussed in Appendix G, while Theorem 1.4 is proved in Appendix H.

2 Notation and preliminaries

Let $\mathbb{N} = \{0, 1, \dots\}$ denote the set of natural numbers, $\mathbb{N}^+ = \{1, 2, \dots\}$ the positive integers. For some integer i, let $[i] = \{0, \dots, i-1\}$. For $x \in \mathbb{R}$, let $\lceil x \rceil$ denote the smallest integer i such that $i \geq x$. For a positive definite $V \in \mathbb{R}^{d\times d}$ and $x \in \mathbb{R}^d$, let $\|x\|_V^2 = x^\top V x$. For matrices A and B, we say that $A \geq B$ if $A - B$ is positive semidefinite. Let $\mathbb{I}$ be the d-dimensional identity matrix. For compatible vectors x, y, let $\langle x, y\rangle$ be their inner product: $\langle x, y\rangle = x^\top y$. Let $\mathcal{M}_1(X)$ denote the space of probability distributions supported on the set X (throughout, we assume that the σ-algebra is implicit). We write $a \approx_\varepsilon b$ for $a, b, \varepsilon \in \mathbb{R}$ if $|a - b| \leq \varepsilon$. We denote by $\tilde{O}(\cdot)$ and $\tilde{\Theta}(\cdot)$ the variants of the big-O notation that hide polylogarithmic factors.

A Markov Decision Process (MDP) is a tuple $M = (\mathcal{S}, \mathcal{A}, Q)$, where $\mathcal{S}$ is a measurable state space, $\mathcal{A}$ is a finite action space, and $Q : \mathcal{S} \times \mathcal{A} \to \mathcal{M}_1(\mathcal{S} \times [0, 1])$ is the transition-reward kernel. We define the transition and reward distributions $P : \mathcal{S} \times \mathcal{A} \to \mathcal{M}_1(\mathcal{S})$ and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathcal{M}_1([0, 1])$ as the marginals of Q. By a slight abuse of notation, for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, let $P(\cdot|s, a)$ and $\mathcal{R}(\cdot|s, a)$ denote the distributions $P(s, a)$ and $\mathcal{R}(s, a)$, respectively. We further denote by $r(s, a) = \int_0^1 x \, \mathrm{d}\mathcal{R}(x|s, a)$ the expected reward for an action $a \in \mathcal{A}$ taken in a state $s \in \mathcal{S}$. Without loss of generality, we assume that there is a designated initial state $s_0 \in \mathcal{S}$.

Starting from any state $s \in \mathcal{S}$, a stationary memoryless policy $\pi : \mathcal{S} \to \mathcal{M}_1(\mathcal{A})$ interacts with the MDP in a sequential manner for time-steps $t \in \mathbb{N}$, defining a probability distribution $\mathcal{P}_{\pi,s}$ over the episode trajectory $\{S_i, A_i, R_i\}_{i\in\mathbb{N}}$ as follows: $S_0 = s$ deterministically, $A_i \sim \pi(S_i)$, and $(S_{i+1}, R_i) \sim Q(S_i, A_i)$. By a slight variation, let $\mathcal{P}_{\pi,s,a}$ denote (for some $a \in \mathcal{A}$) the distribution of the trajectory when $A_0 = a$ deterministically, while the distribution of the rest of the trajectory is defined analogously.

This allows us to conveniently define the expected state-value and action-value functions in the discounted setting we consider, for some discount factor $0 < \gamma < 1$, respectively, as

$$v^\pi(s) = \mathbb{E}_{\pi,s}\left[\sum_{t\in\mathbb{N}} \gamma^t R_t\right] \quad \text{and} \quad q^\pi(s, a) = \mathbb{E}_{\pi,s,a}\left[\sum_{t\in\mathbb{N}} \gamma^t R_t\right] \qquad \text{for all } (s, a) \in \mathcal{S} \times \mathcal{A}\,, \quad (3)$$

where throughout the paper we use the convention that $\mathbb{E}_\bullet$ is the expectation operator corresponding to a distribution $\mathcal{P}_\bullet$ (e.g., $\mathbb{E}_{\pi,s}$ is the expectation with respect to $\mathcal{P}_{\pi,s}$). It is well known (see, e.g., Puterman, 1994) that there exists an optimal stationary deterministic memoryless policy $\pi^\star$ such that

$$\sup_\pi v^\pi(s) = v^{\pi^\star}(s) \qquad \text{and} \qquad \sup_\pi q^\pi(s,a) = q^{\pi^\star}(s,a) \qquad \text{for all } (s,a) \in \mathcal{S} \times \mathcal{A}\,.$$

Let $v^\star = v^{\pi^\star}$ and $q^\star = q^{\pi^\star}$. For any policy π, v^π and q^π are known to satisfy the Bellman equations [Puterman, 1994]:

$$v^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) q^\pi(s,a) \text{ and } q^\pi(s,a) = r(s,a) + \gamma \int_{s'\in\mathcal{S}} v^\pi(s')\, \mathrm{d}P(s'|s,a) \text{ for all } (s,a) \in \mathcal{S}\times\mathcal{A}. \quad (4)$$

Finally, we call a policy π deterministic if for all states, $\pi(s)$ is a distribution that assigns unit weight to one action and zero weight to the others. With a slight abuse of notation, for a deterministic policy π, we denote by $\pi(s)$ the action π chooses (deterministically) in state $s \in \mathcal{S}$.

3 Confident Approximate Policy Iteration

In this section we introduce CONFIDENT APPROXIMATE POLICY ITERATION (CAPI), our new approximate dynamic programming algorithm. In approximate dynamic programming, the methods are designed around oracles that return either an approximation to the application of the Bellman optimality operator to a value function ("approximate value iteration"), or an approximation to the value function of some policy ("approximate policy iteration"). Our setting is the second. The novelty is that we assume access to the accuracy of the approximation and use this knowledge to modify the policy update, which leads to improved guarantees on the suboptimality of the computed policy.

We present the pseudocodes of API [Bertsekas and Tsitsiklis, 1996] and CAPI jointly in Algorithm 1: starting from an arbitrary (deterministic) policy π_0, the algorithm iterates a policy estimation (Line 2) and a policy update step (Line 3) I times. The policy update for API is greedy with respect to the action-value estimates $\hat{q}$ and is defined as $\pi_{\hat{q}}(s) = \arg\max_{a\in\mathcal{A}} \hat{q}(s,a)$. We assume that $\arg\max_{a\in\mathcal{A}}$ breaks ties in a consistent manner by ordering the actions (using the notation $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_{|\mathcal{A}|})$) and always choosing action $\mathcal{A}_i$ with the lowest index i that achieves the maximum. For CAPI, the policy update further relies on a global estimation-accuracy parameter ω, and a set of fixed-states $\mathcal{S}_{\text{fix}} \subseteq \mathcal{S}$. For the purposes of this section, it is enough to keep $\mathcal{S}_{\text{fix}} = \{\}$. CAPI updates the policy to one that acts greedily with respect to $\hat{q}$ *only* on states that are not in $\mathcal{S}_{\text{fix}}$ and where it is confident that this leads to an improvement over the previous policy (Case 5a); otherwise, the new policy will return the same action as the previous one (Case 5b). To decide, $\hat{q}(s,\pi(s)) + \omega$ is treated as the upper bound on the previous policy's value, and $\max_{a\in\mathcal{A}} \hat{q}(s,a) - \omega$ as the lower bound of the action-value of the greedy action (Eq. 5):

$$\pi_{\hat{q},\pi,\mathcal{S}_{\text{fix}}}(s) = \begin{cases} \arg\max\limits_{a\in\mathcal{A}} \hat{q}(s,a)\,, & \text{if } s \notin \mathcal{S}_{\text{fix}} \text{ and } \hat{q}(s,\pi(s)) + \omega < \max_{a\in\mathcal{A}} \hat{q}(s,a) - \omega\,; \quad (5a) \\ \pi(s)\,, & \text{otherwise.} \quad (5b) \end{cases}$$

Note that $\pi_{\hat{q},\pi,\mathcal{S}_{\text{fix}}}$ also depends on ω, however, this dependence is omitted from the notation (as ω is kept fixed throughout).

CAPI can also be seen as a refinement of CONSERVATIVE POLICY ITERATION (CPI) of Kakade and Langford [2002] with some important differences: While CPI introduces a global parameter to ensure the update stays close to the previous policy, CAPI has no such parameter, and it dynamically decides when to stay close to (more precisely, use) the previous policy, individually for every state, based on whether there is evidence for a guaranteed improvement.

Let π be any stationary deterministic memoryless policy, $\hat{q}^\pi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ be any function, $\omega \in \mathbb{R}_+$, and $\mathcal{S}_{\text{fix}} \subseteq \mathcal{S}$. First, we show that as long as $\hat{q}^\pi$ is an ω-accurate estimate of q^π, the CAPI policy update only improves the policy's values:

Lemma 3.1 (No deterioration)**.** *Let $\pi' = \pi_{\hat{q}^\pi,\pi,\mathcal{S}_{fix}}$. Assume that for all $s \in \mathcal{S} \setminus \mathcal{S}_{fix}$ and $a \in \mathcal{A}$, $\hat{q}^\pi(s,a) \approx_\omega q^\pi(s,a)$. Then, for any $s \in \mathcal{S}$, $v^{\pi'}(s) \geq v^\pi(s)$.*

Proof. Fix any $s \in \mathcal{S}$. If $s \in \mathcal{S}_{\text{fix}}$ or $\hat{q}^\pi(s,\pi(s)) + \omega \geq \max_{a\in\mathcal{A}} \hat{q}^\pi(s,a) - \omega$, then $\pi'(s) = \pi(s)$ and therefore $q^\pi(s,\pi'(s)) = v^\pi(s)$. Otherwise, $s \notin \mathcal{S}_{\text{fix}}$ and $\hat{q}^\pi(s,\pi(s)) + \omega \leq \max_{a\in\mathcal{A}} \hat{q}^\pi(s,a) - \omega$, hence $\pi'(s) = \arg\max_{a\in\mathcal{A}} \hat{q}^\pi(s,a)$, and it follows by our assumptions that $q^\pi(s,\pi'(s)) \geq$

Algorithm 1 APPROXIMATE POLICY ITERATION (API) and CONFIDENT APPROXIMATE POLICY ITERATION (CAPI)

1: **for** $i = 1$ to I **do**
2: $\quad \hat{q} \leftarrow \text{ESTIMATE}(\pi_{i-1})$
3: $\quad \pi_i \leftarrow \begin{cases} \pi_{\hat{q}} & \text{API} \\ \pi_{\hat{q},\pi_{i-1},\mathcal{S}_{\text{fix}}} & \text{CAPI} \end{cases}$
4: **return** π_I

$\hat{q}^\pi(s,\pi'(s)) - \omega = \max_{a\in\mathcal{A}} \hat{q}^\pi(s,a) - \omega > \hat{q}^\pi(s,\pi(s)) + \omega \geq q^\pi(s,\pi(s)) = v^\pi(s)$. Therefore, in any case, $q^\pi(s,\pi'(s)) \geq v^\pi(s)$. Since this holds for any $s \in \mathcal{S}$, the Policy Improvement Theorem [Sutton and Barto, 2018, Section 4.2] implies that for any $s \in \mathcal{S}$, $v^{\pi'}(s) \geq v^\pi(s)$. □

Next we introduce two approximate optimality criterion for a policy on a set of states:

Definition 3.2 (Policy optimality on a set of states)**.** *A policy π is Δ-optimal (for some $\Delta \geq 0$) on a set of states $\mathcal{S}' \subseteq \mathcal{S}$, if for all $s \in \mathcal{S}'$, $v^\star(s) - v^\pi(s) \leq \Delta$.*

Definition 3.3 (Next-state optimality on a set of states)**.** *A policy π is next-state Δ-optimal (for some $\Delta \geq 0$) on a set of states $\mathcal{S}' \subseteq \mathcal{S}$, if for all $s \in \mathcal{S}'$ and all actions $a \in \mathcal{A}$, $\int_{s'\in\mathcal{S}} (v^\star(s') - v^\pi(s'))\, \mathrm{d}P(s'|s,a) \leq \Delta$.*

Note that in the special case of $\mathcal{S}' = \mathcal{S}$ the first property implies the second, that is, if π is Δ-optimal on $\mathcal{S}$, then it is also next-state Δ-optimal on $\mathcal{S}$. Next, we show that the suboptimality of a policy updated by CAPI evolves as follows (the proof is relegated to Appendix A):

Lemma 3.4 (Iteration progress)**.** *Let $\pi' = \pi_{\hat{q}^\pi,\pi,\mathcal{S}_{fix}}$. Assume that for all $s \in \mathcal{S} \setminus \mathcal{S}_{fix}$ and $a \in \mathcal{A}$, $\hat{q}^\pi(s,a) \approx_\omega q^\pi(s,a)$, and that π is next-state Δ-optimal on $\mathcal{S} \setminus \mathcal{S}_{fix}$. Then π' is $(4\omega + \gamma\Delta)$-optimal on $\mathcal{S} \setminus \mathcal{S}_{fix}$.*

3.1 CAPI guarantee with accurate estimation everywhere

To obtain a final suboptimality guarantee for CAPI, first consider the ideal scenario in which we assume that we have a mechanism to estimate $q^\pi(s,a)$ up to some ω accuracy for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, and for any policy π:

Assumption 3.5. *There is an oracle called* ESTIMATE *that accepts a policy π and returns $\hat{q}^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ such that for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$, $\hat{q}^\pi(s,a) \approx_\omega q^\pi(s,a)$.*

Theorem 3.6 (CAPI performance)**.** *Assume* CAPI *(Algorithm 1) is run with $\mathcal{S}_{fix} = \{\}$, iteration count to $I = \lceil \log\omega/\log\gamma \rceil$, and suppose that the estimation used in Line 2 satisfies Assumption 3.5. Then the policy π_I returned by the algorithm is $5\omega/(1-\gamma)$-optimal on $\mathcal{S}$.*

Proof. We prove by induction that policy π_i is Δ_i-optimal on $\mathcal{S}$ for $\Delta_i = 4\omega \sum_{j\in[i]} \gamma^j + \frac{\gamma^i}{1-\gamma}$. This holds immediately for the base case of $i = 0$, as rewards are bounded in $[0,1]$ and thus $v^\star(s) \leq 1/(1-\gamma)$ for any s. Assuming now that the inductive hypothesis holds for $i-1$ we observe that π_{i-1} is next-state Δ-optimal on $\mathcal{S} = \mathcal{S} \setminus \mathcal{S}_{\text{fix}}$. Together with Assumption 3.5, this implies that the conditions of Lemma 3.4 are satisfied for $\pi = \pi_{i-1}$, which yields $v^\star(s) - v^{\pi_i}(s) \leq 4\omega + \gamma\Delta_{i-1} = \Delta_i$, finishing the induction. Finally, by the definition of I, π_I is Δ_I-optimal with $\Delta_I \leq \frac{4\omega}{1-\gamma} + \frac{\gamma^I}{1-\gamma} \leq \frac{5\omega}{1-\gamma}$. □

4 Local access planning with q^π-realizability

Our planner, CAPI-QPI-PLAN, is based on the CONFIDENT MC-LSPI algorithm of Yin et al. [2022]. This latter algorithm gradually builds a *core set* of state-action pairs whose corresponding features are informative. The q-values of the state-action pairs in the core set are estimated using rollouts. The procedure is restarted with an extended core set whenever the algorithm encounters a new informative feature. If such a new feature is not encountered, the estimation error can be controlled, and the estimation is extended to all state-action pairs using the least-squares estimator. Finally, the extended estimation is used in Line 2 of API.

CAPI-QPI-PLAN improves upon CONFIDENT MC-LSPI in two ways. First, using CAPI instead of API improves the final suboptimality bound by a factor of the effective horizon. Second, we

Algorithm 2 MEASURE

1: **Input:** state s, action a, deterministic policy π, set of states $\mathcal{S}' \subseteq \mathcal{S}$, accuracy $\omega > 0$, failure probability $\zeta \in (0, 1]$
2: **Initialize:** $H \leftarrow \lceil \log((\omega/4)(1-\gamma))/\log\gamma \rceil$, $n \leftarrow \left\lceil (\omega/4)^{-2}(1-\gamma)^{-2}\log(2/\zeta)/2 \right\rceil$
3: **for** $i = 1$ to n **do**
4: $\quad (S, R_{i,0}) \leftarrow$ SIMULATOR(s, a)
5: $\quad$ **for** $h = 1$ to $H - 1$ **do**
6: $\quad\quad$ **if** $S \notin \mathcal{S}'$ **then return** (discover, S)
7: $\quad\quad A \leftarrow \pi(S)$
8: $\quad\quad (S, R_{i,h}) \leftarrow$ SIMULATOR(S, A) ▹ Call to the simulator oracle
9: **return** (success, $\frac{1}{n}\sum_{i=1}^{n}\sum_{h=0}^{H-1}\gamma^h R_{i,h}$)

apply a novel analysis on a more modular variant of the CONFIDENTROLLOUT subroutine used in CONFIDENT MC-LSPI, which delivers q-estimation accuracy guarantees with respect to a large class of policies simultaneously. This allows for a dynamically evolving version of policy iteration, that does not have to restart whenever a new informative feature is encountered. Intuitively, this prevents duplication of work.

4.1 Estimation oracle

To obtain an algorithm for planning with local access whose performance degrades gracefully with the uniform approximation error, we must weaken Assumption 3.5. This is because under local access, we cannot guarantee to cover all states or hope to obtain accurate q-value estimates for all states. Instead, we are interested in an accuracy guarantee that holds for q-values only on some subset $\mathcal{S}' \subseteq \mathcal{S}$ of states, but holds simultaneously for *any* policy that agrees with π on $\mathcal{S}'$ but may take arbitrary values elsewhere. For this, we define the extended set of policies:

Definition 4.1. *Let Π_{det} be the set of all stationary deterministic memoryless policies, $\pi \in \Pi_{det}$, and $\mathcal{S}' \subseteq \mathcal{S}$. For $(\pi, \mathcal{S}')$, we define $\Pi_{\pi,\mathcal{S}'}$ to be the set of policies that agree with π on $s \in \mathcal{S}'$:*

$$\Pi_{\pi,\mathcal{S}'} = \{\pi' \in \Pi_{det} \,:\, \pi(s) = \pi'(s) \text{ for all } s \in \mathcal{S}'\} \ .$$

We aim to first accurately estimate $q^\pi(s, a)$ for *some specific* (s, a) pairs, based on which we extend the estimates to other state-action pairs using least-squares. To this end, we first devise a subroutine called MEASURE (Algorithm 2). MEASURE is a modularized variant of the CONFIDENTROLLOUT subroutine of Yin et al. [2022]. The modularity of our variant is due to the parameter $\mathcal{S}'$ that corresponds to the set of states on which the planner is "confident" for CONFIDENTROLLOUT. MEASURE unrolls the policy π starting from (s, a) for a number of episodes, each lasting H steps, and returns with the average measured reward. Throughout, we let $H = \lceil \log((\omega/4)(1-\gamma))/\log\gamma \rceil$ be the effective horizon. At the end of this process, MEASURE returns status *success* along with the empirical average q-value, where compared to Eq. (3), the discounted summation of rewards is truncated to H. If, however, the algorithm encounters a state not in its input $\mathcal{S}'$, it returns with status *discover*, along with that state. This is because in such cases, the algorithm could no longer guarantee an accurate estimation with respect to any member of the extended set of policies. The next lemma, proved in Appendix B, shows that MEASURE provides accurate estimates of the action-value functions for members of the extended policy set.

Lemma 4.2. *For any input parameters $s \in \mathcal{S}, a \in \mathcal{A}, \pi \in \Pi_{det}, \mathcal{S}' \subset \mathcal{S}, \omega > 0, \zeta \in (0, 1)$,* MEASURE *either returns with (discover, s') for some $s' \notin \mathcal{S}'$ (Line 6), or it returns with (success, $\tilde{q}$) such that with probability at least $1 - \zeta$,*

$$q^{\pi'}(s, a) \approx_\omega \tilde{q} \quad \text{for all} \quad \pi' \in \Pi_{\pi,\mathcal{S}'}. \tag{6}$$

Suppose we have a list of state-action pairs $C = (s_i, a_i)_{i\in[|C|]}$ and corresponding q-estimates $\bar{q} = (\bar{q}_i)_{i\in|C|}$. We use the regularized least-squares estimator LSE (Eq. 8) to extend the estimates for all state-action pairs, with regularization parameter $\lambda = \omega^2/B^2$ (recall that B is defined in Definition 1.1):

$$V(C) = \lambda\mathbb{I} + \textstyle\sum_{i\in[|C|]}\varphi(s_i, a_i)\varphi(s_i, a_i)^\top \ , \tag{7}$$

$$\mathrm{LSE}_{C,\bar{q}}(s, a) = \left\langle \varphi(s, a), V(C)^{-1}\textstyle\sum_{i\in[|C|]}\varphi(s_i, a_i)\bar{q}_i \right\rangle \ . \tag{8}$$

For $C = \bar{q} = ()$ (the empty sequence), we define $\mathrm{LSE}_{C,\bar{q}}(\cdot,\cdot) = 0$. This estimator satisfies the guarantee below.

Lemma 4.3. *Let π be a stationary deterministic memoryless policy. Let $C = (s_i, a_i)_{i\in[n]}$ be sequences of state-action pairs of some length $n \in \mathbb{N}$ and $\bar{q} = (\bar{q}_i)_{i\in[n]}$ a sequence of corresponding reals such that for all $i \in [n]$, $q^\pi(s_i, a_i) \approx_\omega \bar{q}_i$. Then, for all $s, a \in \mathcal{S} \times \mathcal{A}$,*

$$\left|LSE_{C,\bar{q}}(s,a) - q^\pi(s,a)\right| \le \varepsilon + \|\varphi(s,a)\|_{V(C)^{-1}} \left(\sqrt{\lambda}B + (\omega+\varepsilon)\sqrt{n}\right), \tag{9}$$

where ε is the uniform approximation error from Definition 1.1.

The proof is given in Appendix C. The order of the estimation accuracy bound (Eq. 9) is optimal, as shown by the lower bounds of Du et al. [2019] and Lattimore et al. [2020].

We intend to use the LSE estimator given in Eq. (8) and the bound in Lemma 4.3 only for state-action pairs where $\|\varphi(s,a)\|_{V(C)^{-1}} \le 1$ (and $n = \tilde{O}(d)$). We call these state-action pairs *covered* by C, and we call a state s covered by C if for all their corresponding actions a, the pair (s,a) is covered by C:

$$\mathrm{ActionCover}(C) = \{(s,a) \in \mathcal{S} \times \mathcal{A} \,:\, \|\varphi(s,a)\|_{V(C)^{-1}} \le 1\} \tag{10}$$

$$\mathrm{Cover}(C) = \{s \in \mathcal{S} \,:\, \forall a \in \mathcal{A},\ (s,a) \in \mathrm{ActionCover}(C)\}\,. \tag{11}$$

We will use the parameter $\mathcal{S}_{\mathrm{fix}}$ of CAPI (see CAPI's update rule in Eq. 5) to ensure policies are only updated on covered states, where the approximation error is well-controlled by Eq. (9).

4.2 Main algorithm

Finally, we are ready to introduce CAPI-Qpi-Plan, presented in Algorithm 3, our algorithm for planning with local access under approximate q^π-realizability. For this, we define levels $l = 0, 1, \ldots, H$, and corresponding suboptimality requirements: For any $l \in [H+1]$, let

$$\Delta_l = 8(\varepsilon+\omega)\left(\sqrt{\tilde{d}}+1\right)\sum_{j\in[l]}\gamma^j + \frac{\gamma^l}{1-\gamma}\,,$$

for some $\tilde{d} = \tilde{\Theta}(d)$ defined in Eq. (13). For each level l, the algorithm maintains a policy π_l and a set of covered states on which it can guarantee that π_l is a Δ_l-optimal policy. More specifically, this set is $\mathrm{Cover}(C_l)$, where C_l is a list of state-action pairs with elements $C_{l,i} = (s_l^i, a_l^i)$ for $i \in [|C_l|]$. The algorithm maintains the following suboptimality guarantee below, which we prove in Appendix E after showing some further key properties of the algorithm.

Lemma 4.4. *Assuming that Eq.* (6) *holds whenever* Measure *returns* success*, π_l is Δ_l-optimal on Cover(C_l) (Definition 3.2) for all $l \in [H+1]$ at the end of every iteration of the main loop of* CAPI-Qpi-Plan.

CAPI-Qpi-Plan aims to improve the policies, while *propagating* the members of C_l to C_{l+1}, and so on, all the way to C_H. During this, whenever the algorithm discovers a state-action pair with a sufficiently "new" feature direction, this pair is appended to the sequence C_0 corresponding to level 0, as there are no suboptimality guarantees yet available for such a state. However, such a discovery can only happen $\tilde{O}(d)$ times. When, eventually, all discovered state-action pairs end up in C_H, the final suboptimality guarantee is reached, and the algorithm returns with the final policy. Note that in the local access setting we consider, the algorithm cannot enumerate the set $\mathrm{Cover}(C_l)$, but can answer membership queries, that is, for any $s \in \mathcal{S}$ it encounters, it is able to decide if $s \in \mathrm{Cover}(C_l)$. The algorithm maintains sequences $\bar{q}_l$, corresponding to C_l, for each level l. Whenever a new (s,a) pair is appended to the sequence C_l, a corresponding $\perp$ symbol is appended to the sequence $\bar{q}_l$, to signal that an estimate of $q^{\pi_l}(s,a)$ is not yet known.

After initializing C_0 to cover the initial state s_0 (Lines 4 to 6), the algorithm measures $q^{\pi_\ell}(s,a)$ for the smallest level ℓ for which there still exists a $\perp$ in the corresponding $\bar{q}_\ell$. After a successful measurement, if there are no more $\perp$'s left at this level (i.e., in $\bar{q}_\ell$), the algorithm executes a policy update on π_ℓ (Line 17) using the least-squares estimate obtained from the measurements at this level, but only for states in $\mathrm{Cover}(C_\ell)$ (using $\mathcal{S}_{\mathrm{fix}} = \mathcal{S} \setminus \mathrm{Cover}(C_\ell)$). Next, Line 18 merges this new policy π' with the existing policy $\pi_{\ell+1}$ of the next level, setting $\pi_{\ell+1}$ to be the policy π'' defined as

$$\pi''(s) = \begin{cases} \pi_{\ell+1}(s), & \text{if } s \in \mathrm{Cover}(C_{\ell+1});\\ \pi'(s), & \text{otherwise.}\end{cases}$$

Algorithm 3 CAPI-QPI-PLAN

1: **Input:** initial state $s_0 \in \mathcal{S}$, dimensionality d, parameter bound B, accuracy ω, failure probability $\delta > 0$
2: **Initialize:** $H \leftarrow \lceil \log((\omega/4)(1-\gamma))/\log\gamma \rceil$, for $l \in [H+1]$, $C_l \leftarrow ()$, $\bar{q}_l \leftarrow ()$, $\pi_l \leftarrow$ policy that always returns action $\mathcal{A}_1$, $\lambda \leftarrow \omega^2/B^2$
3: **while** True **do** ▷ main loop
4: **if** $\exists a \in \mathcal{A},\ (s_0, a) \notin \text{ActionCover}(C_0)$ **then**
5: append (s_0, a) to C_0, append $\perp$ to $\bar{q}_0$
6: **break**
7: let ℓ be the smallest integer such that $\bar{q}_{\ell,m} = \perp$ for some m; set $\ell = H$ if no such l exists
8: **if** $\ell = H$ **then return** π_H
9: $(\text{status}, \text{result}) \leftarrow \textsc{Measure}(s^m_\ell, a^m_\ell, \pi_\ell, \text{Cover}(C_\ell), \omega, \delta/(\tilde{d}H))$ ▷ recall $C_{\ell,m} = (s^m_\ell, a^m_\ell)$
10: **if** status = discover **then**
11: append (result, a) to C_0 for some a such that $(\text{result}, a) \notin \text{ActionCover}(C_0)$
12: append $\perp$ to $\bar{q}_0$
13: **break**
14: $\bar{q}_{\ell,m} \leftarrow \text{result}$
15: **if** $\nexists m'$ such that $\bar{q}_{\ell,m'} = \perp$ **then**
16: $\hat{q} \leftarrow \text{LSE}_{C_\ell, \bar{q}_\ell}$
17: $\pi' \leftarrow \pi_{\hat{q}, \pi_\ell, \mathcal{S} \setminus \text{Cover}(C_\ell)}$
18: $\pi_{\ell+1} \leftarrow (s \mapsto \pi_{\ell+1}(s) \text{ if } s \in \text{Cover}(C_{\ell+1}) \text{ else } \pi'(s))$
19: **for** $(s, a) \in C_\ell$ such that $(s, a) \notin C_{\ell+1}$ **do**
20: append (s, a) to $C_{\ell+1}$, $\perp$ to $\bar{q}_{\ell+1}$

This ensures that the existing policy $\pi_{\ell+1}$ remains unchanged by π'' (its replacement) on states that are already covered by $C_{\ell+1}$, and therefore $\pi'' \in \Pi_{\pi_{\ell+1}, \text{Cover}(C_{\ell+1})} = \Pi_{\pi'', \text{Cover}(C_{\ell+1})}$. We also observe that C_l can only grow for any l (elements are never removed from these sequences), thus for any update where C_l is assigned a new value C'_l (Lines 5, 11, and 20), $V(C'_l) \geq V(C_l)$, and therefore $\text{Cover}(C'_l) \supseteq \text{Cover}(C_l)$ and $\Pi_{\pi_l, \text{Cover}(C'_l)} \subseteq \Pi_{\pi_l, \text{Cover}(C_l)}$. Combining these properties yields the following result:

Lemma 4.5. *If for any $l \in [H]$, π_l and C_l take some values π^{old}_l and C^{old}_l at any point in the execution of the algorithm, then at any later point during the execution, $\pi_l \in \Pi_{\pi_l, Cover(C_l)} \subseteq \Pi_{\pi^{old}_l, Cover(C^{old}_l)}$.*

Any value in $\bar{q}_l$ that is set to anything other than $\perp$ will never change again. Since as long as the sample paths generated by MEASURE in Line 9 of CAPI-QPI-PLAN remain in $\text{Cover}(C_l)$, their distribution is the same under any policy from $\Pi_{\pi_l, \text{Cover}(C_l)}$, the $\bar{q}_l$ estimates are valid for these policies, as well. Combined with Lemma 4.5, we get that the accuracy guarantees of Lemma 4.2 continue to hold throughout:

Lemma 4.6. *Assuming that Eq.* (6) *holds whenever* MEASURE *returns* success*, for any level l and index m such that $\bar{q}_{l,m} \neq \perp$, $q^{\pi'}(s^m_l, a^m_l) \approx_\omega \bar{q}_{l,m}$ for all $\pi' \in \Pi_{\pi_l, Cover(C_l)}$ throughout the execution of* CAPI-QPI-PLAN.

Once $\pi_{\ell+1}$ is updated in Line 18, in Line 20 we append to the sequence $C_{\ell+1}$ all members of C_ℓ that are not yet in $C_{\ell+1}$, while adding a corresponding $\perp$ to $\bar{q}_{\ell+1}$ indicating that these q-values are not yet measured for policy $\pi_{\ell+1}$. Thus, whenever all $\perp$ values disappear from some level $l \in [H+1]$, by the end of that iteration $C_{l+1} = C_l$, and hence $\text{ActionCover}(C_l) = \text{ActionCover}(C_{l+1})$. Together with the fact that for any $l \in [H+1]$, whenever a new state-action pair is appended to C_l, an $\perp$ symbol is appended to $\bar{q}_l$, we have by induction the following result:

Lemma 4.7. *Throughout the execution of* CAPI-QPI-PLAN*, after Line 7 when ℓ is set,*

$$ActionCover(C_0) = ActionCover(C_1) = \cdots = ActionCover(C_\ell)\,.$$

As a result, whenever the MEASURE call of Line 9 outputs $(\text{discover}, s)$ for some state s, by Lemma 4.2, there is an action $a \in \mathcal{A}$ such that $(s, a) \notin \text{ActionCover}(C_\ell) = \text{ActionCover}(C_0)$. This explains why adding such an (s, a) pair to C_0 is always possible in Line 11. Consider the ith time Line 11 is executed, and denote s by s_i and a by a_i, and $V_i = \lambda\mathbb{I} + \sum_{t=1}^{i-1} \varphi(s_t, a_t)\varphi(s_t, a_t)^\top$. Observe that as $V_i = V(C)$, $(s_i, a_i) \notin \text{ActionCover}(C_0)$ implies $\|\varphi(s_i, a_i)\|_{V_i^{-1}} > 1$. Therefore,

$\sum_{t=1}^{i} \min\{1, \|\varphi(s_t, a_t)\|_{V_t^{-1}}\} = i$, and thus by the elliptical potential lemma [Lattimore and Szepesvári, 2020, Lemma 19.4], $i \leq 2d \log\left(\frac{d\lambda + iL^2}{d\lambda}\right)$. This inequality is satisfied by the largest value of i, that is, the total number of times MEASURE returns with *discover*. Since any element of C_l is also an element of C_0 for any $l \in [H+1]$, we have that at any time during the execution of CAPI-QPI-PLAN,

$$|C_l| \leq 4d \log\left(1 + \frac{4L^2}{\lambda}\right) =: \tilde{d} = \tilde{O}(d) \,. \tag{13}$$

When CAPI-QPI-PLAN returns at Line 8 with the policy π_H, it is Δ_H-optimal on $\mathrm{Cover}(C_H)$ by Lemma 4.4 when the estimates of MEASURE are correct. Furthermore, $s_0 \in \mathrm{Cover}(C_0)$ is guaranteed by Lines 4 to 6, and hence $s_0 \in \mathrm{Cover}(C_H)$ by Lemma 4.7 when the algorithm finishes. Hence, bounding Δ_H using the definition of H immediately gives the following result:

Lemma 4.8. *Assuming that Eq.* (6) *holds whenever* MEASURE *returns* success*, the policy π returned by* CAPI-QPI-PLAN *is Δ-optimal on $\{s_0\}$ for*

$$\Delta = 9(\varepsilon + \omega)\left(\sqrt{\tilde{d}} + 1\right)(1-\gamma)^{-1} = \tilde{O}\left((\varepsilon + \omega)\sqrt{d}(1-\gamma)^{-1}\right) \,.$$

To finish the proof of Theorem 1.2, we only need to analyze the query complexity and the failure probability (i.e., the probability of Eq. (6) not being satisfied for some MEASURE call that returns *success*) of CAPI-QPI-PLAN:

Proof of Theorem 1.2. Both the total failure probability and query complexity of CAPI-QPI-PLAN depend on the number of times MEASURE is executed, as this is the only source of randomness and of interaction with the simulator. MEASURE can return *discover* at most $|C_0|$ times, which is bounded by $\tilde{d}$ by Eq. (13). For every $l \in [H]$, MEASURE is executed exactly once with returning *success* for each element of C_l. Hence, by Eq. (13) again, MEASURE returns *success* at most $\tilde{d}H$ times, each satisfying Eq. (6) with probability at least $1 - \zeta = 1 - \delta/(\tilde{d}H)$ by Lemma 4.2. By the union bound, MEASURE returns *success* in all occasions with probability at least $1 - \delta$. Hence Eq. (6) holds with probability at least $1 - \delta$, which, combined with Lemma 4.8, proves Eq. (1).

Each successful run of MEASURE executes at most nH queries (n is set in Line 2 of Algorithm 2). Since $H < (1-\gamma)^{-1}\log(4\omega^{-1}(1-\gamma)^{-1}) = \tilde{O}((1-\gamma)^{-1})$, in total CAPI-QPI-PLAN executes at most $\tilde{O}\left(d(1-\gamma)^{-4}\omega^{-2}\right)$ queries. As this happens at most $\tilde{d}H$ times, we obtain the desired bound on the query complexity. □

5 Conclusions and future work

In this paper we presented CONFIDENT APPROXIMATE POLICY ITERATION, a confident version of API, which can obtain a stationary policy with a suboptimality guarantee that scales linearly with the effective horizon $H = \tilde{O}(1/(1-\gamma))$. This scaling is optimal as shown by Scherrer and Lesner [2012].

CAPI can be applied to local planning with approximate q^π-realizability (yielding the CAPI-QPI-PLAN algorithm) to obtain a sequence of policies with successively refined accuracies on a dynamically evolving set of states, resulting in a final, recursively defined policy achieving simultaneously the optimal suboptimality guarantee and best query cost available in the literature. More precisely, CAPI-QPI-PLAN achieves $\tilde{O}(\varepsilon\sqrt{d}H)$ suboptimality, where ε is the uniform policy value-function approximation error. We showed that this bound is the best (up to polylogarithmic factors) that is achievable by any planner with polynomial query cost. We also proved that the $\tilde{O}\left(dH^4\varepsilon^{-2}\right)$ query cost of CAPI-QPI-PLAN is optimal up to polylogarithmic factors in all parameters except for H; whether the dependence on H is optimal remains an open question.

Finally, our method comes at a memory and computational cost overhead, both for the final policy and the planner. It is an interesting question if this overhead necessarily comes with the API-style method we use (as it is also present in the works of Scherrer and Lesner, 2012, Scherrer, 2014), or if it is possible to reduce it by, for example, compressing the final policy into one that is greedy with respect to some action-value function realized with the features.

Acknowledgements

The authors would like to thank Tor Lattimore and Qinghua Liu for helpful discussions.

References

Y. Abbasi-Yadkori, P. Bartlett, K. Bhatia, N. Lazic, Cs. Szepesvári, and G. Weisz. Politex: Regret bounds for policy iteration using expert prediction. In *ICML*, pages 3692–3702, 2019.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control: Approximate dynamic programming*, volume II. 4 edition, 2012.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2019.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

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

Tor Lattimore, Csaba Szepesvári, and Gellért Weisz. Learning with good feature representations in bandits and in RL with a generative model. In *ICML*, pages 9464–9472, 2020.

A Woodbury Max. Inverting modified matrices. In *Memorandum Rept. 42, Statistical Research Group*, page 4. Princeton Univ., 1950.

Remi Munos. Error bounds for approximate policy iteration. In *ICML*, pages 560–567, 2003.

Remi Munos. Error bounds for approximate value iteration. In *AAAI*, pages 1006–1011, 2005.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994.

Walter Rudin et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1976.

Daniel Russo. Approximation benefits of policy gradient methods with aggregated states. *arXiv preprint arXiv:2007.11684*, 2020.

Bruno Scherrer. Approximate policy iteration schemes: a comparison. In *International Conference on Machine Learning*, pages 1314–1322. PMLR, 2014.

Bruno Scherrer and Boris Lesner. On the use of non-stationary policies for stationary infinite-horizon Markov decision processes. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, 2012.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Andrew Wagenmaker, Yifang Chen, Max Simchowitz, Simon S Du, and Kevin Jamieson. Reward-free RL is no harder than reward-aware RL in linear Markov decision processes. *arXiv preprint arXiv:2201.11206*, 2022.

Yuanhao Wang, Ruosong Wang, and Sham Kakade. An exponential lower bound for linearly realizable MDP with constant suboptimality gap. *Advances in Neural Information Processing Systems*, 34, 2021.

Gellért Weisz, Philip Amortila, and Csaba Szepesvári. Exponential lower bounds for planning in MDPs with linearly-realizable optimal action-value functions. In *ALT*, volume 132 of *Proceedings of Machine Learning Research*, pages 1237–1264, 2021.

Chenjun Xiao, Ilbin Lee, Bo Dai, Dale Schuurmans, and Csaba Szepesvari. The curse of passive data collection in batch reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 8413–8438, 2022.

Tengyang Xie and Nan Jiang. Batch value-function approximation with only realizability, 2020.

Dong Yin, Botao Hao, Yasin Abbasi-Yadkori, Nevena Lazić, and Csaba Szepesvári. Efficient local planning with linear function approximation. In *International Conference on Algorithmic Learning Theory*, pages 1165–1192. PMLR, 2022.

Andrea Zanette, Alessandro Lazaric, Mykel Kochenderfer, and Emma Brunskill. Learning near optimal policies with low inherent Bellman error. In *International Conference on Machine Learning*, pages 10978–10989. PMLR, 2020.

Dongruo Zhou, Jiafan He, and Quanquan Gu. Provably efficient reinforcement learning for discounted MDPs with feature mapping. *arXiv preprint arXiv:2006.13165*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proof of Lemma 3.4

Take any $s \in \mathcal{S} \setminus \mathcal{S}_{\text{fix}}$.

$$\begin{aligned} v^\star(s) - v^{\pi'}(s) &= v^\star(s) - q^{\pi'}(s, \pi'(s)) \\ &= v^\star(s) - q^\pi(s, \pi'(s)) + q^\pi(s, \pi'(s)) - q^{\pi'}(s, \pi'(s)) \\ &\leq v^\star(s) - q^\pi(s, \pi'(s)) \,, \end{aligned} \tag{14}$$

where the first equality holds because π' is deterministic, and the inequality is true because

$$q^\pi(s, \pi'(s)) - q^{\pi'}(s, \pi'(s)) = \gamma \int_{s' \in \mathcal{S}} \left(v^\pi(s') - v^{\pi'}(s') \right) \mathrm{d}P(s'|s, \pi'(s)) \leq 0$$

by Lemma 3.1. Next observe that

$$\hat{q}^\pi(s, \pi'(s)) \geq \max_{a \in \mathcal{A}} \hat{q}^\pi(s, a) - 2\omega \tag{15}$$

since, as $s \notin \mathcal{S}_{\text{fix}}$, either $\pi'(s)$ is defined by Case 5a as $\pi'(s) = \arg\max_{a \in \mathcal{A}} \hat{q}^\pi(s, a)$ and so $\hat{q}^\pi(s, \pi'(s)) = \max_{a \in \mathcal{A}} \hat{q}^\pi(s, a)$, or it is defined by Case 5b in which case $\hat{q}^\pi(s, \pi'(s)) = \hat{q}^\pi(s, \pi(s)) \geq \max_{a \in \mathcal{A}} \hat{q}^\pi(s, a) - 2\omega$. Combining Eqs. (14) and (15), we obtain

$$\begin{aligned} v^\star(s) - v^{\pi'}(s) &\leq v^\star(s) - \hat{q}^\pi(s, \pi'(s)) + \hat{q}^\pi(s, \pi'(s)) - q^\pi(s, \pi'(s)) \\ &\leq v^\star(s) - \hat{q}^\pi(s, \pi'(s)) + \omega \\ &\leq v^\star(s) - \max_{a \in \mathcal{A}} \hat{q}^\pi(s, a) + 3\omega \,, \end{aligned}$$

where in the first line we added and subtracted $\hat{q}^\pi(s, \pi'(s))$, and the second inequality holds as $\hat{q}^\pi(s, a) \approx_\omega q^\pi(s, a)$ for $s \notin \mathcal{S}_{\text{fix}}$ and $a \in \mathcal{A}$ by the assumptions of the lemma.

We continue by adding and subtracting $\max_{a \in \mathcal{A}} q^\pi(s, a)$:

$$\begin{aligned} v^\star(s) - v^{\pi'}(s) &\leq v^\star(s) - \max_{a \in \mathcal{A}} q^\pi(s, a) + \max_{a \in \mathcal{A}} q^\pi(s, a) - \max_{a \in \mathcal{A}} \hat{q}^\pi(s, a) + 3\omega \\ &\leq v^\star(s) - \max_{a \in \mathcal{A}} q^\pi(s, a) + 4\omega \\ &= \max_{a \in \mathcal{A}} \left[r(s, a) + \gamma \int_{s' \in \mathcal{S}} v^\star(s') \, \mathrm{d}P(s'|s, a) \right] \\ &\qquad - \max_{a \in \mathcal{A}} \left[r(s, a) + \gamma \int_{s' \in \mathcal{S}} v^\pi(s') \, \mathrm{d}P(s'|s, a) \right] + 4\omega \\ &\leq \max_{a \in \mathcal{A}} \left[\gamma \int_{s' \in \mathcal{S}} \left(v^\star(s') - v^\pi(s') \right) \mathrm{d}P(s'|s, a) \right] + 4\omega \\ &\leq 4\omega + \gamma\Delta \,, \end{aligned}$$

where in the fifth line we used that π is next-state Δ-optimal by assumption. □

B Proof of Lemma 4.2

For an episode trajectory $\{S_h, A_h, R_h\}_{h \in \mathbb{N}}$, let K be the smallest positive integer such that $S_K \notin \mathcal{S}'$. For any $i \in \{1, \ldots, n\}$, let I_i denote the indicator of the event that at the ith iteration of the outer loop of Algorithm 2, the algorithm encounters $S \notin \mathcal{S}'$ in Line 6. Note that $\mathbb{E}_{\pi,s,a}[I_i] = \mathcal{P}_{\pi,s,a}[1 \leq K < H]$. Then, by Hoeffding's inequality (see, e.g., Lattimore and Szepesvári [2020]), with probability at least $1 - \zeta/2$,

$$\left| \mathcal{P}_{\pi,s,a}\left[1 \leq K < H\right] - \frac{1}{n} \sum_{i=1}^{n} I_i \right| \leq \frac{\omega(1-\gamma)}{4} \,.$$

MEASURE only returns *success* if all indicators are zero; therefore, the above inequality implies that if MEASURE returns *success* then, with probability at least $1 - \zeta/2$, we have

$$\mathcal{P}_{\pi,s,a}\left[1 \leq K < H\right] \leq \frac{\omega(1-\gamma)}{4} \,. \tag{16}$$

Recall that if MEASURE returns (success, $\tilde{q}$), then $\bar{q} = \frac{1}{n}\sum_{i=1}^{n}\sum_{h=0}^{H-1}\gamma^h R_{i,h}$. Since

$$0 \le q^\pi(s,a) - \mathbb{E}_{\pi,s,a}\sum_{h=0}^{H-1}\gamma^h R_h = \mathbb{E}_{\pi,s,a}\sum_{h=H}^{\infty}\gamma^h R_h \le \frac{\gamma^H}{1-\gamma} \le \frac{\omega}{4}\,,$$

another application of Hoeffding's inequality yields that $q^\pi(s,a)$ and $\bar{q}$ are close with high probability: with probability at least $1-\zeta/2$,

$$\begin{aligned}|q^\pi(s,a) - \bar{q}| &= \left|q^\pi(s,a) - \frac{1}{n}\sum_{i=1}^{n}\sum_{h=0}^{H-1}\gamma^h R_{i,h}\right| \\ &\le \omega/4 + \left|\mathbb{E}_{\pi,s,a}\sum_{h=0}^{H-1}\gamma^h R_h - \frac{1}{n}\sum_{i=1}^{n}\sum_{h=0}^{H-1}\gamma^h R_{i,h}\right| \le \omega/2\,, \end{aligned} \tag{17}$$

where we also used that the range of the sum of the rewards above for every i is $[0, 1/(1-\gamma)]$.

Pick any $\pi' \in \Pi_{\pi,\mathcal{S}'}$. Observe that for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, the distribution of the trajectory $S_0, A_0, R_0, S_1, A_1, R_1, \ldots, A_{K-1}, R_{K-1}, S_K$ is the same under $\mathcal{P}_{\pi',s,a}$ and $\mathcal{P}_{\pi,s,a}$, as π and π' select the same actions for states in $\mathcal{S}'$. By Eqs. (3) to (4), we can write

$$\begin{aligned}\left|q^{\pi'}(s,a) - q^\pi(s,a)\right| &= \left|\mathbb{E}_{\pi',s,a}\left[\sum_{t\in[K]}\gamma^t R_t + \gamma^K v^{\pi'}(S_K)\right] - \mathbb{E}_{\pi,s,a}\left[\sum_{t\in[K]}\gamma^t R_t + \gamma^K v^{\pi'}(S_K)\right]\right| \\ &= \left|\mathbb{E}_{\pi,s,a}\left[\gamma^K\left(v^{\pi'}(S_K) - v^\pi(S_K)\right)\right]\right| \le \frac{1}{1-\gamma}\mathbb{E}_{\pi,s,a}\left[\gamma^K\right] \\ &\le \frac{1}{1-\gamma}\mathcal{P}_{\pi,s,a}\left[1 \le K < H\right] + \frac{\gamma^H}{1-\gamma} \le \frac{1}{1-\gamma}\mathcal{P}_{\pi,s,a}\left[1 \le K < H\right] + \omega/4\,. \end{aligned} \tag{18}$$

Combining Eqs. (16) to (18), it follows by the union bound that if MEASURE returns with (success, $\tilde{q}$), then with probability at least $1-\zeta$,

$$\left|q^{\pi'}(s,a) - \bar{q}\right| \le \left|q^{\pi'}(s,a) - q^\pi(s,a)\right| + |q^\pi(s,a) - \bar{q}| \le \omega\,. \qquad \square$$

C Proof of Lemma 4.3

We start the proof by showing that there exists a $\theta \in \mathbb{R}^d$ such that

$$\|\theta\|_2 \le B \text{ and for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A},\ q^\pi(s,a) \approx_\varepsilon \langle\theta, \varphi(s,a)\rangle\,. \tag{19}$$

For any finite set $W \subseteq \mathcal{S}\times\mathcal{A}$, $\max_{(s,a)\in W}|q^\pi(s,a) - \langle\varphi(s,a),\theta'\rangle|$ is a continuous function of θ', hence it attains its infimum on the compact set $\{\theta' \in \mathbb{R}^d : \|\theta'\|_2 \le B\}$. By Definition 1.1, this infimum is at most ε. Therefore, the compact sets $\Theta_{s,a} = \{\theta' \in \mathbb{R}^d : \|\theta'\|_2 \le B \text{ and } |q^\pi(s,a) - \langle\varphi(s,a),\theta'\rangle| \le \varepsilon\}$ are non-empty for all $(s,a) \in \mathcal{S}\times\mathcal{A}$, and any intersection of a finite collection of these sets is also non-empty. Therefore, $\bigcap_{(s,a)\in\mathcal{S}\times\mathcal{A}}\Theta_{s,a}$ is non-empty by [Rudin et al., 1976, Theorem 2.36], and any element θ of this set satisfies Eq. (19). For the remainder of this proof, fix such a θ.

For any $i \in [n]$, with a slight abuse of notation, we introduce the shorthand $\varphi_i = \varphi(s_i, a_i)$, and let $\hat{q}_i = \langle\theta,\varphi_i\rangle$ and $\xi_i = \bar{q}_i - \hat{q}_i$. Note that by the triangle inequality, $|\xi_i| \le |\bar{q}_i - q^\pi(s_i,a_i)| + |q^\pi(s_i,a_i) - \hat{q}_i| \le \omega + \varepsilon$. Let $\bar{\theta} = V(C)^{-1}\sum_{i\in[n]}\varphi_i\bar{q}_i$ and $\hat{\theta} = V(C)^{-1}\sum_{i\in[n]}\varphi_i\hat{q}_i$.

For any $v \in \mathbb{R}^d$ by the Cauchy-Schwarz inequality,

$$\left|\left\langle\bar{\theta} - \theta, v\right\rangle\right| \le \left|\left\langle\hat{\theta} - \theta, v\right\rangle\right| + \left|\left\langle\bar{\theta} - \hat{\theta}, v\right\rangle\right| \le \|v\|_{V(C)^{-1}}\left\|\hat{\theta} - \theta\right\|_{V(C)} + \left|\left\langle V(C)^{-1}\sum_{i\in[n]}\varphi_i\xi_i, v\right\rangle\right|\,.$$

To bound the first term on the right-hand side above, observe that

$$\left\|\hat{\theta} - \theta\right\|_{V(C)} = \left\|V(C)^{-1}\left(\sum_{i\in[n]}\varphi_i{\varphi_i}^\top\right)\theta - \theta\right\|_{V(C)} = \lambda\,\|\theta\|_{V(C)^{-1}} \le \lambda\,\|\theta\|_{\frac{1}{\lambda}\mathbb{I}} \le \sqrt{\lambda}B\,,$$

where in the last line we used that $V(C) \succeq \lambda \mathbb{I}$.

The second term can be bounded as

$$\begin{aligned}
\left|\left\langle V(C)^{-1} \sum_{i\in[n]} \varphi_i \xi_i, v \right\rangle\right| &\leq \sum_{i\in[n]} \left|\left\langle V(C)^{-1}\varphi_i \xi_i, v\right\rangle\right| \\
&\leq (\omega+\varepsilon) \sum_{i\in[n]} \left|\left\langle V(C)^{-1}\varphi_i, v\right\rangle\right| \\
&\leq (\omega+\varepsilon)\sqrt{n}\sqrt{\sum_{i\in[n]} \left(\left\langle V(C)^{-1}\varphi_i, v\right\rangle\right)^2} \\
&\leq (\omega+\varepsilon)\sqrt{n}\sqrt{v^\top V(C)^{-1}\left(\sum_{i\in[n]} \varphi_i \varphi_i{}^\top\right) V(C)^{-1} v + v^\top V(C)^{-1} \lambda \mathbb{I} V(C)^{-1} v} \\
&= (\omega+\varepsilon)\sqrt{n}\, \|v\|_{V(C)^{-1}} ,
\end{aligned}$$

where the first inequality holds by the triangle inequality, the second by our bound on $|\xi_i|$, the third by the Cauchy-Schwartz inequality, and the fourth by the positivity of λ. Putting it all together, for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, using the previous bounds with $v = \varphi(s,a)$,

$$\begin{aligned}
\left|\mathrm{LSE}_{C,\bar{q}}(s,a) - q^\pi(s,a)\right| &\leq |q^\pi(s,a) - \langle \theta, \varphi(s,a)\rangle| + \left|\left\langle \bar{\theta} - \theta, \varphi(s,a)\right\rangle\right| \\
&\leq \varepsilon + \|\varphi(s,a)\|_{V(C)^{-1}} \left(\sqrt{\lambda} B + (\omega+\varepsilon)\sqrt{n}\right) ,
\end{aligned}$$

completing the proof. □

D Deriving next-state optimality of π_ℓ for Lemma 4.4

Lemma D.1. *Assume that Eq.* (6) *holds whenever* MEASURE *returns* success. *At any point of* CAPI-QPI-PLAN *after Line 16 is executed, for any* $\pi'' \in \Pi_{\pi_\ell, Cover(C_\ell)}$, $s \in Cover(C_\ell)$, *and* $a \in \mathcal{A}$,

$$\left|\hat{q}(s,a) - q^{\pi''}(s,a)\right| \leq (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) .$$

Proof. By Lemma 4.6 and Eq. (6), $\bar{q}_{l,m} \approx_\omega q^{\pi''}(C_{l,m})$ for all $m \in [|C_\ell|]$ (recall that $C_{l,m}$ is the mth state-action pair in C_l). Therefore, applying Lemma 4.3 with $q^{\pi''}$, C_ℓ and $\bar{q}_\ell$, as $\hat{q} = \mathrm{LSE}_{C_\ell,\bar{q}_\ell}$, we get that for any $s \in \mathrm{Cover}(C_\ell)$ and all $a \in \mathcal{A}$,

$$\begin{aligned}
\left|\hat{q}(s,a) - q^{\pi''}(s,a)\right| &\leq \varepsilon + \|\varphi(s,a)\|_{V(C_\ell)^{-1}} \left(\sqrt{\lambda} B + (\omega+\varepsilon)\sqrt{|C_\ell|}\right) \\
&\leq (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) ,
\end{aligned}$$

where the second inequality holds because $\|\varphi(s,a)\|_{V(C_\ell)^{-1}} \leq 1$ since $s \in \mathrm{Cover}(C_\ell)$, $|C_\ell| \leq \tilde{d}$ by Eq. (13), and the definition of λ. □

Lemma D.2. *Assume that Eq.* (6) *holds whenever* MEASURE *returns* success. *Consider a time when Lines 17 to 20 of* CAPI-QPI-PLAN *are run and assume that at this time, for all* $l \in [H+1]$, π_l *is* Δ_l*-optimal on* $Cover(C_l)$. *Then,* π_ℓ *is next-state* $(\Delta_\ell + 4(\omega+\varepsilon)(\sqrt{\tilde{d}}+1)/\gamma)$*-optimal on* $Cover(C_\ell)$.

Proof. Let π_ℓ^+ be defined as in Eq. (22). As $\pi_\ell^+ \in \Pi_{\pi_\ell, \mathrm{Cover}(C_\ell)}$, by Lemma D.1, for any $s \in \mathrm{Cover}(C_\ell)$ and all $a \in \mathcal{A}$,

$$\left|\hat{q}(s,a) - q^{\pi_\ell^+}(s,a)\right| \leq (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) .$$

Similarly, applying Lemma D.1 with π_ℓ (which trivially belongs to $\Pi_{\pi_\ell, \mathrm{Cover}(C_\ell)}$), we also have

$$|\hat{q}(s,a) - q^{\pi_\ell}(s,a)| \leq (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) .$$

Therefore,

$$\left|q^{\pi_\ell^+}(s,a) - q^{\pi_\ell}(s,a)\right| \le 2(\omega+\varepsilon)(\sqrt{\tilde{d}}+1). \tag{20}$$

Since π_ℓ is Δ_ℓ-optimal on $\mathrm{Cover}(C_\ell)$ by assumption, this makes π_ℓ^+ Δ-optimal on $\mathrm{Cover}(C_\ell)$ for

$$\Delta = \Delta_\ell + 2(\omega+\varepsilon)(\sqrt{\tilde{d}}+1). \tag{21}$$

For a trajectory in the MDP, let the random variable τ be the first time the state is in $\mathrm{Cover}(C_\ell)$:

$$\tau = \min\{t \in \mathbb{N} \mid S_t \in \mathrm{Cover}(C_\ell)\}.$$

Since π_ℓ^+ agrees with $\pi^\star$ on states not in $\mathrm{Cover}(C_\ell)$, the distribution of the trajectory up to and including S_τ is the same under both policies, starting from any state $s \in \mathcal{S}$. Therefore, for any $s \in \mathcal{S}$,

$$\begin{aligned} v^\star(s) - v^{\pi_\ell^+}(s) &= \mathbb{E}_{\pi^\star,s}\left[\sum_{t\in\mathbb{N}} \gamma^t R_t\right] - \mathbb{E}_{\pi_\ell^+,s}\left[\sum_{t\in\mathbb{N}} \gamma^t R_t\right] \\ &= \mathbb{E}_{\pi_\ell^+,s}\left[\gamma^\tau\left(v^\star(S_\tau) - v^{\pi_\ell^+}(S_\tau)\right)\right] \\ &\le \Delta, \end{aligned}$$

as $\gamma^\tau \le 1$ and π_ℓ^+ is Δ-optimal on $\mathrm{Cover}(C_\ell)$. That is, π_ℓ^+ is also Δ-optimal on $\mathcal{S}$ (with Δ defined in Eq. 21). Using this, for any $s \in \mathrm{Cover}(C_\ell)$, and $a \in \mathcal{A}$, we have

$$\begin{aligned} &\int_{s'\in\mathcal{S}} \left(v^\star(s') - v^{\pi_\ell}(s')\right) \mathrm{d}P(s'|s,a) \\ &\le \int_{s'\in\mathcal{S}} \left(v^\star(s') - v^{\pi_\ell^+}(s')\right) \mathrm{d}P(s'|s,a) + \left|\int_{s'\in\mathcal{S}} \left(v^{\pi_\ell^+}(s') - v^{\pi_\ell}(s')\right) \mathrm{d}P(s'|s,a)\right| \\ &\le \Delta_\ell + 2(\omega+\varepsilon)(\sqrt{\tilde{d}}+1) + \frac{1}{\gamma}\left|q^{\pi_\ell^+}(s,a) - q^{\pi_\ell}(s,a)\right| \\ &\le \Delta_\ell + 2(\omega+\varepsilon)(\sqrt{\tilde{d}}+1) + 2(\omega+\varepsilon)(\sqrt{\tilde{d}}+1)/\gamma \\ &= \Delta_\ell + 4(\omega+\varepsilon)(\sqrt{\tilde{d}}+1)/\gamma, \end{aligned}$$

where the third inequality holds by Eq. (20). Therefore π_ℓ is next-state $(\Delta_\ell + 4(\omega+\varepsilon)(\sqrt{\tilde{d}}+1)/\gamma))$-optimal on $\mathrm{Cover}(C_\ell)$. □

E Poof of Lemma 4.4

Proof of Lemma 4.4. We prove by induction on the iterations of the main loop of CAPI-QPI-PLAN the *inductive hypothesis:* at the start of iteration i, for all $l \in [H+1]$, π_l is Δ_l-optimal on $\mathrm{Cover}(C_l)$. We first observe that after initialization, C_l is the empty sequence for every l, so we can apply Lemma 4.3 with $q^\star$ and empty sequences ($n = 0$) to get that for any $s \in \mathrm{Cover}(())$ and $a \in \mathcal{A}$, $q^\star(s,a) \le \varepsilon + \sqrt{\lambda}B = \varepsilon + \omega$. Then, $v^\star(s) \le \varepsilon + \omega \le \Delta_l$. Therefore, at initialization, any policy is Δ_l-optimal on $\mathrm{Cover}(C_l)$ for any $l \in [H+1]$.

Assuming that the inductive hypothesis holds at the start of some iteration, it is left to prove that it continues to hold at the end of the iteration (assuming Eq. (6) holds whenever MEASURE returns *success*); this implies that the hypothesis also holds at the start of the next iteration and hence also proves the lemma. For any (s,a) appended to C_0, the inductive hypothesis trivially continues to hold as $\Delta_0 = 1/(1-\gamma) \ge v^\star(s)$ for any $s \in \mathcal{S}$ because the rewards are bounded in $[0,1]$. The only other case in which C_l or π_l changes for any l is in Lines 18 and 20, where the changes happen only for $l = \ell + 1$.

We will use Lemma 3.4 to analyze the effect of these updates, thus next we show that the conditions of the lemma are satisfied:

(a) In Lemma D.2 we show that π_ℓ is next-state $(\Delta_\ell + 4(\omega+\varepsilon)(\sqrt{\tilde{d}}+1)/\gamma)$-optimal on $\mathrm{Cover}(C_\ell)$. In the proof of the lemma, we introduce a policy in Eq. (22) that acts as π_ℓ on states in $\mathrm{Cover}(C_\ell)$, and as an optimal stationary deterministic memoryless policy $\pi^\star$ otherwise:

$$\pi_\ell^+(s) = \begin{cases} \pi_\ell(s) & \text{if } s \in \mathrm{Cover}(C_\ell); \\ \pi^\star(s) & \text{otherwise.} \end{cases} \tag{22}$$

Intuitively, this policy corrects π_ℓ on the low-confidence states. The proof of Lemma D.2 then uses the fact that this policy is also q^π-realizable (Definition 1.1) and satisfies $\pi_\ell^+ \in \Pi_{\pi_\ell,\text{Cover}(C_\ell)}$ to show (i) that the q-values of π_ℓ and π_ℓ^+ are close on the measured state-action pairs (via Lemma 4.6 and Lemma D.1); (ii) an optimality guarantee on π_ℓ^+ for all $s \in \mathcal{S}$; and, as a consequence, (iii) the next-state optimality of π_ℓ.

(b) Next, to analyze the effect of Line 18, we introduce hypothetical q-approximators $\tilde{q}_l$ for $l \in [H+1]$, defined as follows: At initialization, $\tilde{q}_l(s,a) = 0$ for all $l \in [H+1]$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$. It is updated every time after Line 16 of the algorithm is executed as

$$\tilde{q}_\ell(s,a) \leftarrow \begin{cases} \tilde{q}_\ell(s,a) & \text{if } s \in \text{Cover}(C_{\ell+1}); \quad (23a) \\ \hat{q}(s,a) & \text{otherwise.} \quad (23b) \end{cases}$$

In other words, $\tilde{q}_\ell$ is only updated to the newly computed $\hat{q}$ for states that are not in $\text{Cover}(C_{\ell+1})$, and stays unchanged for other states. We show in Lemma F.2 that the new policy that $\pi_{\ell+1}$ is updated to, which is constructed in two steps (Lines 17–18), can be expressed as the result of a *single* CAPI policy update that uses $\tilde{q}$:

$$\pi_{\ell+1} \leftarrow \pi_{\tilde{q}_\ell,\pi_\ell,\mathcal{S}\setminus\text{Cover}(C_l)} \,.$$

We show in Lemma F.1 that $\tilde{q}_\ell \approx_{\omega'} q^{\pi_\ell}$ with $\omega' = (\omega+\varepsilon)(\sqrt{\tilde{d}}+1)$ on $\text{Cover}(C_\ell)$.

By the above, we can apply Lemma 3.4 with policy π_ℓ, q-approximation $\tilde{q}_\ell$ (with approximation error guarantee ω' on $\text{Cover}(C_\ell)$, and $\mathcal{S}_{\text{fix}} = \mathcal{S} \setminus \text{Cover}(C_\ell)$ to get that the new value of $\pi_{\ell+1}$ is a $\Delta_{\ell+1} = (8(\omega+\varepsilon)(\sqrt{\tilde{d}}+1) + \gamma\Delta_\ell)$-optimal policy on $\text{Cover}(C_\ell)$. By the end of the loop in Line 20, $\text{Cover}(C_{\ell+1}) = \text{Cover}(C_\ell)$, so $\pi_{\ell+1}$ is $\Delta_{\ell+1}$-optimal on $\text{Cover}(C_{\ell+1})$. This finishes the proof that the inductive hypothesis continues to hold at the end of the iteration, finishing the proof of the lemma. □

F Auxiliary results for Lemma 4.4 about $\tilde{q}_l$

Throughout the execution of CAPI-QPI-PLAN, for $l \in [H+1]$, let $\tilde{q}_l^-$, π_l^-, C_l^- denote the values of variables $\tilde{q}_\ell$, π_ℓ, C_ℓ, respectively, at the time when Lines 16–20 were most recently executed with $\ell = l$ in a previous iteration of the main loop of CAPI-QPI-PLAN. If such a time does not exist, let their values be the initialization values. Thus, C_l^- may (only) change at the start of some iteration i if Lines 16–20 were executed with $\ell = l$ in the previous iteration $i-1$. Observe that whenever this happens, Lines 16–20 may also change $C_{\ell+1}$ in iteration $i-1$, and this is the only time C_{l+1} can be changed for any $l \in [H]$. After this, at the beginning of iteration i, C_{l+1} always has the same elements as C_l^-. Therefore, since it also holds at the initialization of the algorithm, we conclude that at the start of each iteration,

$$\text{Cover}(C_{l+1}) = \text{Cover}(C_l^-) \,. \tag{24}$$

Lemma F.1. *Assume that Eq.* (6) *holds whenever* MEASURE *returns* success*. Then, whenever Line 18 of* CAPI-QPI-PLAN *is executed, for all $s \in Cover(C_\ell)$ and $a \in \mathcal{A}$,*

$$\left|\tilde{q}_\ell(s,a) - q^{\pi''}(s,a)\right| \le (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) \qquad \textit{for all } \pi'' \in \Pi_{\pi_\ell,Cover(C_\ell)} \,. \tag{25}$$

Proof. We prove this by induction for every time Line 18 is executed with any value of ℓ. We first observe that after initialization, C_l is the empty sequence for every l, so we can apply Lemma 4.3 with $q^\star$ and empty sequences ($n = 0$) to get that for any $s \in \text{Cover}(())$ and $a \in \mathcal{A}$, $q^{\pi''}(s,a) \le q^\star(s,a) \le \varepsilon + \sqrt{\lambda}B = \varepsilon + \omega$. Also, $\tilde{q}_l(\cdot,\cdot) = 0$ at initialization, so Eq. (25) holds for any value of ℓ.

Consider a time when Line 18 is executed and assume the inductive hypothesis holds for the previous time Line 18 was executed with the same value of ℓ (or at the initialization if this is the first time), that is,

$$\left|\tilde{q}_\ell^-(s,a) - q^{\pi''}(s,a)\right| \le (\omega+\varepsilon)(\sqrt{\tilde{d}}+1) \qquad \text{for all } \pi'' \in \Pi_{\pi_\ell^-,\text{Cover}(C_\ell^-)},\ s \in \text{Cover}(C_\ell^-) \,.$$

To prove that the statement now holds for any $s \in \text{Cover}(C_\ell)$, first consider any $s \in \text{Cover}(C_{\ell+1}) = \text{Cover}(C_\ell^-)$. For such an s, by Lemma 4.5 we have that $\Pi_{\pi_\ell,\text{Cover}(C_\ell)} \subseteq \Pi_{\pi_\ell^-,\text{Cover}(C_\ell^-)}$. Also, by definition, $\tilde{q}_\ell(s,\cdot) = \tilde{q}_\ell^-(s,\cdot)$ for $s \in \text{Cover}(C_{\ell+1})$. Combining with the inductive hypothesis, it follows that Eq. (25) holds for $s \in \text{Cover}(C_{\ell+1})$.

It remains to show that Eq. (25) also holds for $s \in \text{Cover}(C_\ell) \setminus \text{Cover}(C_{\ell+1})$. For such an s, $\tilde{q}_\ell(s,\cdot) = \hat{q}(s,\cdot)$ by definition, and hence Lemma D.1 implies that Eq. (25) holds in this case.

Combining the two cases, it follows that the inductive hypothesis continues to hold when Line 18 is executed. □

Lemma F.2. *Throughout the execution of* CAPI-QPI-PLAN*, at the start of any iteration, for all* $l \in [H]$*,*

$$\pi_{l+1} = \pi_{\tilde{q}_l^-, \pi_l^-, \mathcal{S} \setminus Cover(C_l^-)} \,. \tag{26}$$

Proof. We prove this by induction for the start of any iteration. Eq. (26) holds at the start of the algorithm due to its initialization (because at initialiaztion, $\tilde{q}_l^-(s,a) = 0$ for all s, a, and hence by our tie-breaking rule, the policy on the right-hand side of Eq. (26) always chooses action $\mathcal{A}_1$, which is the initial policy for π_l).

In what follows, we use the fact that for any $q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, policy π, and $\mathcal{S}_{\text{fix}} \subseteq \mathcal{S}$, the CAPI policy update $\pi_{q,\pi,\mathcal{S}_{\text{fix}}}$ is a policy whose value at any $s \in \mathcal{S}$ only depends on $q(s,\cdot)$, $\pi(s)$, and whether or not $s \in \mathcal{S}_{\text{fix}}$, by definition (Eq. 5). Therefore, for an alternative q', π', $\mathcal{S}'_{\text{fix}}$, for any $s \in \mathcal{S}$, $\pi_{q,\pi,\mathcal{S}_{\text{fix}}}(s) = \pi_{q',\pi',\mathcal{S}'_{\text{fix}}}(s)$ whenever the following three conditions hold: (C1) $q(s,a) = q'(s,a)$ for all $a \in \mathcal{A}$; (C2) $\pi(s) = \pi'(s)$; and (C3) either both or none of $\mathcal{S}_{\text{fix}}$ and $\mathcal{S}'_{\text{fix}}$ include s.

Assume the inductive hypothesis holds at the beginning of some iteration. Let π'' be the policy Line 18 updates $\pi_{\ell+1}$ to, noting that this is the only place where policies are updated. All we need to prove is that π'' is equal to

$$\tilde{\pi} = \pi_{\tilde{q}_\ell, \pi_\ell, \mathcal{S} \setminus \text{Cover}(C_\ell)} \,.$$

First, for any $s \notin \text{Cover}(C_{\ell+1})$, $\pi''(s) = \pi'(s) = \pi_{\hat{q},\pi_\ell,\mathcal{S}\setminus\text{Cover}(C_\ell)}(s)$ and $\hat{q}(s,\cdot) = \tilde{q}_\ell(s,\cdot)$ by definition. Hence, $\pi''(s) = \pi_{\hat{q},\pi_\ell,\mathcal{S}\setminus\text{Cover}(C_\ell)}(s) = \pi_{\tilde{q}_\ell,\pi_\ell,\mathcal{S}\setminus\text{Cover}(C_\ell)}(s) = \tilde{\pi}(s)$, as all of conditions (C1)-(C3) are satisfied for s (C2 and C3 hold trivially).

Next, take any $s \in \text{Cover}(C_{\ell+1}) = \text{Cover}(C_\ell^-)$. Then, by Line 18, $\pi''(s) = \pi_{\ell+1}(s)$. By the inductive hypothesis, the current value of $\pi_{\ell+1}$ can be written as $\pi_{\tilde{q}_\ell^-,\pi_\ell^-,\mathcal{S}\setminus\text{Cover}(C_\ell^-)}$. We prove that this policy takes the same value as $\tilde{\pi}$ at s, by showing conditions (C1)-(C3). First, by Lemma 4.5, $\pi_\ell \in \Pi_{\pi_\ell^-,\text{Cover}(C_\ell^-)}$. Thus, as $s \in \text{Cover}(C_\ell^-)$, $\pi_\ell(s) = \pi_\ell^-(s)$, showing condition (C2). Furthermore, as $s \in \text{Cover}(C_{\ell+1})$, by definition, $\tilde{q}_\ell(s,\cdot) = \tilde{q}_\ell^-(s,\cdot)$, showing condition (C1). Finally, as $s \in \text{Cover}(C_{\ell+1}) = \text{Cover}(C_\ell^-) \subseteq \text{Cover}(C_\ell)$, $s \notin \mathcal{S} \setminus \text{Cover}(C_\ell^-)$ and $s \notin \mathcal{S} \setminus \text{Cover}(C_\ell)$, showing condition (C3).

Combining the two cases, $\pi''(s) = \tilde{\pi}(s)$ for any $s \in \mathcal{S}$, finishing the induction. □

G Efficient implementation and proof of Theorem 1.3

In this section we consider the efficient implementation of CAPI-QPI-PLAN in terms of memory and computational costs of both the algorithm itself and the final policy it outputs.

Focusing on the memory cost, first we can observe that throughout the execution of the algorithm, C_l for all $l \in [H+1]$ only stores up to $\tilde{d}$ unique state-action pairs altogether (cf. Eq. (13)), as they use the same pairs; let $W = (s_i, a_i)_{i \in \hat{d}}$ denote these for some $\hat{d} \le \tilde{d}$. Furthermore, throughout the execution of the algorithm, for any level l, the only features that π_l depends on are the features associated with members of W. Storing all these features takes $d\hat{d}$ memory. Denote all the policies that CAPI-QPI-PLAN constructs in Line 18, in order, as $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(n-1)}$, where n is the number of times Line 18 is executed. Recall from the proof of Theorem 1.2 that the number of times MEASURE returns *success*, which is an upper bounds on n, is itself bounded by $\tilde{d}H$, hence $n \le \tilde{d}H$. Together, Lines 17-18 construct a policy that, for an $s \in \mathcal{S}$, decides whether the action should be $\arg\max_{a \in \mathcal{A}} \langle \varphi(s,a), \theta \rangle$ for some θ given by LSE (Eq. (8)), or the value of the policy should be determined by a recursive call to a previously constructed policy, either $\pi_{\ell+1}$ or π_ℓ (through π'). Now there exist some $a, b \in [n]$ such that $\pi^{(a)} = \pi_\ell$ and $\pi^{(b)} = \pi_{\ell+1}$ before the new policy is constructed in Line 18. To implement the new $\pi_{\ell+1}$ constructed policy, it is enough therefore to store, in addition to the existing policies, θ (from $\hat{q}$), the decision rules, and the indices a and b. The decision rules are fully defined by θ, C_ℓ, and $C_{\ell+1}$. It is therefore enough to further store $C_\ell, C_{\ell+1} \subseteq W$, which can be

encoded as $\hat{d}$-dimensional vectors each, storing the bitmask of which state-action pairs are included. We also store the current value of ℓ (the level) for the newly constructed policy. Together, a policy thus consumes $3 + d + 2\hat{d}$ memory. We store all policies constructed, along with the features of W, and the final value of $V(C_H)^{-1}$, at a memory cost of $d\hat{d} + \tilde{d}H(3 + d + \hat{d}) + d^2 = \tilde{O}(d^2/(1-\gamma))$. This is the memory cost of the final policy outputted by CAPI-QPI-PLAN. The memory cost of running CAPI-QPI-PLAN itself is of the same order, as additionally storing C_l, $\bar{q}_l$, and $V(C_l)^{-1}$ for $l \in [H+1]$ takes $\tilde{O}(d^2/(1-\gamma))$ memory.

To efficiently implement the final policy found by CAPI-QPI-PLAN with the stored information described above, we start from evaluating the last policy constructed, $\pi^{(i)}$ for $i = n-1$. We introduce auxiliary variables $\tilde{V}(C_l)^{-1}$ and $\tilde{C}_l$ for $l \in [H+1]$ to efficiently track the required values of $V(C_l)^{-1}$ and C_l. We keep updating these variables so that for $l \in \{\ell, \ell+1\}$, they match the values of $V(C_l)^{-1}$ and C_l, respectively, at the time of construction of the current policy $\pi^{(i)}$ under consideration, where ℓ is the (saved) level of $\pi^{(i)}$. For $i = n-1$, observe that when it was constructed, $C_0 = C_1 = \cdots = C_H$ by Lemma 4.7. We therefore start by initializing variables $\tilde{V}(C_0)^{-1}, \ldots, \tilde{V}(C_H)^{-1}$ to the saved final value of $V(C_H)^{-1}$, and variables $\tilde{C}_0, \ldots, \tilde{C}_H$ to W. Implementing the decisions of a policy takes an order of $|\mathcal{A}|d^2$ computation ($|\mathcal{A}|$ vector and matrix multiplications), after which we recover either the policy output or a previously constructed policy to recurse into. For the latter case, we have to consider the evaluation of this policy, denoted by $\pi^{(i')}$. Let the (saved) level of $\pi^{(i')}$ be ℓ'. Before we set i to i' and start evaluating it, we need to update the values of $\tilde{V}(C_l)$ and C_l for $l \in \{\ell', \ell'+1\}$. The updates are needed for these two levels only, as the decision rule of policy i' only depends on these levels, as shown before. Let us describe the update procedure for some $l \in \{\ell', \ell'+1\}$: Since $\pi^{(i')}$ was constructed earlier than $\pi^{(i)}$ (i.e., $i' < i$), and $C_{l'}$ can only grow during the algorithm for any $l' \in [H+1]$, we only need to remove members of the variable $\tilde{C}_l$ to match the value of C_l at the time of construction of $\pi^{(i')}$. The members to be removed are given by the difference of the members of $\tilde{C}_l$ and the bitmasks stored for $\pi^{(i')}$ for level l. For each state-action pair (s, a) removed, we also need to update $\tilde{V}(C_l)^{-1}$ to $\left(\tilde{V}(C_l) - \varphi(s,a)\varphi(s,a)^\top\right)^{-1}$, which can be done in order d^2 computation using the Sherman–Morrison–Woodbury formula [Max, 1950]. The total number of such removal operations for any level l is bounded by the sum of the number of state-action pairs in the initialization of $\tilde{C}_{l'}$ (for $l' \in [H+1]$), that is, by $(H+1)\hat{d}$. As a result, the computational cost of the final policy of CAPI-QPI-PLAN is $\tilde{O}((H+1)\hat{d}d^2) + n\tilde{O}(|\mathcal{A}|d^2) = \tilde{O}(d^3|\mathcal{A}|/(1-\gamma))$.

Finally, we consider the computational cost of running CAPI-QPI-PLAN. The number of iterations of the outer loop is bounded by $\tilde{O}(dH) = \tilde{O}(d/(1-\gamma))$, as each iteration involves either a MEASURE call that returns *success*, or a new member added to some C_l. For each iteration, Line 4 takes $\tilde{O}(d^2|\mathcal{A}|)$, Line 7 takes $\tilde{O}(d/(1-\gamma))$, Line 11 takes $\tilde{O}(d^2|\mathcal{A}|)$ computation; for Line 16, calculating θ, the second component of the inner product of the least-squares predictor in Eq. (8) takes $\tilde{O}(d^2)$ computation, and if C_l ever changes for some l, updating $V(C_l)^{-1}$ by the Sherman–Morrison–Woodbury takes $\tilde{O}(d^2)$ computation. Overall, all the operations except those associated to the MEASURE call of Line 9 take $\tilde{O}(d^3|\mathcal{A}|/(1-\gamma))$ computation in total. We conclude our calculations by considering the computational cost of the MEASURE calls, which will dominate the overall computational cost. Line 6 of Algorithm 2 has a computational cost of order $d^2|\mathcal{A}|$, while the majority of the computational cost comes from evaluating the policy at Line 7. By our previous calculations, this takes $\tilde{O}(d^3|\mathcal{A}|/(1-\gamma))$ computation and happens (at most) once for each simulator call. Using the query cost bound of Theorem 1.2, we conclude that the computational cost of CAPI-QPI-PLAN is $\tilde{O}(d^4|\mathcal{A}|(1-\gamma)^{-5}\omega^{-2})$. □

H Query cost lower bounds with random access

In this section we prove lower bounds on the worst-case expected query cost of planning algorithms with a simulator supporting *random access*. Recall from Section 1 that in this setting a planner can issue queries for any state-action pair, not just the ones already visited. As this is a more powerful access to the simulator than *local access*, statements that hold for *all* planners using *random access* (as such, all lower bounds presented in this section) trivially hold for planners using *local access*. We prove two bounds, Theorem H.2 and Theorem H.3, whose combination trivially implies Theorem 1.4.

Formally, the planner interacts with a *random access* simulator that simulates some MDP M as follows: at step t starting from 1, given the whole interaction history $H_t =$

$(S_1, A_1, R_1, S'_1, \ldots, S_{t-1}, A_{t-1}, R_{t-1}, S'_{t-1})$ (where H_1 is the empty sequence by definition), the planner either selects a state-action pair (S_t, A_t), or halts and outputs a stationary memoryless policy. The planner is allowed to randomize. Let τ denote the number of queries the planner sends to the simulator before it halts, and π_τ the policy it outputs. If the planner does not stop, the simulator responds to the query (S_t, A_t) by returning (S'_t, R_t) sampled independently from the transition-reward kernel $Q(S_t, A_t)$ of M. Let $\mathcal{P}_M$ denote the probability measure associated with this procedure, and let $\mathbb{E}_M$ denote the expectation operator corresponding to $\mathcal{P}_M$. Both $\mathcal{P}_M$ and $\mathbb{E}_M$ implicitly depend on the planner, which is omitted in the notation for brevity but will always be clear from the context. Using this notation, clearly $\mathbb{E}_M(\tau)$ is the expected query cost of the planner on M.

As usual, we only consider the query complexity of planners which are reasonable in the sense that they can find a near-optimal policies for a class of MDPs:

Definition H.1 (Soundness and query complexity)**.** *A planner is said to be (α, δ)-sound for an MDP M if, when used with a simulator of M, it halts almost surely (i.e., $\mathcal{P}_M(\tau < \infty) = 1$) and outputs a policy π_τ that is α-optimal for M with probability at least $1 - \delta$, that is,*

$$\mathcal{P}_M\left(v^\star(s_0) - v^{\pi_\tau}(s_0) \le \alpha\right) \ge 1 - \delta\,,$$

where $v^\star$ and v^{π_τ} are the value-functions of the optimal policy and π_τ in the MDP M and s_0 is the initial state of M. A planner is (α, δ)-sound for a class of MDPs $\mathcal{M}$ if it is (α, δ)-sound for every MDP in the class. The query complexity of a planner over $\mathcal{M}$ is defined as the maximum of its expected query cost over the members of the class.

In the rest of the section, for $d \ge 1$ and $L > 0$, we use $\mathcal{B}_d(L) = \{x \in \mathbb{R}^d : \|x\| \le L\}$ to denote the d-dimensional Euclidean ball of radius L centered at the origin.

H.1 Exponential lower bound for planners with small suboptimality

We first show an exponential query complexity lower bound for sound planners that guarantee a small suboptimality bound. The result is a simple application of the techniques in Lattimore et al. [2020], and establishes the barrier for the suboptimality attainable by query-efficient planners:

Theorem H.2. *Let $\delta \le 0.9$, $\alpha \le 0.49/(1-\gamma)$, and $\varepsilon \ge 0$, $d \ge 3$. There is a class of MDPs $\mathcal{M}$ with uniform policy value-function approximation error ε for some d-dimensional feature map such that the query complexity of any (α, δ)-sound planner over $\mathcal{M}$ is at least* $\exp\left(\Omega\left(d\left(\frac{\varepsilon}{\alpha(1-\gamma)}\right)^2\right)\right)$.

Proof. Our proof is based on a similar complexity lower bound of Lattimore et al. [2020] for the multi-armed bandit setting, which is a special case of our problem. As such, we start by rewriting the class of bandit problems they used in their proof in our MDP framework, introducing a set of MDPs $\tilde{\mathcal{M}}$ each of which gets into a terminal state with no rewards after the first step. Let $\alpha' = 2.01\alpha(1-\gamma) \le 1$ and $k = \left\lfloor \exp\left(\frac{d-2}{8}\left(\frac{\varepsilon}{\alpha'}\right)^2\right)\right\rfloor$. $\tilde{\mathcal{M}} = \{\tilde{M}_1, \ldots, \tilde{M}_k\}$ is defined to be a set of k MDPs as follows: Each MDP in $\tilde{\mathcal{M}}$ has k actions (i.e., $\mathcal{A} = [k]$) and two states: $\mathcal{S} = (s_0, s_1)$ with s_0 being the initial state, and deterministic transitions $P(s_1|s,a) = 1$ and $P(s_0|s,a) = 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. For any $i \in [k]$, the reward distribution $\mathcal{R}_i$ for MDP $\tilde{M}_i$ is defined as follows: rewards for state s_1 are deterministically zero, that is, $\mathcal{R}_i(0|s_1, a) = 1$ for all $a \in \mathcal{A}$, making s_1 an absorbing state with zero reward, while rewards for state s_0 are deterministically α' for action i and zero otherwise, that is, $\mathcal{R}_i(\alpha'|s_0, i) = 1$ and $\mathcal{R}_i(0|s_0, j) = 1$ for $j \in [\mathcal{A}]$ with $j \ne i$. Since this class of MDPs is equivalent to the class of muti-armed bandit problems defined by Lattimore et al. [2020], their proof of Corollary 3.3 implies that

- there exists a feature map $\tilde{\varphi} : \mathcal{S} \times \mathcal{A} \to \mathcal{B}_{d-1}(1)$ such that ε is the maximum uniform policy value-function approximation error (Definition 1.1) over $\tilde{\mathcal{M}}$ equipped with features $\tilde{\varphi}$; and
- any planner that almost surely outputs an α'-optimal *deterministic* policy for all $\tilde{M} \in \tilde{\mathcal{M}}$ (when run with a random access simulator for $\tilde{M}$) executes at least

$$\frac{1}{2}\exp\left(\frac{d-2}{8}\left(\frac{\varepsilon}{\alpha'}\right)^2\right) \tag{27}$$

  queries in expectation.

We construct a new set $\mathcal{M} = \{M_1, \dots, M_k\}$ of k MDPs where for each $i \in [k]$, M_i is a slight modification of $\tilde{M}_i$, always returning to the initial state s_0 instead of stopping after the first step: as such, the only modification is that the transition probabilities for all $M \in \mathcal{M}$ are $P(s_0|s,a) = 1$ and $P(s_1|s,a) = 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. Let $\varphi : \mathcal{S} \times \mathcal{A} \to \mathcal{B}_d(2)$ be the features for all MDPs in $\mathcal{M}$, where for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, $\varphi(s,a)$ is a concatenation of the $(d-1)$-dimensional $\tilde{\varphi}(s,a)$ and the scalar 1, so that the dth coordinate of $\varphi(s,a)$ is $\varphi(s,a)_d = 1$.

Fix any $i \in [k]$ and any stationary deterministic memoryless policy π, and let $\tilde{\theta}$ be the parameter realizing the low approximation error for $\tilde{M}_i$ and $\tilde{\varphi}$, that is, satisfying Eq. (19) (see Appendix C for a proof that such a $\tilde{\varphi}$ exists). In what follows, we denote q- and v-functions (with arbitrary superscripts) of an MDP M by adding M as a superscript to the corresponding function. Let θ be a concatenation of $\tilde{\theta}$ and the scalar $\gamma v^{\pi}_{M_i}(s_0)$. For any $(s,a) \in \mathcal{S} \times \mathcal{A}$,

$$q^{\pi}_{M_i}(s,a) = q^{\pi}_{\tilde{M}_i}(s,a) + \gamma v^{\pi}_{M_i}(s_0) \approx_{\varepsilon} \left\langle \tilde{\varphi}(s,a), \tilde{\theta} \right\rangle + \gamma v^{\pi}_{M_i}(s_0) = \langle \varphi(s,a), \theta \rangle \ .$$

The uniform policy value-function approximation error therefore remains at most ε for M_i with feature map φ, and this is true for any $i \in [k]$. We can therefore take any (α, δ)-sound planner with query complexity T (for some $T \geq 0$) over $\mathcal{M}$, and provide it with a simulator of M_i for any $i \in [k]$ (which we can trivially build with access to a simulator of $\tilde{M}_i$), to get a policy π that is α-optimal for M_i with $\mathcal{P}_{M_i}$-probability at least $1 - \delta$. Recall that the rewards of M_i are 0 for every action apart from a single optimal action, i, where the reward is α'. Thus, $v^{\star}_{M_i}(s_0) = \alpha'/(1-\gamma)$ and $v^{\pi}_{M_i}(s_0) = \alpha'\pi(i|s_0)/(1-\gamma) = \pi(i|s_0)v^{\star}_{M_i}(s_0)$. Thus, with probability at least $1-\delta$, $v^{\star}_{M_i}(s_0) - v^{\pi}_{M_i}(s_0) \leq \alpha < 0.5\alpha'/(1-\gamma) = 0.5 v^{\star}_{M_i}(s_0)$. Therefore, $\pi(i|s_0) > 0.5$. As we know that the optimal action achieves a deterministic reward of α', we can test with a single query whether the action that π assigns the highest probability to is optimal. If not, we can run the planner again and repeat the check. Since each run of the planner is successful with probability at least $1-\delta$, independently of each other, almost surely one of the checks eventually passes and we output the deterministic policy that chooses the optimal action. Now the number of times the planner needs to be run is a stopping time (with respect to the sequence of the runs) with expectation at most $1/(1-\delta)$, hence the expected query cost of the whole procedure is at most $(T+1)/(1-\delta)$ by Wald's equation. Note that the same policy is α'-optimal for $\tilde{M}_i$. Therefore, the planner defined above almost surely outputs an α'-optimal *deterministic* policy for any MDP in $\tilde{\mathcal{M}}$, and hence by Eq. (27) we have

$$T \geq \frac{1}{2}(1-\delta)\exp\left(\frac{d-2}{8}\left(\frac{\varepsilon}{\alpha'}\right)^2\right) - 1 \ .$$

Therefore $T = \exp\left(\Omega\left(d\left(\frac{\varepsilon}{\alpha(1-\gamma)}\right)^2\right)\right)$, finishing the proof. □

H.2 Lower bound for linear MDPs

We close this section by proving two lower bounds on the query complexity of *random access* planners for linear MDPs: Theorem H.3 and Corollary H.4.

We start by recalling the definition of linear MDPs [Zanette et al., 2020]: An MDP with countable state space is said to be *linear* if there exists a feature map $\varphi : \mathcal{S} \times \mathcal{A} \to \mathcal{B}_d(L)$, a state-transition feature map $\psi : \mathcal{S} \to \mathbb{R}^d$, and a reward parameter $\theta_r \in \mathcal{B}_d(B)$ such that $r(s,a) = \langle \varphi(s,a), \theta_r \rangle$ and $P(s'|s,a) = \langle \varphi(s,a), \psi(s') \rangle$ for any $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, and $\sum_{s \in \mathcal{S}} \|\psi(s)\|_2 \leq B$. Clearly, any linear MDP satisfies Definition 1.1 with $\varepsilon = 0$. As such, the lower bounds presented below trivially transfer to the ε uniform policy value-function approximation error case for any $\varepsilon \geq 0$.

Theorem H.3. *Let $\delta \in (0, 0.2)$, $\gamma \in [\frac{7}{12}, 1]$, $H = 1/(1-\gamma)$, $\alpha \in (0, 0.1\gamma H/(1+\gamma)^2]$, and $d \geq 3$. Then there is a class of linear MDPs $\mathcal{M}$ such that the query complexity of any (α, δ)-sound planner over $\mathcal{M}$ is at least $\Omega\left(d^2H^3/\alpha^2\right)$.*

Corollary H.4. *Under the conditions of Theorem H.3, there is a class of linear MDPs $\mathcal{M}'$ such that the expected query cost of any (α, δ)-sound planner over $\mathcal{M}'$ is at least $\Omega\left(d^2H^3/\alpha^2 + dH^3\log(1/\delta)/\alpha^2\right)$.*

In the remainder of the section we prove the above bounds. Throughout we assume that the conditions in Theorem H.3 are satisfied. We start with the construction of the class $\mathcal{M}$ of MDPs, then prove several auxiliary results, before presenting the proofs of Theorem H.3 and Corollary H.4.

The construction of $\mathcal{M}$ is based on a combination of hard tabular MDPs [Xiao et al., 2022] and hard linear bandit problems [Lattimore and Szepesvári, 2020, Section 24.1]. Each MDP in $\mathcal{M}$ has two states: $\mathcal{S} = \{s_0, s_1\}$ with s_0 being the initial state. The action space is the intersection of a unit sphere and a $(d-2)$-dimensional hypercube: $\mathcal{A} = \{\pm 1/\sqrt{d-2}\}^{d-2}$. We construct MDPs M_β for all $\beta \in \mathcal{A}$, and let $\mathcal{M} = \{M_\beta \mid \beta \in \mathcal{A}\}$. The feature map φ is defined, for any $a \in \mathcal{A}$, as

$$\varphi(s_0, a) = (1, 0, a^\top)^\top \quad \text{and} \quad \varphi(s_1, a) = (0, 1, 0, \dots, 0)^\top .$$

We define the linear MDPs M_β to have deterministic rewards for any $\beta \in \mathcal{A}$. Thus, M_β is fully defined by its reward parameter θ_r and state-transition feature map ψ, according to the definition of linear MDPs. Let $\theta_r = (1, 0, \dots, 0)^\top$, making state s_0 the only rewarding state, as then for all $a \in \mathcal{A}$,

$$r_\beta(s_0, a) = \langle \theta_r, \varphi(s_0, a) \rangle = 1 \quad \text{and} \quad r_\beta(s_1, a) = \langle \theta_r, \varphi(s_0, a) \rangle = 0.$$

Let $\Delta = 2(1+\gamma)^2 \alpha / (\gamma H^2)$; since $\alpha \le 0.1\gamma H/(1+\gamma)^2$, $\Delta \le 0.2/H = 0.2(1-\gamma)$. Let

$$\psi(s_0) = (\gamma, 0, \Delta\beta^\top)^\top \quad \text{and} \quad \psi(s_1) = (1-\gamma, 1, -\Delta\beta^\top)^\top.$$

This implies that

$$\begin{aligned} P_\beta(s_0|s_0, a) &= \gamma + \Delta\beta^\top a, & P_\beta(s_1|s_0, a) &= 1 - \gamma - \Delta\beta^\top a, \\ P_\beta(s_0|s_1, a) &= 0, & P_\beta(s_1|s_1, a) &= 1. \end{aligned}$$

Our assumptions guarantee that P_β defines a valid transition kernel with probabilities in $[0, 1]$. The MDP starts in s_0 and rewards are collected until the state transitions to s_1, which is a terminal state with zero reward.

For the proof, we also need the following notation and supporting lemmas.

Notation. The probability measure $\mathcal{P}_{M_\beta}$ induced by the interconnection of a planner and a simulator for M_β is written for simplicity as $\mathcal{P}_\beta$. Similarly, $\mathbb{E}_{M_\beta}$ is written as $\mathbb{E}_\beta$. v_β (with arbitrary superscripts) denotes value functions (corresponding to the superscripts) of M_β. For any integer $i \in \{1, \dots, d-2\}$, $\mathrm{err}_i(\pi, \beta) = \sum_{a \in \mathcal{A}} \pi(a|s_0) I_{\mathrm{sgn}(a_i) \neq \mathrm{sgn}(\beta_i)}$ denotes the average error of a policy π at the ith coordinate, where a_i and β_i are the ith components of a and β, respectively, and I_E is the indicator function of event E. With a slight abuse of notation, for a stationary memoryless policy π, we let $\pi^\top \beta$ denote $\sum_{a \in \mathcal{A}} \pi(a|s_0) a^\top \beta$.

Lemma H.5. *For any $M_\beta \in \mathcal{M}$, the value function of a stationary memoryless policy π is given by*

$$v_\beta^\pi(s_0) = \frac{1}{1 - \gamma^2 - \gamma\Delta\pi^\top\beta}, \quad \textit{and} \quad v_\beta^\pi(s_1) = 0 .$$

Proof. It clearly holds that $v_\beta^\pi(s_1) = 0$. From the Bellman equation, $v_\beta^\pi(s_0) = 1 + \gamma(\gamma + \Delta\pi^\top\beta) v_\beta^\pi(s_0)$, and the claim follows from solving this equation for $v_\beta^\pi(s_0)$. □

It is easy to see that the optimal policy in M_β is defined by $\pi_\beta^\star(\beta|s_0) = 1$ (the actions in s_1 do not matter). Hence, by the above lemma,

$$v_\beta^\star(s_0) - v_\beta^\pi(s_0) = \frac{\gamma\Delta(1 - \pi^\top\beta)}{\left(1 - \gamma^2 - \gamma\Delta\right)\left(1 - \gamma^2 - \gamma\Delta\pi^\top\beta\right)} . \tag{28}$$

Because $1 - \pi^\top\beta = 2\sum_{i=1}^{d-2} \mathrm{err}_i(\pi, \beta)/(d-2)$,

$$v_\beta^\star(s_0) - v_\beta^\pi(s_0) = \frac{2\gamma\Delta \sum_{i=1}^{d-2} \mathrm{err}_i(\pi, \beta)}{(d-2)\left(1 - \gamma^2 - \gamma\Delta\right)\left(1 - \gamma^2 - \gamma\Delta\pi^\top\beta\right)} . \tag{29}$$

Accordingly, to prove a lower bound on the suboptimality of π, we need a lower bound for the sum of errors, $\sum_{i=1}^{d-2} \mathrm{err}_i(\pi, \beta)$. To this end, Lemma H.6 below plays a key role.

Lemma H.6 (Error Probability Lower Bound)**.** *For any planner with stopping time τ, there exists a $\beta \in \mathcal{A}$ such that*

$$\sum_{i=1}^{d-2} \mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau, \beta) \ge \frac{1}{2}\right) \ge \frac{d-2}{2} - \frac{d-2}{2}\sqrt{1 - \exp\left(-\frac{5\Delta^2 H \mathbb{E}_\beta[\tau]}{(d-2)^2}\right)} . \tag{30}$$

To prove Lemma H.6, we need some technical lemmas. First, let $\mathcal{F}_t$ for any $t \in \mathbb{N}^+$ denote the σ-algebra generated by random variables in H_t, with $\mathcal{F}_1$ being the trivial σ-algebra. $\mathbb{F} = (\mathcal{F}_t)_{t=1}^{\infty}$ is chosen to be the filtration. The following lemma is adopted from Exercise 15.7 of Lattimore and Szepesvári [2020] with a slight modification.

Lemma H.7 (KL-divergence decomposition)**.** *Let M and M' be two MDPs differing only in their transition probability kernels, denoted by P and P', respectively. Then, for any any $\mathbb{F}$-adapted stopping time τ satisfying $\mathcal{P}_M(\tau < \infty) = 1$, and an $\mathcal{F}_\tau$-measurable*[1] *random variable Z,*

$$\mathrm{KL}\left(\mathcal{P}_M^Z \middle\| \mathcal{P}_{M'}^Z\right) \leq \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathbb{E}_M\left[\mathcal{N}_\tau(s,a)\right] \mathrm{KL}\left(P(\cdot|s,a) \| P'(\cdot|s,a)\right),$$

where $\mathcal{P}_M^Z$ and $\mathcal{P}_{M'}^Z$ are the laws of Z under $\mathcal{P}_M$ and $\mathcal{P}_{M'}$, respectively, $\mathcal{N}_t(s,a)$ denotes the number of queries with $(s,a) \in \mathcal{S}\times\mathcal{A}$ up to time step t, and $\mathrm{KL}(\cdot,\cdot)$ denotes the Kullback-Leibler (KL-) divergence of two distributions.

The next lemma provides an upper bound on the KL-divergence of certain next-state distributions. A similar result appears in the proof of Lemma 6.8 of Zhou et al. [2020], but it requires that $\gamma \geq 2/3$; ours only requires the weaker assumption that $\gamma \geq 7/12$.

Lemma H.8. *Take any $\beta, \beta' \in \mathcal{A}$ that only differ at a single coordinate. Then for any action $a \in \mathcal{A}$,*

$$\mathrm{KL}\left(P_\beta(\cdot|s_0,a) \middle\| P_{\beta'}(\cdot|s_0,a)\right) \leq \frac{5\Delta^2 H}{(d-2)^2}.$$

Proof. Our proof relies on Proposition 2 of Xiao et al. [2022]: for two Bernoulli distributions $\mathrm{Ber}(p)$ and $\mathrm{Ber}(p')$ with parameters $p, p' \in (0,1)$, it holds that

$$\mathrm{KL}\left(\mathrm{Ber}(p) \| \mathrm{Ber}(p')\right) \leq \frac{(p-p')^2}{2\min\{p(1-p), p'(1-p')\}}.$$

Since $P_\beta(s_1|s_0,a) = 1-\gamma-\Delta\beta^\top a$ and $P_{\beta'}(s_1|s_0,a) = 1-\gamma-\Delta(\beta')^\top a$,

$$\begin{aligned}\mathrm{KL}\left(P_\beta(\cdot|s_0,a) \middle\| P_{\beta'}(\cdot|s_0,a))\right) &\leq \frac{\Delta^2((\beta-\beta')^\top a)^2}{2\min_{b\in\mathcal{A}}(\gamma+\Delta\beta^\top b)(1-\gamma-\Delta\beta^\top b)} \\ &= \frac{2\Delta^2}{(d-2)^2\min_{b\in\mathcal{A}}(\gamma+\Delta\beta^\top b)(1-\gamma-\Delta\beta^\top b)}\end{aligned}$$

for any action $a \in \mathcal{A}$. Note that

$$\min_{b\in\mathcal{A}}(\gamma+\Delta\beta^\top b)(1-\gamma-\Delta\beta^\top b) \overset{(a)}{\geq} (\gamma+\Delta)(1-\gamma-\Delta) \overset{(b)}{\geq} \frac{1-\gamma-\Delta}{2} \overset{(c)}{\geq} \frac{2(1-\gamma)}{5},$$

where (a) is due to the fact that $x(1-x)$ is monotone decreasing for $x \geq 0.5$ and $\gamma+\Delta\beta^\top b \geq \gamma-\Delta \geq 0.5$ since $\gamma \geq 7/12$ and $\Delta \leq 0.2(1-\gamma)$, (b) follows since $0.5 \leq \gamma+\Delta$, and (c) holds because $\Delta \leq 0.2(1-\gamma)$. This concludes the proof of the lemma. □

Now we are ready to prove Lemma H.6.

Proof of Lemma H.6. Let $\beta^{(i)}$ be a vector obtained by flipping the sign of β's i^{th} coordinate. Then,

$$\begin{aligned}&\mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta) \geq \frac{1}{2}\right) + \mathcal{P}_{\beta^{(i)}}\left(\mathrm{err}_i(\pi_\tau,\beta^{(i)}) \geq \frac{1}{2}\right) \\ &= \mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta) \geq \frac{1}{2}\right) + \mathcal{P}_{\beta^{(i)}}\left(\mathrm{err}_i(\pi_\tau,\beta) \leq \frac{1}{2}\right) \\ &\geq \mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta) \geq \frac{1}{2}\right) + \mathcal{P}_{\beta^{(i)}}\left(\mathrm{err}_i(\pi_\tau,\beta) < \frac{1}{2}\right) \\ &\geq 1 - \sqrt{1-\exp\left(-\mathrm{KL}\left(\mathcal{P}_\beta^{\mathrm{err}_i(\pi_\tau,\beta)} \middle\| \mathcal{P}_{\beta^{(i)}}^{\mathrm{err}_i(\pi_\tau,\beta)}\right)\right)}\end{aligned}$$

[1] By a slight abuse of notation, $\mathcal{F}_\tau$ is the σ-algebra generated by the random vector (with random length) $(S_1, A_1, R_1, S_1', \ldots, S_{\tau-1}, A_{\tau-1}, R_{\tau-1}, S_{\tau-1}')$.

where $\mathcal{P}_\beta^{\mathrm{err}_i(\pi_\tau,\beta)}, \mathcal{P}_{\beta^{(i)}}^{\mathrm{err}_i(\pi_\tau,\beta)} \in \mathcal{M}_1([0,1])$ are the laws of the random variable $\mathrm{err}_i(\pi_\tau,\beta)$ in M_β and $M_{\beta^{(i)}}$, respectively , and the last line follows from an improved Bretagnolle-Huber inequality (inequality (14.11) of Lattimore and Szepesvári [2020]). Applying Lemmas H.7 and H.8 to the KL-divergence in the exponent in the right hand side of the above inequality together with the fact that $\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathbb{E}_\beta\left[\mathcal{N}_\tau(s,a)\right] \le \mathbb{E}_\beta[\tau]$, we can further lower-bound the last line by

$$1-\sqrt{1-\exp\left(-\mathrm{KL}\left(\mathcal{P}_\beta^{\mathrm{err}_i(\pi_\tau,\beta)}\middle\|\mathcal{P}_{\beta^{(i)}}^{\mathrm{err}_i(\pi_\tau,\beta)}\right)\right)} \ge 1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}.$$

Therefore,

$$\begin{aligned}
&\frac{1}{|\mathcal{A}|}\sum_{\beta\in\mathcal{A}}\sum_{i=1}^{d-2}\mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta)\ge\frac{1}{2}\right)\\
&\quad=\frac{1}{|\mathcal{A}|}\sum_{i=1}^{d-2}\frac{1}{2}\sum_{\beta\in\mathcal{A}}\left[\mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta)\ge\frac{1}{2}\right)+\mathcal{P}_{\beta^{(i)}}\left(\mathrm{err}_i(\pi_\tau,\beta^{(i)})\ge\frac{1}{2}\right)\right]\\
&\quad\ge\frac{d-2}{2}-\frac{d-2}{2}\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}
\end{aligned}$$

where the first equality holds because for any β, there is exactly one $\beta^{(i)}$ in $\mathcal{A}$. As $\max_{\beta\in\mathcal{A}} f(\beta) \ge \sum_{\beta\in\mathcal{A}} f(\beta)/|\mathcal{A}|$ for any $f:\mathcal{A}\to\mathbb{R}$, $\arg\max_{\beta\in\mathcal{A}}\sum_{i=1}^{d-2}\mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta^{(i)})\ge 1/2\right)$ satisfies the claim of the lemma. □

Now we are ready to prove Theorem H.3.

Proof of Theorem H.3. Take any (α,δ)-sound planner on $\mathcal{M}$ with stopping time τ. Let $\mathrm{err}(\pi,\beta) := \sum_{i=1}^{d-2}\mathrm{err}_i(\pi,\beta)$ for brevity. From Eq. (29),

$$\begin{aligned}
\mathbb{E}_\beta\left[v_\beta^\star(s_0)-v_\beta^{\pi_\tau}(s_0)\right] &\ge \frac{2\gamma\Delta\mathbb{E}_\beta\left[\mathrm{err}(\pi_\tau,\beta)\right]}{(d-2)\left(1-\gamma^2-\gamma\Delta\right)\left(1-\gamma^2+\gamma\Delta\right)}\\
&\ge \frac{\gamma\Delta}{(d-2)\left(1-\gamma^2-\gamma\Delta\right)\left(1-\gamma^2+\gamma\Delta\right)}\sum_{i=1}^{d-2}\mathcal{P}_\beta\left(\mathrm{err}_i(\pi_\tau,\beta)\ge\frac{1}{2}\right)\\
&\ge \frac{\gamma\Delta}{2\left(1-\gamma^2-\gamma\Delta\right)\left(1-\gamma^2+\gamma\Delta\right)}\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right), \quad (31)
\end{aligned}$$

where the first inequality holds because $\pi^\top\beta \ge -1$ for any stationary memoryless policy π, the second inequality is due to the Markov inequality, while the last inequality holds by Lemma H.6. From Eq. (29) and $\pi^\top\beta \le 1$ we also have that

$$\begin{aligned}
\mathbb{E}_\beta\left[v_\beta^\star(s_0)-v_\beta^{\pi_\tau}(s_0)\right] &\le \frac{2\gamma\Delta\mathbb{E}_\beta\left[\mathrm{err}(\pi_\tau,\beta)\right]}{(d-2)\left(1-\gamma^2-\gamma\Delta\right)^2}\\
&\le \frac{\gamma\Delta}{4\left(1-\gamma^2-\gamma\Delta\right)^2}\left[3\mathcal{P}_\beta\left(\mathrm{err}(\pi_\tau,\beta)>\frac{d-2}{4}\right)+1\right],
\end{aligned}$$

where the second inequality holds because

$$\begin{aligned}
\mathbb{E}_\beta\left[\mathrm{err}(\pi_\tau,\beta)\right] &= \mathbb{E}_\beta\left[\mathrm{err}(\pi_\tau,\beta)I_{\mathrm{err}(\pi_\tau,\beta)>\frac{d-2}{4}}+\mathrm{err}(\pi_\tau,\beta)I_{\mathrm{err}(\pi_\tau,\beta)\le\frac{d-2}{4}}\right]\\
&\le \mathbb{E}_\beta\left[(d-2)I_{\mathrm{err}(\pi_\tau,\beta)>\frac{d-2}{4}}+\frac{d-2}{4}I_{\mathrm{err}(\pi_\tau,\beta)\le\frac{d-2}{4}}\right]\\
&= \frac{3(d-2)}{4}\mathcal{P}_\beta\left(\mathrm{err}(\pi_\tau,\beta)>\frac{d-2}{4}\right)+\frac{d-2}{4}.
\end{aligned}$$

Combining this result with Eq. (31),

$$\begin{aligned}
\mathcal{P}_\beta\left(\mathrm{err}(\pi_\tau,\beta) > \frac{d-2}{4}\right) &\geq \frac{2}{3}\frac{1-\gamma^2-\gamma\Delta}{1-\gamma^2+\gamma\Delta}\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3} \\
&= \frac{2}{3}\left(1-\frac{2\gamma\Delta}{1-\gamma^2+\gamma\Delta}\right)\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3} \\
&> \frac{2}{3}\left(1-\frac{2\gamma\Delta}{1-\gamma^2}\right)\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3}\,.
\end{aligned}$$

Note that $\mathrm{err}(\pi_\tau,\beta) > (d-2)/4$ implies that $v^\star_\beta(s_0) - v^{\pi_\tau}_\beta(s_0) > \alpha$ since as before

$$v^\star_\beta(s_0) - v^{\pi_\tau}_\beta(s_0) \geq \frac{2\gamma\Delta\mathrm{err}(\pi_\tau,\beta)}{(d-2)\left((1-\gamma^2)^2-\gamma^2\Delta^2\right)} > \frac{1}{2}\frac{\gamma\Delta}{(1-\gamma^2)^2-\gamma^2\Delta^2} > \frac{1}{2}\frac{\gamma\Delta}{(1-\gamma^2)^2} = \alpha\,,$$

where the last equality follows because $\Delta = 2(1+\gamma)^2\alpha/(\gamma H^2) = 2(1-\gamma^2)^2\alpha/\gamma$. Therefore

$$\begin{aligned}
\mathcal{P}_\beta\left(v^\star_\beta(s_0) - v^{\pi_\tau}_\beta(s_0) > \alpha\right) &> \frac{2}{3}\left(1-\frac{2\gamma\Delta}{1-\gamma^2}\right)\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3} \\
&\overset{(a)}{\geq} \frac{2}{3}\left(1-\frac{0.4\gamma}{1+\gamma}\right)\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3} \\
&\overset{(b)}{\geq} \frac{8}{15}\left(1-\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\right)-\frac{1}{3} \\
&\geq \frac{3}{15}-\frac{8}{15}\sqrt{1-\exp\left(-\frac{5\Delta^2 H\mathbb{E}_\beta[\tau]}{(d-2)^2}\right)}\,.
\end{aligned}$$

where (a) follows since $\Delta = 2(1+\gamma)^2\alpha/(\gamma H^2) \leq 0.2/H$ because $\alpha \leq 0.1\gamma H/(1+\gamma)^2$, and (b) follows since $0 \leq 0.4x/(1+x) \leq 0.2$ for $x \in [0,1]$.

This implies that unless $\mathbb{E}_\beta[\tau] \geq \Omega\left(d^2H^3/\alpha^2\right)$, the algorithm is not (α,δ)-sound. Indeed if

$$\mathbb{E}_\beta[\tau] \leq -\frac{(d-2)^2}{5\Delta^2 H}\log\left(1-\frac{9}{64}\left(1-5\delta\right)^2\right)\,,$$

it holds that $\mathcal{P}_\beta\left(v^\star_\beta(s_0) - v^{\pi_\tau}_\beta(s_0) > \alpha\right) > \delta$, contradicting the assumption that the planner is (α,δ)-sound on $\mathcal{M}$. □

Finally we prove Corollary H.4. We augment $\mathcal{M}$ with a class of linear MDPs similar to those in $\mathcal{M}$ but with $\mathcal{A} = \{e_i \in \mathbb{R}^{d-2} \mid i = 1,\ldots,d-2\}$, $\beta \in \mathcal{A}$, and appropriate $\Delta \in (0,1)$, where e_i is a unit vector whose i^{th} coordinate is non-zero. This class of linear MDPs is essentially the same as the one in Xie and Jiang [2020, Figure 2], and a lower bound of $\Omega(dH^3\log(1/\delta)/\alpha^2)$ is known to hold for it. Therefore we have the following lower bound

$$\Omega\left(\max\left\{\frac{d^2H^3}{\alpha^2}, \frac{dH^3}{\alpha^2}\log\frac{1}{\delta}\right\}\right)\,,$$

and we may derive the claimed lower bound by using $\max\{a,b\} \geq (a+b)/2$.