Best Policy Identification in Linear MDPs

Jérôme Taupin*, Yassir Jedra*, Alexandre Proutiere*
*KTH Royal Institute of Technology, Stockholm, Sweden
†ENS, Paris, France
Email: {jedra, alepro}@kth.se, jerome.taupin@ens.psl.eu

Abstract—We consider the problem of best policy identification in discounted Linear Markov Decision Processes in the fixed confidence setting, under both generative and forward models. We derive an instance-specific lower bound on the expected number of samples required to identify an ε-optimal policy with probability 1−δ. The lower bound characterizes the optimal sampling rule as the solution of an intricate non-convex optimization program, but can be used as the starting point to devise simple and near-optimal sampling rules and algorithms. We devise such algorithms. In the generative model, our algorithm exhibits a sample complexity upper bounded by \( O((d(1−γ)−1/(ε + Δ)²)(log(1/δ) + d)) \) where \( Δ \) denotes the minimum reward gap of sub-optimal actions and \( d \) is the dimension of the feature space. This upper bound holds in the moderate-confidence regime (i.e., for all \( δ \)), and matches existing minimax and gap-dependent lower bounds. In the forward model, we determine a condition under which learning approximately optimal policies is possible; this condition is weak and does not require the MDP to be ergodic nor communicating. Under this condition, the sample complexity of our algorithm is asymptotically (as \( δ \) approaches 0) upper bounded by \( O(σ²(1−γ)−1/(ε + Δ)²)(log(1/δ)) \) where \( σ² \) is an instance-specific constant, value of an optimal experiment-design problem. To derive this bound, we establish novel concentration results for random matrices built on Markovian data.

I. INTRODUCTION

In Reinforcement Learning (RL), an agent interacts with an unknown controlled stochastic dynamical system, with the objective of identifying as quickly as possible an approximately optimal control policy. In this paper, we consider dynamical systems modelled through discounted Markov Decision Processes (MDPs), and investigate the problem of best policy identification in the fixed confidence setting. More precisely, we aim at devising (ε, δ)-PAC RL algorithms, i.e., algorithms identifying ε-optimal policies with a level of certainty greater than 1 − δ, using as few samples as possible. Such a learning objective has been considered extensively in tabular MDPs both in the discounted and episodic settings, most often using a mimimax approach, see e.g. [1–10] and more recently adopting an instance-specific analysis [11, 12]. According to the aforementioned work, in tabular MDPs, the minimal sample complexity for identifying an ε-optimal policy with probability at least 1 − δ scales as \( \frac{S A}{ε²} \log(1/δ) \) (ignoring the dependence in the time-horizon or discount factor), where \( S \) and \( A \) represent the sizes of the state and action spaces, respectively. These results illustrate the curse of dimensionality (tabular MDPs with limited state and action spaces only are learnable), and highlight the need for the use of function approximation towards the design of scalable RL algorithms. Despite the empirical successes of RL algorithms leveraging function approximation, our theoretical understanding of these methods remain limited. In this paper, we investigate the so-called linear MDPs, where linear functions are used to approximate the system dynamics and rewards. We aim at devising statistically and computationally efficient algorithms for the best policy identification with fixed confidence learning task. We address this task under both (i) the generative model, where in each round, a sample of the transition and reward from any given state-action pair can be observed; and (ii) the forward model, where the learner has access to a single controlled trajectory of the system. Our contributions are summarized below.

(a) Sample complexity lower bounds. We derive instance-specific lower bounds that any (ε, δ)-PAC algorithm must satisfy, for both the generative and forward models. These lower bounds are characterized by the solution of an intricate optimization problem. We propose a careful relaxation of these optimization problems. These relaxations suggest an experiment design approach based on G-optimal design to define the sampling strategy used to explore the MDP.

(b) Algorithms with a generative model. When the learner has access to a generative model, inspired by our sample complexity lower bounds, we devise G-Sample-and-Stop (GSS), a simple (ε, δ)-PAC algorithm that relies on G-optimal design [13, Chap. 21], least-squares estimators, and a proper stopping rule. We show that the expected sample complexity of GSS scales at most as

\[ \left( \frac{(d(1−γ)−1)/(ε + Δ)²}{log(1/δ)} \right) \]

up to logarithmic factors, where \( Δ \) is an appropriately defined instance-specific sub-optimality gap that depends on the MDP \( M \).

This upper bound holds in the moderate-confidence regime (i.e., for all \( δ \in (0,1) \)), and matches existing minimax and gap-dependent lower bounds.

(c) Algorithms with the forward model. Again inspired by our sample complexity lower bounds, we propose G-Navigate-and-Stop (GNS). The analysis of GNS or other algorithms for the forward model presents many challenges: (i) In contrast with episodic setting, we do not have the ability to restart the trajectory at each episode. Hence, suitable conditions are required to ensure that learning is even possible from a single controlled trajectory. (ii) Because of the linear structure, the uniqueness of the optimal sampling policy that arise from our lower bounds is not guaranteed, and the set of such optimal policies does not have nice properties such as convexity. Therefore, a careful sampling scheme is required. (iii) The data generated when exploring the MDP
is Markovian, which implies that new concentration results for random matrices with Markovian data must be derived. We overcome these challenges. First, we determine conditions under which learning approximately optimal policies is possible; these conditions are weak and do not require the MDP to be ergodic nor communicating. Then, under such conditions, we establish concentration bounds on random matrices with Markovian data. Finally, we show that the sample complexity of GNS, under the learnability conditions, is asymptotically (as $\delta$ approaches 0) upper bounded by $O((1 - \gamma)^2 d H^3/\Delta_{\min})$ where $\Delta_{\min}$ is an instance-specific constant, value of an optimal experiment design problem.

II. RELATED WORK

Linear models in RL have attracted a lot of attention over the last few years. We summarize below the recent results, related to first episodic MDPs and then discounted MDPs.

**Episodic linear MDPs.** Most of the studies have aimed at devising algorithms minimizing regret. Jin et al. [14] propose an optimistic Least Squares Value Iteration (LSVI) algorithm that achieves a regret upper bound of order $O(\sqrt{d^3 T})$ and that can be implemented in polynomial time. [15] presents UCRL-VTR, a confidence based algorithm adapted to the linear MDP setting. The algorithm achieves a gap dependent regret of order $O((d^2 H^5)/\Delta_{\min}) \log(T/\delta)$. When it comes to best policy identification problems, in [16], Wagenmaker et al. establish a sample complexity minimax lower bound of order $\Theta(d^2 H^2/\epsilon^2)$. The authors also propose an a reward-free algorithm with sample complexity of order $O(d/\epsilon^2 + dH^2)$. In a subsequent work, Wagenmaker et al. [17] introduce PEDEL, an elimination based algorithm with instance-specific sample complexity guarantees. In the worst case, the sample complexity upper bound scales as $O(dH^2/\epsilon^2 + d^2 H^2 + \log(1/\delta))$. This bound hides a dependence on $\lambda_{\min}$, the maximal minimum eigenvalue of the covariates matrix that can be induced by a policy. As in our work, the derived instance-specific sample complexity guarantees are related to G-optimal design and take the following form: $C_0G + C_1$, where $G = H^4 \sum_{h=1}^{H} \inf_{A_{\epsilon,p}} \max_{\pi \in \Pi} \log\left(\frac{\|\phi, h\|_{A_{\epsilon,p}}}{\lambda_{\min}(\Pi)}\right)$ with $C_0 = \log\left(\frac{1}{\epsilon}\right) \right.$ polylog$(H, \log(1/\epsilon))$ and $C_1 = \text{poly}(d, H, 1/\lambda_{\min}, \log(1/\delta), \log(1/\epsilon), \log(\|\Pi\|))$. Note that PEDEL requires as input a set of policies $\Pi$. The authors propose a way to approximate the set of all policies using restricted linear soft-max policies $\Pi_{e}$ which leads to an overall sample complexity of order $C_0H^4 \sum_{h=1}^{H} \inf_{A_{\epsilon,p}} \max_{\pi \in \Pi_{e}} \log\left(\frac{\|\phi, h\|_{A_{\epsilon,p}}}{\lambda_{\min}(H^2 + \log(1/\delta))}\right) + C_1$. In [18], the authors also investigate the problem of identifying an $\epsilon$-optimal policy with a generative model and propose a Linear Approximate Value Iteration algorithm (LAVI). They leverage the idea of anchor (state, action) pairs but require a set of such anchor pairs for each layer $h \in [H]$. 

**Discounted linear MDPs.** In [19], Yang et al. focus on the $\epsilon$-optimal policy identification problem in the generative setting and present Phased Parametric Q-Learning (PPQ-learning), an algorithm with sample complexity of order $\tilde{O}(\frac{d}{\epsilon^2 (1 - \gamma)^2})$ under the restrictive assumption that a so-called set of (state, action) anchor pairs exist (see Assumption 2) and that it is of size $d$. More precisely, this assumption states that there exists $K \subseteq S \times A$, a set of anchor (state, action) pairs such that for all $(s, a) \in K \subseteq S \times A$, $\phi(s, a)$ can be written as convex combination of features of anchor pairs. The authors further assume that $|K| = d$ and that all features have non-negative entries and that the features correspond to probability vectors. The authors finally provide a matching minimax lower bound of order $\Omega(\frac{d}{\epsilon^2 (1 - \gamma)^2})$.

Lattimore et al. [20] also consider the $\epsilon$-optimal policy identification problem in the generative setting. They devise a sampling rule based on G-optimal design and use an approximate policy iteration algorithm to recover the optimal policy. Their algorithm seeks to estimate the Q function directly at each iteration, by first evaluating the value of Q at anchor (state, action) pairs (determined by the G-optimal design) via rollout, and then by generalizing using least squares. The sample complexity of their algorithm is of the order $O(\frac{d\sqrt{d}}{\epsilon (1 - \gamma)^2})$.

Finally it is worth mentioning [21], where Zhou et al. consider the regret minimization problem in the forward model. The notion of regret for discounted MDPs is not easy to define. Here, the authors consider the accumulated difference of rewards between an Oracle policy and the proposed policy but along the trajectory followed under the latter policy (this policy could well lead the system into regions of the state space). The proposed algorithm achieves a regret scaling at most as $O(\frac{d\sqrt{d}}{1 - \gamma^2})$.

III. MODELS AND OBJECTIVES

**A. Notation**

We denote by $\|x\|$ the Euclidean norm of a vector $x \in \mathbb{R}^d$. For a given definite positive matrix $M \in \mathbb{R}^{d \times d}$, we denote by $\|x\|_M = \sqrt{x^T M x}$ the weighted Euclidean norm of the vector $x \in \mathbb{R}^d$. We denote by $\|M\|$ the operator norm of a matrix $M \in \mathbb{R}^{d \times d}$. For a positive definite matrix $M \in \mathbb{R}^{d \times d}$, we denote its highest (resp. smallest) eigenvalue by $\lambda_{\max}(M)$ (resp. $\lambda_{\min}(M)$), respectively. For a given pair of two symmetric matrices $A, B \in \mathbb{R}^{d \times d}$, we write $A \succeq B$ (resp. $A \succ B$) to mean that $A - B$ is positive definite (resp. positive semi-definite).

**B. Discounted linear MDPs**

We consider an infinite time-horizon discounted MDP, denoted $M = (S, A, p_M, q_M, \gamma)$, where $S$ and $A$ are the state and action spaces, respectively, $p_M$ and $q_M$ are the dynamics and reward distributions, respectively, and $\gamma \in (0, 1)$ is the discount factor. More precisely, starting from state $s$ and given that action $a$ is selected, the probability to move to state $s'$ is $p_{Ms}(s, a, s')$ and the distribution of the collected reward is $q_M(s, a)$. We assume that $q_M(s, a)$ has support on $[0, 1]$, and
we denote by \( r_{\mathcal{M}}(s, a) \) the expected reward of \( q_{\mathcal{M}}(s, a) \). \( \mathcal{S} \) and \( \mathcal{A} \) are finite sets of respective cardinalities \( \mathcal{S} \) and \( \mathcal{A} \).

Each state-action pair \((s, a)\) is associated to a feature vector \( \phi(s, a) \in \mathbb{R}^d \). We assume that the feature map \( \phi \) is known to the learner, that for all \((s, a) \in \mathcal{S} \times \mathcal{A} \), \( \|\phi(s, a)\| \leq 1 \), and that the features \( (\phi(s, a))(s,a) \in \mathcal{S} \times \mathcal{A} \) span \( \mathbb{R}^d \). We are interested in the class of the so-called Linear MDPs, denoted by \( \mathbb{M} \), and defined as follows [14]:

**Definition 1** (Linear MDPs). We say \( \mathcal{M} \) is a Linear MDP if there exists \( \mu_{\mathcal{M}}, \theta_{\mathcal{M}} \in \mathbb{R}^d \), such that for all \((s, s', a) \in \mathcal{S} \times \mathcal{A} \), and \( \phi(s, a) \), we have \( q_{\mathcal{M}}(s, a, s') = \phi(s, a)^T \theta_{\mathcal{M}}(s') \) and \( r_{\mathcal{M}}(s, a) = \phi(s, a)^T \mu_{\mathcal{M}}(s) \) with \( \max(\|\theta_{\mathcal{M}}\|, \|\mu_{\mathcal{M}}(s)\|) \leq \sqrt{d} \).

A deterministic stationary control policy \( \pi \) maps states to actions. We denote by \( s_t \) the state at time \( t \) under the policy \( \pi \), and by \( \pi(s) \) the action selected by \( \pi \). The performance of a policy \( \pi \) is expressed through its state value function \( V_{\pi, \mathcal{M}} \) and its state-action value function \( Q_{\pi, \mathcal{M}} \) defined by: for all state-action pairs \((s, a) \in \mathcal{S} \times \mathcal{A} \),

\[
V_{\pi, \mathcal{M}}(s) = \mathbb{E}_{\mathcal{M}} \left[ \sum_{t=0}^{\infty} \gamma^t r_{\mathcal{M}}(s_t, \pi(s_t)) | s_0 = s \right],
\]

\[
Q_{\pi, \mathcal{M}}(s, a) = r_{\mathcal{M}}(s, a) + \gamma \sum_{s' \in \mathcal{S}} p_{\mathcal{M}}(s, a, s') V_{\pi, \mathcal{M}}(s').
\]

An optimal policy \( \pi_{\mathcal{M}}^* \) for the MDP \( \mathcal{M} \) maximizes the value function for any state, i.e., for any policy \( \pi \), we have \( V_{\pi_{\mathcal{M}}^*, \mathcal{M}}(s) \geq V_{\pi, \mathcal{M}}(s) \) for all \( s \in \mathcal{S} \). The state and state-action value functions of \( \pi_{\mathcal{M}}^* \) are referred to as the value function \( V_{\mathcal{M}}^* \) and the Q function \( Q_{\mathcal{M}}^* \), respectively. A policy \( \pi \) is said \( \epsilon \)-optimal if \( \max_{s \in \mathcal{S}} V_{\pi, \mathcal{M}}(s) - V_{\mathcal{M}}^*(s) \leq \epsilon \) point-wise, and we denote by \( \Pi_{\epsilon, \mathcal{M}} \) the set of \( \epsilon \)-optimal policies of \( \mathcal{M} \).

**C. Best policy identification**

We aim at designing a learning algorithm interacting with the MDP \( \mathcal{M} \) so as to identify an \( \epsilon \)-optimal policy as quickly as possible. We formalize this objective in a PAC framework, where a learning algorithm consists of (i) a sampling rule, (ii) a stopping rule and (iii) a decision rule.

(i) **Sampling rule**: We distinguish between the generative and the forward model:

1. Generative model: In each round \( t \), the sampling rule may select any (state, action) \((s_t, a_t)\) to explore depending on past observations.

2. Forward model: Under this model, the learner is forced to follow the trajectory of the system, and only the action may be selected.

Under both models, from the selected pair, the learner observes the next state and receives a sample of the corresponding reward.

(ii) **Stopping rule**: This rule is defined through a stopping time \( \tau \) deciding when the learner stops gathering information and wishes to output an estimated \( \epsilon \)-optimal policy.

(iii) **Decision rule**: Based on the observations gathered before stopping, the learner outputs an estimated optimal policy \( \hat{\pi}_\tau \).

We are interested in learning algorithms that are \((\epsilon, \delta)\)-PAC in the following sense:

**Definition 2** ((\(\epsilon, \delta)\)-PAC algorithms). An algorithm is said \((\epsilon, \delta)\)-PAC if at the time it stops \( \tau \), it outputs a policy \( \hat{\pi}_\tau \) satisfying:

\[
\Pr_{\mathcal{M}} \left( \max_{s \in \mathcal{S}} \left( V_{\hat{\pi}_\tau, \mathcal{M}}(s) - V_{\mathcal{M}}^*(s) \right) < \epsilon \right) \geq 1 - \delta.
\]

Our goal is to design \((\epsilon, \delta)\)-PAC algorithms with minimal sample complexity \( \mathbb{E}_{\mathcal{M}}[\tau] \). In contrast with most existing analyses, we will derive instance-specific lower and upper bounds on the sample complexity of such algorithms \((\epsilon, \delta)\)-PAC algorithms. In particular, we wish these bounds to depend on the sub-optimality gap of the MDP \( \mathcal{M} \) defined by \( \Delta_{\mathcal{M}} = \min_{s \in \mathcal{S}, a \neq \pi^*_{\mathcal{M}}(s)} (V_{\mathcal{M}}^*(s) - Q_{\mathcal{M}}^*(s, a)) \).

IV. **Sample Complexity Lower Bounds**

To state our instance-specific lower bounds, we first introduce the following notation. Given two MDPs \( \mathcal{M} \) and \( \mathcal{M}' \) in \( \mathbb{M} \), we write \( \mathcal{M} \ll \mathcal{M}' \) if for every pair \((s, a) \in \mathcal{S} \times \mathcal{A} \), we have \( p_{\mathcal{M}}(s, a, \cdot) \ll p_{\mathcal{M}'}(s, a, \cdot) \) and \( q_{\mathcal{M}}(s, a) \ll q_{\mathcal{M}'}(s, a) \). In this case, we define the Kullback-Leibler divergence between \( \mathcal{M} \) and \( \mathcal{M}' \) by:

\[
KL_{\mathcal{M}, \mathcal{M}'}(s, a) = KL(q_{\mathcal{M}}(s, a) || q_{\mathcal{M}'}(s, a)) + KL(p_{\mathcal{M}}(s, a, \cdot) || p_{\mathcal{M}'}(s, a, \cdot)).
\]

We also denote by \( kl(a, b) \) the Kullback-Leibler divergence of two Bernoulli distributions of respective means \( a \) and \( b \). Finally, we introduce the following set of MDPs. This set includes MDPs for which the set of \( \epsilon \)-optimal policies does not contain an \( \epsilon \)-optimal policy for \( \mathcal{M} \).

\[
\text{Alt}_{\epsilon}(\mathcal{M}) = \left\{ \mathcal{M}' \in \mathbb{M} : \left\{ \Pi_{\epsilon, \mathcal{M}}(\mathcal{M}) \right\} = \emptyset \right\}
\]

We refer to \( \text{Alt}_{\epsilon}(\mathcal{M}) \) as the set of alternative MDPs w.r.t. \( \mathcal{M} \). Let \( \Sigma_{\mathcal{S} \times \mathcal{A}} \) be the probability simplex in \( \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \), and define for all \( \omega \in \Sigma_{\mathcal{S} \times \mathcal{A}} \):

\[
T_{\mathcal{M}}(\omega)^{-1} = \inf_{\mathcal{M}' \in \text{Alt}_{\epsilon}(\mathcal{M})} \sum_{s,a} \omega_{s,a} KL_{\mathcal{M}, \mathcal{M}'}(s, a). \tag{1}
\]

a) **With a generative model.** For the generative model, we establish the following lower bound.

**Proposition 1.** Let \( \epsilon > 0 \), \( \delta \in (0, 1) \). The sample complexity \( \tau \) of any \((\delta, \epsilon)\)-PAC algorithm must satisfy:

\[
\mathbb{E}_{\mathcal{M}}[\tau] \geq T^{*}_{\mathcal{M}, \text{gen}} \cdot kl(\delta, 1 - \delta)
\]

where \( T^{*}_{\mathcal{M}, \text{gen}} = \inf_{\omega \in \Sigma_{\mathcal{S} \times \mathcal{A}}} T_{\mathcal{M}}(\omega) \).

The derivation of the lower bound in Proposition 1 relies on standard change-of-measure arguments. We defer the proof to [22]. The vector \( \omega \in \Sigma_{\mathcal{S} \times \mathcal{A}} \) solving the optimization problem and leading to \( T^{*}_{\mathcal{M}, \text{gen}} \) can be interpreted as the optimal proportions of times an optimal algorithm should sample the various (state, action) pairs. It turns out, as in the
case of tabular MDPs (see [11]), that analyzing and computing this allocation is difficult. Instead, our strategy will be to derive instance-specific upper bounds of the $T_{M,\text{gen}}$ that can be computed in a computationally efficient manner. To state the upper bounds, we introduce the following quantities: let $\omega \in \Sigma_{S \times A}$, $\Lambda(\omega) = \sum_{(s,a) \in S \times A} \omega_{s,a} \phi(s, a) \phi(s, a)^T$, and $\sigma(\omega) = \max_{(s,a) \in S \times A} \|\phi(s, a)\|_2^2 \Lambda(\omega)^{-1}$. $\Lambda(\omega)$ is referred to as the feature matrix. Furthermore, observe that the function $\sigma(\cdot)$ corresponds to the so-called G-optimality criterion (see e.g. Chap. 21 in [13]). Our next result is to establish a link between $T_{M,\text{gen}}(\cdot)$ and $\sigma(\cdot)$.

**Theorem 1.** For all $\omega \in \Sigma_{S \times A}$, it holds that

$$T_{M,\text{gen}}(\omega) \leq \frac{10\sigma(\omega)}{3(1-\gamma)^4(\Delta_M + \varepsilon)^2}. \tag{2}$$

Consequently, we have $T_{M,\text{gen}} \leq U_{M,\text{gen}} \triangleq \frac{10d}{3(1-\gamma)^4(\Delta_M + \varepsilon)^2}$.

Theorem 1 relates the experiment-design approach based on G-optimality to our instance dependent lower bound. A similar link has been established in the case of best-arm identification in linear bandits [23]. However, establishing such a link in the case of Linear Discounted MDPs is more challenging and requires a careful relaxation of the optimization problem leading to the definition of $T_{M,\text{gen}}(\omega)$ in (1). The proof of Theorem 1 is deferred to [22].

From an algorithmic perspective, Theorem 1 tells us that sampling according to a G-optimal design $\omega^* \in \arg\min_{\omega \in \Sigma_{S \times A}} \sigma(\omega)$ is sufficient to identify an $\epsilon$-optimal policy with a sample complexity upper bounded by the sample complexity $U_{M,\text{gen}} \log(1/\delta)$. $\omega^*$ only depends on the feature map $\phi$ and not the unknowns $\mu_M$ and $\theta_M$, and therefore may be computed prior to the learning process.

b) With a forward model. Proposition 1 and Theorem 1 can be immediately extended to the forward model. To simplify the exposition, we will restrict our attention to the asymptotic lower bounds when $\delta \to 0$. As in [12], we can establish that if $\omega_{sa}$ denotes the expected proportion of rounds where the state-action pair $(s, a)$ is visited, then the allocation $\omega$, asymptotically, must satisfy the balance equations of the Markov chain induced by the controlled system dynamics: for all $s \in S$,

$$\sum_{a \in A} \omega_{s,a} = \sum_{(s',a') \in S \times A} p_{M}(s',a',s) \omega_{s',a'} \tag{3}$$

Define $\Omega(M) = \{\omega \in \Sigma_{S \times A} : \text{the constraints (3) hold}\}$.

**Proposition 2.** Let $\varepsilon > 0$, $\delta \in (0,1)$. In the forward model, the sample complexity $\tau$ of any $(\varepsilon, \delta)$-PAC algorithm must satisfy: $E_M[\tau] \geq T_{M,\text{for}}(\delta(1-\delta))$ where $T_{M,\text{for}} = \inf_{\omega \in \Omega(M)} T_{M,\text{for}}(\omega)$.

**Theorem 2.** Let $\sigma_{M,\text{for}} = \inf_{\omega \in \Omega(M)} \sigma(\omega)$. Then, we have

$$T_{M,\text{for}} \leq U_{M,\text{for}} \triangleq \frac{10 \sigma_{M,\text{for}}}{3(1-\gamma)^4(\Delta_M + \varepsilon)^2}. \tag{4}$$

The proof of Proposition 2 and Theorem 2 are presented in [22]. The upper bound we obtain on $T_{M,\text{for}}$, suggests an experiment design approach where the objective is to sample according to an allocation $\omega^* \in \arg\min_{\omega \in \Omega(M)} \sigma(\omega)$. This objective is similar in spirit to that considered in [17] for Episodic Linear MDPs.

V. THE G-SAMPLE-AND-STOP ALGORITHM

We propose G-Sample-and-Stop (GSS), an algorithm whose sample complexity matches the complexity measure $U_{M,\text{gen}} \log(1/\delta)$ presented in Theorem 1. The algorithm samples the state-action pairs according to a G-optimal design, and stops when it has gathered enough information. The adaptive nature of the stopping rule ensures a gap-dependent sample complexity upper bound.

**A. Sampling rule**

Prior to the learning process, under the GSS algorithm, we start by finding\(^2\) an optimal allocation $\omega^* \in \arg\min_{\omega \in \Sigma_{S \times A}} \sigma(\omega)$. Then, at each round $t$, the algorithm proceeds by sampling a state-action pair $(s_t, a_t)$ according to $\omega^*$. Define $P_t = \sum_{t=1}^{\infty} \phi(s_t, a_t) \phi(s_t, a_t)^T$. Standard concentration arguments on random matrices ensure that the random matrix $P_t$ converges to the matrix $\Lambda(\omega^*)$. In particular, $t \max_{(s,a) \in S \times A} \|\phi(s,a)\|_2^2 \leq 2 \sigma(\omega^*)$. We present this fact in the following proposition, and its proof is deferred to [22].

**Proposition 3.** Let $\delta \in (0,1)$. We have

$$\Pr \left( \max_{(s,a) \in S \times A} \|\phi(s,a)\|_2^2 \leq 2 \sigma(\omega^*) \right) \geq 1 - \delta,$$

provided $t \geq 10d \log \left( \frac{2d}{\delta} \right)$.

**B. Least-squares estimation**

The stopping and decision rules of GSS leverage the least-squares estimators of the parameters $\mu_M$ and $\theta_M$. We provide below explicit expressions for these estimators and derive concentration inequalities characterizing their performance. When the algorithm selects (state, action) pair $(s_t, a_t)$ in round $t$, it observes the next state $s_{t+1}$ and receives the reward $r_t$. Overall, in round $t$, the algorithm gathers the experience $(s_t, a_t, r_t, s_{t+1})$. The regularized least-squares estimators with parameter $\lambda > 0$ of $\mu_M$ and $\theta_M$ after $t$ experiences are given by: for all $s \in S$,

$$\hat{\mu}_t(s) = (P_t + \lambda I_d)^{-1} \sum_{\ell=1}^{t} \phi(s_{\ell}, a_{\ell}) \mathbf{1}_{\{s_{\ell+1} = s\}} \tag{5}$$

$$\hat{\theta}_t = (P_t + \lambda I_d)^{-1} \sum_{\ell=1}^{t} \phi(s_{\ell}, a_{\ell}) r_{\ell}.$$
the least-squares estimators can be controlled in the following sense:

**Proposition 4.** Irrespective of the sampling rule, we have for all $\delta \in (0, 1)$,

$$\mathbb{P} \left( \forall t \geq 1, \left\| \hat{\theta}_t - \theta_{\mathcal{M}} + \gamma (\mu_t - \mu_{\mathcal{M}})^\top \hat{V}_t^* \right\|_{F_t} \leq \beta (\delta, t) \right) \geq 1 - \delta$$

(6)

with the threshold $\beta (\delta, t) = \frac{C}{1 - \gamma} \log (e / \delta) + d \log (dt)$ for some universal constant $C > 0$.

The proof of Proposition 4 is presented in [22] along with the precise constants. Importantly, the threshold $\beta$ does not exhibit any dependence in $S$ but only in $d$. This is thanks to the linear structure that characterizes the value function. Such a structure allows us to use a net argument on the space of all possible optimal value functions. This idea is borrowed from [14] and repurposed to our needs.

**C. Stopping and decision rules**

Let us start by describing the stopping rule. For all $t \geq 1$, we define the random variable $Z(t)$ and the threshold $\beta (\delta, t)$ as follows

$$Z(t) = \frac{3(1 - \gamma)^2 (\Delta_{\mathcal{M}} + \epsilon)^2}{10 \max_{(s, a) \in \mathcal{S} \times \mathcal{A}} \| \phi (s, a) \|^2_{P_t^{-1}}},$$

$$\beta (\delta, t) = C \left( \log \left( \frac{e}{\delta} \right) + d \log (dt) \right).$$

The random variable $Z(t)/t$ may be interpreted as an empirical estimator of the lower bound on $U_{\mathcal{M}, \text{gen}}^{*}$ established in Theorem 1. The choice of the threshold $\beta (\delta, t)$ is motivated by the concentration result of Proposition 4 with $C$ being the universal constant in the statement of the proposition. Finally, the stopping rule of GSS is defined by the stopping time

$$\tau = \inf \{ t \geq 1 : Z(t) > \beta (\delta, t) \}.$$  

(7)

This stopping rule is inspired by classical log-likelihood based stopping rules. When the algorithm stops, it computes $\hat{\pi}_\tau$, an optimal policy for the MDP $\mathcal{M}_t$. The description of GSS is now complete and summarized in Algorithm 1.

**Algorithm 1 G-Sample-and-Stop (GSS)**

Compute $\omega^* = \arg \min_{\omega \in \Sigma_{\mathcal{S} \times \mathcal{A}}} \sigma (\omega)$

while $Z(t) \leq \beta (\delta, t)$ do

sample $(s_t, a_t)$ according $\omega^*$

observe the experience $(s_t, a_t, r_t, s_{t+1})$

update $(\mu_t, \theta_t)$ according to (5) and set $t = t + 1$

return $\hat{\pi} = \pi_t^*$ the optimal policy of $\mathcal{M}_t$

The following Lemma establishes the $(\varepsilon, \delta)$-PAC correctness of GSS.

**Lemma 1.** Under the GSS algorithm, we have:

$$\mathbb{P} (\tau < +\infty, \hat{\pi} \notin \Pi_* (\mathcal{M}) \leq \delta).$$

**D. Sample complexity guarantees under GSS**

Finally, in Theorem 3 we present the sample complexity guarantee enjoyed by GSS.

**Theorem 3.** The sample complexity of GSS satisfies, for all $\varepsilon > 0$, $\delta \in (0, 1)$,

$$\mathbb{E} [\tau] \leq C U_{M, \text{gen}}^{*} \left( \log \left( \frac{e}{\delta} \right) + d \log (U_{M, \text{gen}}^{*}) \right)$$

(8)

where $C > 0$ is a universal constant. Furthermore, GSS is an $(\varepsilon, \delta)$-PAC algorithm.

The proof of Theorem 3 is presented in Appendix ???. First, observe that the sample complexity guarantee is valid for all $\delta \in (0, 1)$ which contrasts with most existing asymptotic results in best policy identification. Additionally, our guarantee is matching, up to a constant multiplicative factor, the upper bound established in Theorem 1 as $\delta \to 0$.

VI. THE G-Navigate-and-Stop Algorithm

In this section, we present G-Navigate-and-Stop (GNS), an algorithm whose sample complexity matches the complexity measure $U_{M, \text{for}}^{*} \log (1/\delta)$ presented in Theorem 2. The design of GNS, as that of GSS, is guided by our lower bounds. In particular, the stopping and decision rules are the same as those of GSS and all guarantees related to these components also hold for GNS, namely Proposition 4 and Lemma 1. The major difference lies in the sampling rule where now we have to account for navigation constraints.

**A. Sampling rule**

In what follows, we denote, for ease of notations, for all $\ell \geq 1$, $\phi_{\ell} = \phi (s_{\ell}, a_{\ell})$. Recall that $P_t = \sum_{\ell=1}^{t} \phi_{\ell} \phi_{\ell}^\top$. As already mentioned in the study of the generative model, this random matrix plays a crucial role. In the forward model, the role $P_t$ is more pronounced and in fact all our learnability conditions concern this matrix.

1) Forced exploration: Learning from a single trajectory requires the existence of at least a policy that explores the MDP sufficiently. Additionally if there is any hope for finding an optimal exploration strategy then we need at least to guarantee that while searching for such a policy, we do not get trapped in states that irrevocably limit our exploration. This motivates the definition of $(m, \lambda)$-covering policies.

**Definition 3** $(m, \lambda)$-covering policy. A policy $\pi$ is said to be an $(m, \lambda)$-covering policy of $\mathcal{M}$ if there exists $m \geq 1$ and $\lambda > 0$ such that:

$$\min_{s \in \mathcal{S}} \lambda \min \left( \frac{1}{m} \mathbb{E}_{\mathcal{M}} \left[ \sum_{t=1}^{m} \phi_t \phi_t^\top | s_1 = s \right] \right) > \lambda.$$  

(9)

We make the following assumption, which is necessary to ensure that learnability is possible.

**Assumption 1.** There exists an $(m, \lambda)$-covering policy $\pi_e$. Furthermore, the learner is aware of the policy $\pi_e$ and of $m$.

It is worth noting that Assumption 1 does not require a priori that the MDP $\mathcal{M}$ is ergodic nor communicating. For a
where \( x_t = 1 \) with probability \( t^{-1/2m} \) and \( x_t = 0 \) with probability \( 1 - t^{-1/2m} \). Then, we have
\[
\mathbb{P} \left( \lambda_{\min} \left( \sum_{t=1}^T \phi_t \phi_t^\top \right) \geq t^{-1/8} \right) \geq 1 - \delta, \quad \text{provided that} \quad t \geq \left( \frac{3m}{\delta^2} \log \left( \frac{2d}{\delta} \right) \right)^2.
\]

The proof of Lemma 2 relies on a careful decomposition of \( P_t \) and using a matrix martingale Bernstein concentration bound. We refer the reader to [22] for the proof. As it turns out, the high probability guarantee on the growth of the smallest eigenvalue in Lemma 2 is sufficient to ensure consistency of the least-squares estimator of \( \mu_M \). This is required for the sample complexity analysis of GNS.

2) Tracking: Before we present our tracking procedure, we present what we refer to as the oracle policy of a given allocation \( \omega \).

a) Oracle policy.: As in [24], given an allocation \( \omega \in \Omega(M) \), we define the oracle policy \( \pi^o(\omega) \) as follows: for all \((s,a) \in S \times A\),
\[
\pi^o(\omega)(a|s) = \begin{cases} 
\frac{\omega_{s,a}}{\sum_{a' \in A} \omega_{s,a'}} & \text{if } \sum_{a' \in A} \omega_{s,a'} > 0, \\
0 & \text{otherwise.}
\end{cases}
\]

It is not difficult to verify that the policy \( \pi^o(\omega) \) indeed induces the allocation \( \omega \).

b) Optimal allocations.: Next, we make the following assumption to avoid unnecessary technical issues that may arise characterizing the set of optimal allocations.

Assumption 2. There exists \( \eta > 0 \) such that
\[
\left\{ \arg \min_{\omega \in \Omega(M)} \sigma(\omega) \right\} \cap \Omega_\eta(M) = \emptyset
\]
where \( \Omega_\eta(M) \triangleq \{ \omega \in \Omega(M) : \lambda(\omega) \geq 2\eta I_d \} \). Furthermore, the learner has access to \( \eta \).

Under Assumption 2, we can characterize the set of optimal allocations\(^3\) as being non-empty, compact and convex. However, it is not guaranteed that the optimal allocation is unique\(^4\). This complicates the design of the tracking procedure. In particular, we cannot use the C-tracking rule used for instance in tabular MDPs as in [24]. To circumvent this issue, we use lazy updates or a doubling trick. Let \( T = \{ 2^k : k \in \mathbb{N} \} \). The allocation \( \omega_t \) that GNS tracks is updated only when \( t \in T \).

c) Optimization oracle.: We assume that the learner has access to an optimization oracle that given a model \( \hat{M} \), outputs an allocation \( \omega^* \in \arg \min_{\omega \in \Omega_M} \sigma(\omega) \). This optimization problem is convex and therefore computationally tractable.

We are now ready present the sampling rule of GNS. When \( t \in T \), the algorithm computes \( \omega^* \in \arg \min_{\omega \in \Omega_{1/2}(\hat{M})} \sigma(\omega) \), and updates \( \pi_t \) as \( \pi_t^o(\omega_t) \). Now in each round \( t, b_t \) is sampled according to \( \pi_t(\cdot|s_t) \) and GNS selects the action \( a_t \) defined in (10). The pseudo-code of GNS is presented in Algorithm 2.

Algorithm 2 The G-Navigate-and-Stop

Initialize \( \pi_1 \) to be the uniform policy while \( Z(t) \leq \beta(\delta, t) \) do

if \( t \in T \) then compute \( \omega_t \in \arg \min_{\omega \in \Omega_{1/2}(\hat{M}_t)} \sigma(\omega) \) and set \( \pi_t \leftarrow \pi_t^o(\omega_t) \) following (11)

sample \( b_t \sim \pi_t(\cdot|s_t) \), and \( a_t \) according to (10)

update \((\mu_t, \theta_t)\) according to (5), and set \( t = t + 1 \)
return \( \hat{\pi}_t \) an optimal policy of \( \hat{M}_t \)

Next, we provide tools for the sample complexity analysis of GNS. One crucial step is to guarantee that under our sampling scheme, certain random matrices concentrate.

Assumption 3. There exists \( \kappa > 0 \), such that for all \( \omega \in \Omega_{1/2}(\hat{M}) \), \( u \in \mathbb{R}^{SA} \), \( (s,a) \in S \times A \), the following holds
\[
\mathbb{E}^{\pi^o(\omega)} \left( \lim_{t \to \infty} M_t(u) | s_1 = s, a_1 = a \right) \leq \kappa,
\]
where \( \mathbb{E}^{\pi^o(\omega)} \) means that the expectation is taken with respect to trajectories generated by policy \( \pi^o(\omega) \), and \( M_t(u) = \sum_{t=1} \left( u^\top \Lambda(\omega)^{-1/2} \phi_t \right) \).

Essentially, Assumption 3 guarantees the convergence of \( \frac{1}{t} \sum_{t=1} \phi_t \phi_t^\top \) towards \( \Lambda(\omega) \) when the sample trajectory is generated with the fixed policy \( \pi^o(\omega) \). The uniform bound \( \kappa \) across state-action pairs in \( S \times A \), allocations in \( \Omega_{1/2}(\hat{M}) \), and all unit vector in \( \mathbb{R}^{SA} \) may appear strong. However, it can be shown for instance that if \( M \) is ergodic then \( \kappa = \mathcal{O}(t_{\text{mix}}/\eta) \) where \( t_{\text{mix}} \) is a the mixing time of the MDP \( M \). Now, we present our concentration bounds on random matrices with Markovian data.

Proposition 5. Let \( \omega_k \) be the optimal allocation used by GNS between \( t_k < t < t_{k+1} \) for some \( k \geq 1 \). Furthermore, assume that \( \omega_k \in \Omega_{\eta}(\hat{M}) \). Then, under GNS with the forced exploration (10), under Assumption 3, we have, for all \( \varepsilon > 0, \delta \in (0, 1) \)
\[
\mathbb{P} \left( \frac{1}{t_{k+1} - t_k} \sum_{t=t_k+1}^{t_{k+1}} \phi_t \phi_t^\top \geq (1 - \varepsilon) \Lambda(\omega_k) \right) \geq 1 - \delta, \quad \text{provided that} \quad t_{k+1} - t_k \geq \max \left( \frac{16 \varepsilon}{C^2}, \frac{16 \varepsilon}{C^2} \right) \log \left( \frac{2d}{\delta} \right) \log \left( \frac{2d}{\delta} \right)
\]
for some universal constant \( C > 0 \).

The proof of Proposition 5 relies on decomposing \( \sum_{t=t_k+1}^{t_{k+1}} \phi_t \phi_t^\top \) using Poisson’s equation [26] so as to obtain a martingale that can be easily controlled under Assumption

\(^3\)This claim follows from Berge’s maximum theorem. Refer to appendix ?? for a formal statement.

\(^4\)The non-uniquity of the optimal allocation also occurs in best arm identification for linear bandits (see e.g.,[25]). The non-unicty is a consequence of Carathéodory’s theorem.
3. Refer to [22] for a complete proof along with the precise constants.

B. Sample complexity guarantees under GNS

Finally, in Theorem 4, we present a sample complexity upper bound for GNS.

**Theorem 4.** The sample complexity of GNS, satisfies for all \( \varepsilon > 0 \),

\[
E[\tau] \leq C \mu_{\text{for}}^* \left( \log \left( \frac{e}{\delta} \right) \right) + o \left( \log \left( \frac{e}{\delta} \right) \right) \tag{12}
\]

for some universal constant \( C > 0 \). Furthermore, the GNS algorithm is \((\varepsilon, \delta)\)-PAC.

The proof of Theorem 4 is slightly more complex than that of Theorem 3 due to the navigation constraints. We present the proof in Appendix ???. Observe that the GNS algorithm attains a sample complexity that matches, up to some multiplication constant and asymptotically (as \( \delta \to 0 \)), the complexity measure \( \mu_{\text{for}}^* \log(1/\delta) \) presented in Theorem 2.

**VII. CONCLUSION**

In this paper, we have first derived instance-dependent lower bounds on the sample complexity of best policy identification in discounted linear MDPs. As of now, these instance-dependent bounds remain challenging to exploit algorithmically. Instead, we proposed a relaxation that links these lower bounds to experiment-design criteria based on the G-optimal design. These criteria lead to the sample complexity measures \( \mu_{\text{gen}}^* \log(1/\delta) \) and \( \mu_{\text{for}}^* \log(1/\delta) \) for the generative model and forward model, respectively. Importantly, these complexity measures are instance-dependent as they exhibit a dependence on the minimum gap \( \Delta_M \).

Furthermore, we have established that these experiment design criteria can be exploited algorithmically by proposing the algorithms GSS and GNS with sample complexity upper bounds matching asymptotically \( \mu_{\text{gen}}^* \log(1/\delta) \) and \( \mu_{\text{for}}^* \log(1/\delta) \), respectively, as \( \delta \to 0 \). In fact, GSS enjoys a stronger guarantee that holds for all \( \delta \in (0, 1) \) and matches existing minimax lower bounds (in the episodic case, these bounds are of the order \( \Omega(d^2/e^2) \)). In the forward model, we are the first, to the best of our knowledge, to investigate the problem of \( \varepsilon \)-best policy identification for discounted linear MDPs. Notably, we establish, for this model, conditions under which learnability is possible. These conditions are a priori weaker than ergodicity and communication.

As a future direction, we believe that it would be interesting to improve the relaxations of the lower bounds, as well as devising, for the forward model, algorithms with sample complexity guarantees in the moderate confidence regimes.

**VIII. ACKNOWLEDGMENTS**

Yassir Jedra and Alexandre Proutiere are supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

**REFERENCES**

[1] Michael Kearns and Satinder Singh. Finite-sample convergence rates for q-learning and indirect algorithms. *Advances in Neural Information Processing, 11*, 04 1999.

[2] Sham Machandranath Kakade. On the sample complexity of reinforcement learning. PhD thesis, University of London, England, 2003.

[3] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research, 7*(Jun):1079–1105, 2006.

[4] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J Kappen. Minimax pac bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning, 91*(3):325–349, 2013.

[5] Aaron Sidford, Mengdi Wang, Xian Wu, Lin Yang, and Yinyu Ye. Near-optimal time and sample complexities for solving markov decision processes with a generative model. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 5186–5196. Curran Associates, Inc., 2018.

[6] Alekh Agarwal, Sham Kakade, and Lin F. Yang. Model-based reinforcement learning with a generative model is minimax optimal. *Vol. 125 of Proceedings of Machine Learning Research*, pages 67–83. PMLR, 09–12 Jul 2020.

[7] Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. Breaking the sample size barrier in model-based reinforcement learning with a generative model. *arXiv preprint arXiv:2005.12900*, 2020.

[8] Jiafan He, Dongruo Zhou, and Quanquan Gu. Minimax optimal reinforcement learning for discounted mdps. *CsRR*, abs/2010.00587, 2020.

[9] Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning, 2015.

[10] Omar Darwiche Domingues, Pierre Ménard, Emilie Kaufmann, and Michal Valko. Episodic reinforcement learning in finite mdps: Minimax lower bounds revisited. *CsRR*, abs/2010.03531, 2020.

[11] Aymen Al Marjani and Alexandre Proutiere. Adaptive sampling for best policy identification in markov decision processes, 2020.

[12] Aymen Al Marjani, Aurélien Garivier, and Alexandre Proutiere. Navigating to the best policy in markov decision processes, 2021.

[13] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

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

[15] Jiafan He, Dongruo Zhou, and Quanquan Gu. Logarithmic regret for reinforcement learning with linear function approximation. In *International Conference on Machine Learning*, pages 4171–4180. PMLR, 2021.

[16] 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.

[17] Andrew Wagenmaker and Kevin Jamieson. Instance-dependent near-optimal policy identification in linear mdps via online experiment design. *arXiv preprint arXiv:2207.02575*, 2022.

[18] Andrea Zanette, Alessandro Lazaric, Mykel J Kochenderfer, and Emma Brunskill. Limiting extrapolation in linear approximate value iteration. *Advances in Neural Information Processing Systems*, 32, 2019.

[19] Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pages 6995–7004. PMLR, 2019.

[20] Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning*, pages 5662–5670. PMLR, 2020.

[21] Dongruo Zhou, Jiafan He, and Quanquan Gu. Provably efficient reinforcement learning for discounted mdps with feature mapping. *In Proceedings of Machine Learning Research*, volume 125, pages 67–83. PMLR, 2020.
[22] Jerome Taupin, Yassir Jedra, and Alexandre Proutiere. Best policy identification in linear mdps. Technical report, arxiv, https://arxiv.org/abs/2208.05633, 2022.

[23] Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. Advances in Neural Information Processing Systems, 27, 2014.

[24] Aymen Al Marjani, Aurélien Garivier, and Alexandre Proutiere. Navigating to the best policy in markov decision processes. Advances in Neural Information Processing Systems, 34:25852–25864, 2021.

[25] Yassir Jedra and Alexandre Proutiere. Optimal best-arm identification in linear bandits. Advances in Neural Information Processing Systems, 33:10007–10017, 2020.

[26] Eugene A Feinberg and Adam Shwartz. Handbook of Markov decision processes: methods and applications, volume 40. Springer Science & Business Media, 2012.