European Journal of Control xxx (xxxx) xxx

Contents lists available at ScienceDirect

European Journal of Control

journal homepage: www.elsevier.com/locate/ejcon

Learning parametric policies and transition probability models of markov decision processes from data☆

Tingting Xu[a,1], Henghui Zhu[a,1], Ioannis Ch. Paschalidis[b,*]

[a] Center for Information and Systems Engineering, Boston University, Boston, 02215, United States
[b] Department of Electrical and Computer Engineering, Division of Systems Engineering, and Department of Biomedical Engineering, Boston University, 8 St. Mary's St., Boston, MA 02215

ARTICLE INFO

Article history:
Received 16 November 2019
Revised 25 February 2020
Accepted 18 April 2020
Available online xxx

Recommended by Prof. T Parisini

Keywords:
Policy Learning
Learning Transition Dynamics
Markov Decision Processes
Regularization
Maximum likelihood estimation

ABSTRACT

We consider the problem of estimating the policy and transition probability model of a Markov Decision Process from data (state, action, next state tuples). The transition probability and policy are assumed to be parametric functions of a sparse set of features associated with the tuples. We propose two regularized maximum likelihood estimation algorithms for learning the transition probability model and policy, respectively. An upper bound is established on the regret, which is the difference between the average reward of the estimated policy under the estimated transition probabilities and that of the original unknown policy under the true (unknown) transition probabilities. We provide a sample complexity result showing that we can achieve a low regret with a relatively small amount of training samples. We illustrate the theoretical results with a healthcare example and a robot navigation experiment.

1. Introduction

Markov Decision Processes (MDPs) provide a framework for solving dynamic optimization problems under uncertainty. When the cardinality of the state-action space is small and the transition probabilities are known, one can easily calculate an optimal policy using value iteration or policy iteration. However, in most applications, an MDP has a very large state-action space, and it is not even realistic to assume knowledge of the model, especially the transition probabilities for all state-action pairs.

In an increasing number of settings, it has become possible to collect large amounts of data by observing (state, action, next state) tuples of an agent who uses an unknown policy. In such "data-rich" settings, the problem we consider is to learn the (unknown) original policy used by the agent and also obtain a good estimate of the transition probabilities. Clearly, and because collecting data is cumbersome and expensive, it is of interest to develop efficient learning methods that can handle limited data availability.

The problem we consider has many potential applications. For instance, in the healthcare domain, disease progression can be modeled as an MDP with states corresponding to the condition of the patient and actions associated with physician actions, drugs prescribed, etc. Typically, one does not have access to good models of state transitions and computing a good policy requires *exploration*, which can be done through expensive clinical trials. Yet, there is a wealth of information in *Electronic Health Records (EHRs)*, containing doctors' prescriptions and patients' relevant medical information (demographics, diagnoses, etc). Leveraging this information to learn a prescription policy can help homogenize care across multiple locations and reinforce best practices. In addition, it is also critical to estimate a disease progression model under various treatments. Given a policy and a model one could ask many what-if questions and use a variety of methods to improve the policy.

Another potential example concerns robot navigation, where the robot's position and velocity can be seen as a state and the direction of movement and acceleration as an action. Rather than deriving an optimal navigation policy using traditional methods that are subject to the curse of dimensionality, we can leverage data collected from an expert human operator/driver in order to estimate the unknown policy of the expert.

To achieve the best estimates of transition dynamics and the associated policy in an MDP from data, we design two regularized

☆ Research partially supported by the NSF under grants IIS-1914792, DMS-1664644, and CNS-1645681, by the ONR under grant N00014-19-1-2571, by the NIH under grant 1R01GM135930, by the Boston University Data Science Initiative, and by the BU Center for Information and Systems Engineering.

* Corresponding author. Tel.: +16173530434.
E-mail addresses: tingxu@bu.edu (T. Xu), henghuiz@bu.edu (H. Zhu), yannisp@bu.edu (I.Ch. Paschalidis).
URL: http://sites.bu.edu/paschalidis/ (I.Ch. Paschalidis)

[1] Contributed equally to this manuscript.

https://doi.org/10.1016/j.ejcon.2020.04.003
0947-3580/

logistic regression methods. We assume the policy and transition probabilities are "Boltzmann-type" parametric functions, and the parameters can be derived using maximum likelihood estimation. Our proposed algorithm is shown to achieve a regret of $O(\sqrt{\varepsilon})$ with $\Omega(\log(n)\text{poly}(1/\varepsilon))$ samples, where n indicates the number of adopted features used to represent the policy and $\text{poly}(\cdot)$ denotes a polynomial function. This result implies that by utilizing only a relatively small amount of training data, the estimated policy performs similarly to the unknown policy whose state-action pairs we observe. The sample complexity guarantee is more valuable especially in cases where it is expensive to obtain training samples.

1.1. Related Work

In healthcare applications, using data mining to predict future states of patients has been studied with a Markov chain model [12], deep learning [5], and Bayesian networks [17]. However, these works are not efficient in learning policies. Although some recent works (including [2,4,20]) develop methods to learn actions from data, they formulate the problem as a static classification problem and do not account for the sequential nature of these actions and their impact on future states. There is substantial work on learning MDP policies by observing experts' demonstrations. A survey on robot learning from demonstrations is studied in [1]. The work in [15] learns policies for MDPs in continuous spaces, and [22] conducts strategy learning for non-task-oriented conversational systems. [18] learns an MDP policy from demonstrations in a setting where side information on task requirements is available. Other works consider the problem of learning transition probabilities. In [7], an algorithm for learning the health state transition matrix via wireless body area networks modeled as Partially Observed MDPs (POMDPs) is proposed. Another work, [11], learns the transition functions and obtains an optimal policy through value iteration that uses the learned transition functions. However, these works only validate the efficiency of proposed algorithms through experiments without providing any theoretical guarantees. [21] considers the non-parametric estimation of Markov transition functions and establishes a convergence guarantee. [8] utilizes Poisson regression to estimate general conditional distributions. A number of works [10,14,19] have focused on obtaining an optimal policy while alleviating the sensitivity to uncertainty in the underlying transition probabilities. These methods are not tractable in many MDP applications, since it can take overwhelming computational effort to compute an optimal solution due to Bellman's curse of dimensionality.

More closely related to this work, [9] investigates the question of learning an MDP policy from data, where the explicit MDP model is known. Here, we consider a more general situation where the MDP model is unknown, therefore, both the policy and transition probabilities need to be estimated from data.

1.2. Contributions

To address the problem of estimating both the agent's policy and conditional transition probabilities, we propose two learning models and an estimation algorithm based on regularized logistic regression. The use of regularization is motivated by a recent body of work (see [3,4,16] and references therein) which establish that to render the estimates robust to outliers in the training data one needs to solve properly regularized empirical loss minimization problems.

We characterize the sample complexity of estimating model parameters. We also derive a bound on the regret, which indicates the difference between the average reward of the estimated policy under the estimated transition probabilities and the average reward of the original policy under the true transition dynamics. Compared to our earlier work [9], this paper proposes a learning algorithm that has a solid average reward guarantee even in the case where the MDP dynamics are unknown. A preliminary conference version of this work has appeared in [23]. This paper expands on the preliminary work by establishing a theoretical guarantee on sample complexity and including two large case numerical studies: one related to disease progression and another to robot navigation.

We organize the remaining parts of this paper as follows. In Section 2, we introduce the MDP model and the learning problem formulation. In Section 3, we present the proposed learning algorithm and the corresponding performance metrics. In Section 4, we establish the algorithm's performance on log loss or the distance of the estimated parameters from the original ones. In Section 5, we bound the regret of the estimated policy under the estimated MDP dynamics. In Section 6, we illustrate the theoretical results using two simulation experiments. Conclusions are in Section 7.

Notational conventions: Matrices and vectors are denoted by bold letters; uppercase for matrices and lowercase for vectors. Prime denotes transpose and all vectors are column vectors. For economy of space, we write $\mathbf{x} = (x_1, \ldots, x_n)$ for the column vector $\mathbf{x} \in \mathbb{R}^n$. $\|\mathbf{x}\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ denotes the p-norm of vector $\mathbf{x}$. For a matrix $\mathbf{P}$, $\|\mathbf{P}\|_\infty$ denotes the maximum absolute row sum. Sets are denoted by script letters.

2. Problem Formulation

Consider a finite-state Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, \mathbf{P}, R, \mu)$, where $\mathcal{S}$, $\mathcal{A}$ are the sets of possible states and actions, respectively. For any (state, action, next state) tuple (s, a, q), denote by $\mathrm{P}(q|s,a)$ the transition probability from state s to state q conditional on taking action a. We will use $\mathbf{P}(\cdot|s,a)$ to denote the transition probability vector at state s under action a. The function R represents the one-step reward of the MDP. The function μ is a policy that maps a state to a probability distribution of actions; specifically, $\mu(a|s)$ represents the probability of adopting action a given state s.

For general MDPs with a large state-action space, we use a class of parametric (Boltzmann-type) functions to approximate the policy and conditional transition probabilities:

$$\mathrm{P}_{\boldsymbol{\xi}}(q|s,a) = \frac{\exp\{\boldsymbol{\xi}'\boldsymbol{\psi}(s,a,q)\}}{\sum_{y\in\mathcal{N}_s}\exp\{\boldsymbol{\xi}'\boldsymbol{\psi}(s,a,y)\}}, \tag{1}$$

$$\mu_{\boldsymbol{\theta}}(a|s) = \frac{\exp\{\boldsymbol{\theta}'\boldsymbol{\phi}(s,a)\}}{\sum_{b\in\mathcal{A}}\exp\{\boldsymbol{\theta}'\boldsymbol{\phi}(s,b)\}}, \tag{2}$$

where $\boldsymbol{\xi} \in \mathbb{R}^N$ and $\boldsymbol{\theta} \in \mathbb{R}^n$ are parameters to be learned and $\mathcal{N}_s$ is the set of all feasible next states from the current state s. The kernel functions $\boldsymbol{\psi} : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0,1]^N$ and $\boldsymbol{\phi} : \mathcal{S}\times\mathcal{A} \to [0,1]^n$ transform (state, action, next state) tuples (s, a, q) and (state, action) pairs (s, a) into corresponding desired features, respectively. For the sake of brevity, we will use the term transition probabilities $\boldsymbol{\xi}$ to refer to the conditional transition probabilities induced by the model (1) with parameter $\boldsymbol{\xi}$, and, similarly, we will use the term policy $\boldsymbol{\theta}$ to refer to the policy defined in (2) induced by parameter $\boldsymbol{\theta}$.

We note that the models in (1) and (2) use a given set of feature vector functions $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ that encode important aspects of the raw states and actions. Learning the lower-dimensional models $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$ is certainly easier than learning the entire mappings $\mathrm{P}_{\boldsymbol{\xi}}(q|s,a)$ and $\mu_{\boldsymbol{\theta}}(a|s)$. Features $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ may be designed using intuition about a particular application or obtained more systematically using kernel function ideas (see [9, Sec. IV] and the discussion in Section 6.2).

Given an MDP driven by policy $\boldsymbol{\theta}$ with conditional transition probabilities $\mathbf{P}_{\boldsymbol{\xi}}$, the MDP states form a Markov chain. We

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003

use $\mathbf{M}_{\xi,\theta}$ to represent its transition matrix, where $M_{\xi,\theta}(q|s) = \sum_{a\in\mathcal{A}} \mu_\theta(a|s)\mathrm{P}_\xi(q|s,a)$ for all (state, next state) pairs (s, q). Both the policy and conditional transition probabilities are Boltzmann-type, hence, there is a unique stationary distribution $\pi_{\xi,\theta}(s)$ of the induced Markov chain. This implies a unique stationary distribution of state-action pairs (s, a), denoted by $\eta_{\xi,\theta}(s,a) = \pi_{\xi,\theta}(s)\mu_\theta(a|s)$, based on which we can define the infinite horizon average reward induced by ξ, θ as

$$\overline{R}(\xi,\theta) = \sum_{(a,s)} \eta_{\xi,\theta}(s,a)R(s,a).$$

3. Estimating the Policy and Transition Probabilities

Given an observed collection of states and corresponding actions by an agent, we aim to learn the agent's policy and the corresponding transition probabilities, denoted by θ^* and ξ^*, respectively. Here, we implicitly assume that both the agent's policy and transition probabilities are of the Boltzmann-type (cf. (1) and (2)) but with unknown parameters.

The set of (state, action, next state) tuples driven by policy θ^* with conditional transition probabilities ξ^* is denoted by $\mathcal{X} := \mathcal{X}(\xi^*,\theta^*) = \{(s_i,a_i,q_i);\ i=1,\dots,m\}$. Assume that the state-action pairs $\{(s_i,a_i);\ i=1,\dots,m\}$ are i.i.d. and sampled from the stationary distribution η_{ξ^*,θ^*}. Given state s_i and action a_i, the next state q_i is selected based on the conditional transition probability ξ^*. Therefore, the tuples $\{(s_i, a_i, q_i)\}$ in the data set $\mathcal{X}$ are i.i.d and follow the distribution $\mathcal{D} \sim \mathrm{P}_{\xi^*}(q|s,a)\eta_{\xi^*,\theta^*}(s,a)$. Since usually only a small subset of features are significant in learning appropriate models, we assume θ^* and ξ^* are sparse, having $r<n$ and $q<N$ non-zero elements, respectively. Also, assume that θ^* and ξ^* are bounded by K, element-wise.

Due to the probabilistic nature of (1) and (2), we can adopt *Maximum Likelihood (ML)* estimation with logistic regression to estimate the parameters from data. Given our sparsity assumption on the models θ^* and ξ^*, the distance between two sample points in the feature-label space (ϕ, a) and (ψ, q), respectively, is best measured using the sparse ℓ_∞ norm which accounts for the most dominant feature instead of the entire (and potentially irrelevant) feature vector. This suggests [3], that one needs to use the dual norm, i.e., an ℓ_1 norm, to regularize the ML estimation problem. We will introduce this regularization as a constraint in the formulation. Specifically, we formulate the learning problem of conditional transition probabilities as follows:

$$\max_{\xi\in\Re^N} \sum_{i=1}^m \log \mathrm{P}_\xi(q_i|s_i,a_i) \tag{3}$$
$$\text{s.t.} \quad \|\xi\|_1 \le B_\xi,$$

in which B_ξ adjusts the sparsity of the estimated transition probabilities. Similarly, we formulate the policy learning as follows:

$$\max_{\theta\in\mathcal{R}^n} \sum_{i=1}^m \log \mu_\theta(a_i|s_i) \tag{4}$$
$$\text{s.t} \quad \|\theta\|_1 \le B_\theta,$$

in which B_θ adjusts the sparsity of the estimated policy.

The performance of the ML estimate can be evaluated through a log-loss metric, which is defined as the expected value of the negative log-likelihood over the sample distribution. Specifically, for any parameters ξ and θ, the log loss of the transition probability model and the policy are defined as follows:

$$\epsilon(\xi) = \mathbb{E}_{(s,a,q)\sim\mathcal{D}}[-\log P_\xi(q|s,a)],$$

$$\zeta(\theta) = \mathbb{E}_{(s,a,q)\sim\mathcal{D}}[-\log \mu_\theta(a|s)].$$

The expectation is taken over the distribution $\mathcal{D}$. Since the explicit distribution is unknown, we can only observe (state, action, next state) tuples. Given any data set $\mathcal{X}$, an empirical version of log-losses can be defined as follows:

$$\hat{\epsilon}_\mathcal{X}(\xi) = \frac{1}{m}\sum_{i=1}^m(-\log \mathrm{P}_\xi(q_i|s_i,a_i)),$$

$$\hat{\zeta}_\mathcal{X}(\theta) = \frac{1}{m}\sum_{i=1}^m(-\log \mu_\theta(a_i|s_i)).$$

The empirical log-losses are calculated by the log losses over the training data set $\mathcal{X}(\xi^*,\theta^*)$ and denoted by

$$\hat{\epsilon}(\xi) \triangleq \hat{\epsilon}_{\mathcal{X}(\xi^*,\theta^*)}(\xi), \quad \hat{\zeta}(\theta) \triangleq \hat{\zeta}_{\mathcal{X}(\xi^*,\theta^*)}(\theta).$$

We summarize our algorithms to learn the policy θ^* and transition probability model ξ^* in Algorithm 1.

Algorithm 1 Training algorithm to estimate the policy θ^* (or transition probabilities ξ^*) from samples $\mathcal{X}$.

Initialization: Fix hyper-parameters $C > rK$ $(C > lK)$ and $0 < \gamma < 1$. Randomly split the whole data set $\mathcal{X}$ into two sets $\mathcal{X}_1$ and $\mathcal{X}_2$ with size γm and $(1-\gamma)m$ respectively. Use $\mathcal{X}_1$ and $\mathcal{X}_2$ as training set and cross-validation set, respectively.
Training phase:
for $B = 0, 1, 2, \dots, C$ **do**
solve optimization problem (4) (or (3)) with the training set $\mathcal{X}_1$ and right hand side of the constraint equalto B. Let θ_B (or ξ_B) denote the obtained optimal solution.
end for
Validation phase: Among the obtained solutions θ_B's (or ξ_B's) from the training phase, select the best solution with the lowest "hold-out" error on the validation set $\mathcal{X}_2$, i.e, $\hat{B} = \arg\min_{B\in\{0,1,\dots,C\}} \hat{\zeta}_{\mathcal{X}_2}(\theta_B)$ and set $\hat{\theta} = \theta_{\hat{B}}$ (or $\hat{B} = \arg\min_{B\in\{0,1,\dots,C\}} \hat{\epsilon}_{\mathcal{X}_2}(\xi_B)$ and set $\hat{\xi} = \xi_{\hat{B}}$), where $\hat{\zeta}_{\mathcal{X}_2}(\cdot)$ (or $\hat{\epsilon}_{\mathcal{X}_2}(\cdot)$) denotes the empirical log loss of thepolicy function (transition probabilities) on the validation set $\mathcal{X}_2$.

4. Log-Loss Generalization Guarantees

In this section, we will establish theoretical results on sample complexity, showing that our estimation algorithm (Algorithm 1) is guaranteed to achieve high accuracy with relatively few training samples.

We first relate the difference between log losses of two conditional transition probabilities vectors (policies) to their relative entropy, or Kullback-Leibler (KL) divergence, which characterizes the difference between two distributions. The KL divergence of two conditional transition probability vectors ξ_1 and ξ_2 at (s, a) is defined as:

$$D(\mathbf{P}_{\xi_1}(\cdot|s,a)\|\mathbf{P}_{\xi_2}(\cdot|s,a)) = \sum_q \mathrm{P}_{\xi_1}(q|s,a)\log\frac{\mathrm{P}_{\xi_1}(q|x,a)}{\mathrm{P}_{\xi_2}(q|s,a)}.$$

Similarly, the KL divergence between policies θ_1 and θ_2 at state s is

$$D(\mu_{\theta_1}(\cdot|s)\|\mu_{\theta_2}(\cdot|s)) = \sum_a \mu_{\theta_1}(a|s)\log\frac{\mu_{\theta_1}(a|s)}{\mu_{\theta_2}(a|s)}.$$

Next, we will define the average KL divergence of the policy and transition probability model under some sample distribution. Recall that the stationary distributions of the state and state-action

Markov chains are denoted by $\pi_{\xi,\theta}$ and $\eta_{\xi,\theta}$, respectively. The average KL divergence between policies $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ is defined as follows:

$$D_{\xi,\theta}(\boldsymbol{\mu}_{\theta_1}\|\boldsymbol{\mu}_{\theta_2}) = \sum_s \pi_{\xi,\theta}(s) D(\boldsymbol{\mu}_{\theta_1}(\cdot|s)\|\boldsymbol{\mu}_{\theta_2}(\cdot|s)),$$

and the KL divergence between conditional transition probabilities $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ is defined as

$$D_{\xi,\theta}(\mathbf{P}_{\xi_1}\|\mathbf{P}_{\xi_2}) = \sum_{s,a} \eta_{\xi,\theta}(s,a) D\big(\mathbf{P}_{\xi_1}(\cdot|s,a)\|\mathbf{P}_{\xi_2}(\cdot|s,a)\big).$$

According to [9], the difference between log losses of two policies is equal to their Kullback-Leibler (KL) divergence.

Lemma 4.1 ([9]). *Let $\hat{\boldsymbol{\theta}}$ be an estimate of the policy $\boldsymbol{\theta}$, then*

$$\zeta(\hat{\boldsymbol{\theta}}) - \zeta(\boldsymbol{\theta}) = D_{\xi,\theta}(\boldsymbol{\mu}_{\boldsymbol{\theta}}\|\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}).$$

Similarly, we can prove that the difference between log losses of two transition probability models are equal to their KL divergence. The proof is similar to that for Lemma 4.1 and hence omitted.

Corollary 4.2. *Let $\hat{\boldsymbol{\xi}}$ be an estimate of the policy $\boldsymbol{\xi}$, then*

$$\epsilon(\hat{\boldsymbol{\xi}}) - \epsilon(\boldsymbol{\xi}) = D_{\xi,\theta}(\mathbf{P}_{\boldsymbol{\xi}}\|\mathbf{P}_{\hat{\boldsymbol{\xi}}}).$$

In [9], we have obtained a bound of the difference between log losses of the policy $\boldsymbol{\mu}_{\boldsymbol{\theta}^*}$ and the estimated policy $\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}$ as follows.

Theorem 4.3 ([9]). *Given any positive values $\varepsilon > 0$ and $\delta > 0$, if the sample size satisfies the following condition*

$$m = \Omega\Big((\log n)\cdot \mathrm{poly}(r,\ K,\ C,\ H,\ \log(1/\delta),\ 1/\varepsilon)\Big),$$

where H is the maximum number of feasible actions at a state of the MDP, then with probability at least $1-\delta$,

$$|\zeta(\hat{\boldsymbol{\theta}}) - \zeta(\boldsymbol{\theta})| = D_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\boldsymbol{\mu}_{\boldsymbol{\theta}^*}\|\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}) \le \varepsilon.$$

This implies that $\hat{\boldsymbol{\theta}}$ obtained by Algorithm 1 can be arbitrarily close to the unknown policy $\boldsymbol{\theta}^*$. The function poly(s) denotes a polynomial function with respect to elements of s. Specifically, $m = \Omega(H^3)$ in terms of only H.

Similarly, in the following corollary, we bound the difference between the log losses of the original and estimated conditional transition probability, $\mathbf{P}_{\boldsymbol{\xi}^*}$ and $\mathbf{P}_{\hat{\boldsymbol{\xi}}}$.

Corollary 4.4. *Given any positive values $\varepsilon > 0$ and $\delta > 0$, if the sample size satisfies the following condition*

$$m = \Omega\Big((\log N)\cdot \mathrm{poly}(l,\ K,\ C,\ M,\ \log(1/\delta),\ 1/\varepsilon)\Big),$$

where $M = \max_s |\mathcal{N}_s|$, then with probability at least $1-\delta$,

$$|\epsilon(\hat{\boldsymbol{\xi}}) - \epsilon(\boldsymbol{\xi})| = D_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\mathbf{P}_{\boldsymbol{\xi}^*}\|\mathbf{P}_{\hat{\boldsymbol{\xi}}}) \le \varepsilon.$$

This implies that $\hat{\boldsymbol{\xi}}$ obtained by Algorithm 1 can be arbitrarily close to the original transition probability $\boldsymbol{\xi}^*$, with $m = \Omega(M^3)$ in terms of only M. The proof is similar to that for Theorem 4.3 and hence omitted.

5. Bounds on Regret

With the estimated conditional transition probabilities from Section 3, we are able to compute the average reward of the estimated policy by simulating the MDP. A natural question is how good this simulated average reward is, compared to the average reward of the original policy under the true MDP dynamics.

In this section, we develop a bound on the regret using the estimated policy as opposed to the original policy. Specifically, we define the regret $\mathrm{Reg}(\mathcal{X})$ as

$$\mathrm{Reg}(\mathcal{X}) = \bar{R}(\boldsymbol{\xi}^*,\boldsymbol{\theta}^*) - \bar{R}(\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}),$$

in which $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\xi}}$ are the estimated policy and estimated conditional transition probabilities from the samples $\mathcal{X}$, respectively. We need the following preliminary definitions and lemmata.

Definition 1 ([9]). Given a Markov chain with transition probability matrix $\mathbf{M}_{\xi,\boldsymbol{\theta}}$ induced by policy $\boldsymbol{\theta}$ and conditional transition probability parameter $\boldsymbol{\xi}$, the fundamental matrix of the Markov chain is defined as

$$\mathbf{Z}_{\xi,\boldsymbol{\theta}} = (\mathbf{A}_{\xi,\boldsymbol{\theta}} + \mathbf{e}\boldsymbol{\pi}'_{\xi,\boldsymbol{\theta}})^{-1},$$

where $\mathbf{A}_{\xi,\boldsymbol{\theta}} = \mathbf{I} - \mathbf{M}_{\xi,\boldsymbol{\theta}}$, $\boldsymbol{\pi}_{\xi,\boldsymbol{\theta}}$ represents the stationary distribution associated with $\mathbf{M}_{\xi,\boldsymbol{\theta}}$, and $\mathbf{e}$ is the vector with all elements equal to 1.

Definition 2 ([9]). The group inverse of a square matrix $\mathbf{A}$, denoted as $\mathbf{A}^{\#}$, is the unique square matrix satisfying the following conditions

$$\mathbf{A}\mathbf{A}^{\#}\mathbf{A} = \mathbf{A},\ \mathbf{A}^{\#}\mathbf{A}\mathbf{A}^{\#} = \mathbf{A},\ \mathbf{A}\mathbf{A}^{\#} = \mathbf{A}^{\#}\mathbf{A}.$$

Definition 3 ([9]). Given a matrix $\mathbf{B}$ with equal row sums, its ergodic coefficient is defined as

$$\tau(\mathbf{B}) = \sup_{\mathbf{v}'\mathbf{e}=0;\|\mathbf{v}\|_1=1} \|\mathbf{v}'\mathbf{B}\|_1 = \frac{1}{2}\max_{i,j}\sum_s |b_{is} - b_{js}|. \tag{5}$$

Lemma 5.1. *([6, Lemma 11.6.1]) Consider any two probability vectors $\mathbf{p}_1,\ \mathbf{p}_2 \in \mathbb{R}^n$. It holds that*

$$D(\mathbf{p}_1\|\mathbf{p}_2) \ge \frac{1}{2\ln 2}\|\mathbf{p}_1 - \mathbf{p}_2\|_1^2. \tag{6}$$

Lemma 5.2 ([9]). *Given two stochastic matrices $\mathbf{P}_1$ and $\mathbf{P}_2$ (and associated fundamental matrices $\mathbf{Z}_i$ and matrices $\mathbf{A}_i = \mathbf{I} - \mathbf{P}_i,\ i = 1, 2$), assume $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ are the corresponding unique stationary distributions, respectively. Let $\mathbf{E} = \mathbf{P}_1 - \mathbf{P}_2$, then*

$$\|\boldsymbol{\pi}_1 - \boldsymbol{\pi}_2\|_1 \le \kappa\|\boldsymbol{\pi}'_1\mathbf{E}\|_1, \tag{7}$$

where κ is a constant that can take one of the following values: $\kappa = \|\mathbf{Z}_2\|_\infty$, or $\kappa = \|\mathbf{A}_2^{\#}\|_\infty$, or $\kappa = 1/(1-\tau(\mathbf{P}_2))$, or $\kappa = \tau(\mathbf{Z}_2) = \tau(\mathbf{A}_2^{\#})$.

Now we are ready to bound the regret of the estimated policy using the above preliminary definitions and results.

Theorem 5.3. *Given any positive values $\varepsilon > 0$, $\delta > 0$, assume Algorithm 1 adopts*

$$\Omega((\log(\max(N,n)))\cdot \mathrm{poly}(r, l, K, C, M, \log(1/\delta),\ 1/\varepsilon, H)$$

i.i.d. training samples to learn parameters $\hat{\boldsymbol{\xi}}$ and $\hat{\boldsymbol{\theta}}$. Then, with probability of at least $1-\delta$, the following result holds

$$|\bar{R}(\boldsymbol{\xi}^*,\boldsymbol{\theta}^*) - \bar{R}(\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}})| \le 2\sqrt{\ln 2\varepsilon}R_{\max}(1+2\kappa),$$

in which $R_{\max} = \max_{(s,a)\in\mathcal{S}\times\mathcal{A}}|R(s,a)|$ and κ is a constant value that depends on the transition probability matrix $\mathbf{M}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}$ (and its corresponding fundamental matrix $\mathbf{Z}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}$ and matrix $\mathbf{A}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}} = \mathbf{I} - \mathbf{M}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}$). This constant κ can be any of the following: $\kappa = \|\mathbf{Z}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}\|_\infty$, or $\kappa = \|\mathbf{A}^{\#}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}\|_\infty$, or $\kappa = 1/(1-\tau(\mathbf{M}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}))$, or $\kappa = \tau(\mathbf{Z}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}) = \tau(\mathbf{A}^{\#}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}})$.

Proof. We will first express the regret as the sum of two parts, and then bound each part separately.

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003

$$
\begin{aligned}
&\mathrm{Reg}(\mathcal{X}) = \bar{R}(\boldsymbol{\xi}^*, \boldsymbol{\theta}^*) - \bar{R}(\hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\theta}}) \\
&= \sum_s \sum_a [\eta_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s,a) - \eta_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s,a)] R(s,a) \\
&= \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a \mu_{\boldsymbol{\theta}^*}(a|s) R(s,a) \\
&\quad - \sum_s \pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a) \\
&= \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a [\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)] R(s,a) \\
&\quad - \sum_s [\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)] \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a) \\
&\le \left| \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a [\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)] R(s,a) \right| \\
&\quad + \left| \sum_s [\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)] \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a) \right|. \qquad (8)
\end{aligned}
$$

Note that the first term in equation (8) depends on the value $\sum_a [\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)]$, which is the learning error in estimating the policy. The second term depends on $\sum_s |\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)|$, which is the perturbation error of the stationary distribution of the Markov chain by applying the estimated policy $\hat{\boldsymbol{\theta}}$. In the following, we will bound the two terms separately, and begin with the first term.

$$
\begin{aligned}
&\left| \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a [\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)] R(s,a) \right| \\
&\le \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a |(\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s))| \cdot |R(s,a)| \\
&\le R_{\max} \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a |(\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s))| \\
&= R_{\max} \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \|(\boldsymbol{\mu}_{\boldsymbol{\theta}^*}(\cdot|s) - \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\cdot|s))\|_1. \qquad (9)
\end{aligned}
$$

The bound in (9) depends on the difference between the log-loss of the estimated policy $\hat{\boldsymbol{\theta}}$ and that of target policy $\boldsymbol{\theta}^*$. Based on Lemma 5.1, we have

$$
\begin{aligned}
&\left| \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a [\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)] R(s,a) \right| \\
&\le R_{\max} \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sqrt{2(\ln 2) D\big(\boldsymbol{\mu}_{\boldsymbol{\theta}^*}(\cdot|s) \| \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\cdot|s)\big)} \\
&\le \sqrt{2\ln 2} R_{\max} \cdot \sqrt{\sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) D\big(\boldsymbol{\mu}_{\boldsymbol{\theta}^*}(\cdot|s) \| \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\cdot|s)\big)} \\
&= \sqrt{2\ln 2} R_{\max} \sqrt{D_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\boldsymbol{\mu}_{\boldsymbol{\theta}^*} \| \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}})} \\
&\le \sqrt{2(\ln 2)\varepsilon} R_{\max}. \qquad (10)
\end{aligned}
$$

The first inequality holds by applying Lemma 5.1 with $\mathbf{p}_1 = \boldsymbol{\mu}_{\boldsymbol{\theta}^*}(\cdot|s)$ and $\mathbf{p}_2 = \boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}(\cdot|s)$ for each state s. The second inequality is obtained by applying Jensen's inequality. The final inequality holds using Theorem 4.3.

Next, we will bound the second term in (8).

$$
\begin{aligned}
&\left| \sum_s (\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)) \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a) \right| \\
&\le \sum_s |\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)| \sum_a |\mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a)| \\
&\le R_{\max} \sum_s |\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)| \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) \\
&= R_{\max} \sum_s |\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)| \qquad (11) \\
&\le R_{\max} \kappa \|\boldsymbol{\pi}'_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\mathbf{M}_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*} - \mathbf{M}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}})\|_1. \qquad (12)
\end{aligned}
$$

The equality (11) can be obtained since $\sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) = 1$ for all s, and (12) holds by using Lemma 5.2.

For the last term in equation (12), by applying the definition of the transition probability $\mathbf{M}_{\boldsymbol{\xi},\boldsymbol{\theta}}$ associated with parameters $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$, we obtain

$$
\begin{aligned}
&\|\boldsymbol{\pi}'_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\mathbf{M}_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*} - \mathbf{M}_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}})\|_1 \\
&= \sum_q \Big| \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \cdot \\
&\quad \sum_a [\mathrm{P}_{\boldsymbol{\xi}^*}(q|s,a) \mu_{\boldsymbol{\theta}^*}(a|s) - \mathrm{P}_{\hat{\boldsymbol{\xi}}}(q|s,a) \mu_{\hat{\boldsymbol{\theta}}}(a|s)] \Big| \\
&\le \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_q \sum_a \Big| \mathrm{P}_{\hat{\boldsymbol{\xi}}}(q|s,a) \big[\mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s)\big] \Big| \\
&\quad + \sum_s \sum_a \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \mu_{\boldsymbol{\theta}^*}(a|s) \sum_q \Big| \mathrm{P}_{\hat{\boldsymbol{\xi}}}(q|s,a) - \mathrm{P}_{\boldsymbol{\xi}^*}(q|s,a) \Big| \\
&\le \sum_s \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s) \sum_a \big| \mu_{\boldsymbol{\theta}^*}(a|s) - \mu_{\hat{\boldsymbol{\theta}}}(a|s) \big| \\
&\quad + \sum_{s,a} \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s,a) \sum_q \Big| \mathrm{P}_{\hat{\boldsymbol{\xi}}}(q|s,a) - \mathrm{P}_{\boldsymbol{\xi}^*}(q|s,a) \Big|,
\end{aligned}
$$

in which the last inequality holds by using $\sum_q \mathrm{P}(q|s,a) = 1$ for all (s,a). Now, using inequality (), Theorem 4.3 and Corollary 4.4, we are able to bound (9) as

$$
\begin{aligned}
&\left| \sum_s (\pi_{\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\theta}}}(s) - \pi_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(s)) \sum_a \mu_{\hat{\boldsymbol{\theta}}}(a|s) R(s,a) \right| \\
&\le 2\sqrt{2\varepsilon \ln 2} \kappa R_{\max}. \qquad (13)
\end{aligned}
$$

Finally, by combining (10), (13) and (8), we obtain the result of Theorem 5.3. □

Note that in Theorem 5.3, the regret relies on the constant value κ, which is referred to as the condition number of the estimated policy under the estimated transition dynamics. The regret monotonically increases with the condition number.

6. Experimental Results

In this section, we illustrate the efficiency of our algorithms using two examples: a disease progression experiment where we seek to learn the prescription policy reflected in the data, and a robot navigation experiment where we learn the robot navigation policy on a 2-D grid.

6.1. Experimental Settings

6.1.1. A Disease Progression Example

We consider patients with a chronic disease and design an MDP to model the effect of drugs. We denote the state of the MDP by $\mathbf{s} = (s_1, s_2)$, where $s_1, s_2 \in \{0, 1, \ldots, 20\}$. Here, s_1 denotes the severity of the disease itself, and s_2 the severity of the comorbidities that the patient may face. The actions in the MDP represent the various treatments for the patient. Assume there are two drug types with different treatment effects, one for the main symptoms of the disease, and another for the comorbidities. The actions can be denoted by $\mathbf{a} = (a_1, a_2)$, where $a_i \in \{0, 1\}$ is a indicator of whether the patient takes the i-th type of drug or not.

We assume that the disease state can only transition from the current state to neighboring states; specifically, the state difference $\mathbf{z}_t = \mathbf{s}_{t+1} - \mathbf{s}_t$ at time t should satisfy $\|\mathbf{z}_t\|_1 \le 1$ for all $\mathbf{s}_t$, $\mathbf{s}_{t+1}$ and $\mathbf{a}_t$. Furthermore, we assume that the transition probability $\mathrm{P}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ depends only on the state difference $\mathbf{z}_t = \mathbf{s}_{t+1} - \mathbf{s}_t$

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003

Table 1
Disease state transition probabilities conditioned on actions.

State Diff $\mathbf{z}_t =$	(0, 0)	(0, 1)	(0, −1)	(−1, 0)	(1, 0)
$\mathbf{a}_t = (0, 0)$	0.7	0.1	0.05	0.05	0.1
$\mathbf{a}_t = (1, 0)$	0.4	0.1	0.05	0.35	0.1
$\mathbf{a}_t = (0, 1)$	0.4	0.1	0.35	0.05	0.1
$\mathbf{a}_t = (1, 1)$	0.2	0.1	0.3	0.3	0.1

Table 2
Robot transition probability conditioned on actions.

State Diff $\mathbf{z}_t$	(0, 0)	(0, 1)	(0, −1)	(−1, 0)	(1, 0)
$\mathbf{a}_t =$ "down"	0.05	0.05	0.8	0.05	0.05
$\mathbf{a}_t =$ "right"	0.05	0.05	0.05	0.05	0.8
$\mathbf{a}_t =$ "up"	0.05	0.8	0.05	0.05	0.05
$\mathbf{a}_t =$ "left"	0.05	0.05	0.05	0.8	0.05

and action $\mathbf{a}_t$. The transition probability matrix (conditioned on all actions) is shown in Table 1. The MDP is also assumed to have a bouncing boundary, i.e., when the patient takes an action leading the state outside of the boundary, the transition is bounced back without changing the current state.

Consistent with Table 1, if the patient takes none of the two treatments, i.e., $\mathbf{a}_t = (0, 0)$, both disease and comorbidity indicators s_1 and s_2 will tend to stay the same or become more severe. The disease severity decreases if the patient uses the type-1 drug, i.e., $\mathbf{a}_t = (1, 0)$. The comorbidity symptoms are relieved when the patient takes type-2 drug, i.e., $\mathbf{a}_t = (0, 1)$. Because of the drug interactions, if both drugs are used together, i.e., $\mathbf{a}_t = (1, 1)$, they have diminished effects compared to being individually used.

When a patient enters a disease state, he/she can collect the immediate reward associated at the state. Assume doctors (experts) use a policy to optimize the long-term expected average reward. We wish to learn the prescription policy and disease transition probabilities from observed tuples of (state, action, next state).

Assume that the best state (0, 0) is associated with reward R_0, and the associated reward can "spread" according to a Gaussian distribution, i.e., immediate rewards are defined as follows:

$$R(\mathbf{s}, \mathbf{a}) = R(\mathbf{s}) = R_0 \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-\|\mathbf{s}\|^2}{2\sigma^2}},$$

for all actions $\mathbf{a}$. The parameter σ controls the discounting rate of reward being decreased as the disease state becomes worse. The parameters are tunable to achieve desirable behaviors [13]. In this experiment, we set $R_0 = 30$ and $\sigma = 10$.

6.1.2. A Robot Navigation Example

We also design a robot navigation experiment on a 2-D grid. The robot's navigation trajectory is driven by an MDP, where the state is the current location of the robot, denoted by $\mathbf{s} = (s_1, s_2)$, where $s_1, s_2 \in \{0, 1, \ldots, 20\}$. The robot has 4 feasible actions corresponding to its direction of movement, and the action set at each state is {"up", "down", "left", "right"}.

We assume that the robot can only move from a location to a neighboring location; specifically, the state difference $\mathbf{z}_t = \mathbf{s}_{t+1} - \mathbf{s}_t$ at time t should satisfy $\|\mathbf{z}_t\|_1 \leq 1$ for all $\mathbf{s}_t$, $\mathbf{s}_{t+1}$ and $\mathbf{a}_t$. Furthermore, the transition probability $\mathrm{P}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ depends only on the state difference $\mathbf{z}_t = \mathbf{s}_{t+1} - \mathbf{s}_t$ and action $\mathbf{a}_t$. The transition probability matrix (conditioned on all actions) is shown in Table 2. We assume that the robot's transition is subject to uncertainty; in particular, one action may lead the robot to its neighboring grid points instead of the intended direction. For instance, when the robot takes action "down", it will shift to an adjacent grid position below (state difference $(0, -1)$) with probability 0.8, and to all other four neighboring positions on the grid with probability 0.05, respectively. The MDP is also assumed to have a bouncing boundary, i.e., when the robot takes an action leading it outside the boundary, it is bounced back in the opposite direction without changing its current position.

The robot collects immediate reward at each grid point and the ultimate goal is to maximize the long-term expected average reward. In this example, given historical trajectories of identical robots driven by the same policy, we are interested in learning the policy and the unknown transition probabilities from observed (state, action, next state) tuples. We associate source rewards near the four grid corners as specified below. Grid points (1, 1) and (19, 19) each have an associated source reward equal to 10, whereas grid points (1, 19) and (19, 1) each have an associated source reward equal to 1. Assume that the source rewards "spread" according to a Gaussian distribution. The immediate reward at each grid point is the superposition of rewards spread from each source to this point. Specifically, the immediate reward for state $\mathbf{s}$ is:

$$R(\mathbf{s}, \mathbf{a}) = R(\mathbf{s}) = \sum_{\nu} R_\nu \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-\|\mathbf{s}-\mathbf{s}_\nu\|^2}{2\sigma^2}},$$

for all $\mathbf{a}$ and $\mathbf{s}$, where R_ν specifies the source reward at source $\mathbf{s}_\nu$ and σ adjusts the spread rate of the source rewards. We can tune such a reward function for desirable behavior [13]. In our experiments, $\sigma = 1$.

6.2. Policy and Transition Probability Learning

Designing appropriate features is essential for our learning algorithm. Following the approach of [9, Sec. IV], the features are based on a set of representative states. Regarding transition probabilities, we assume

$$\mathrm{P}_{\boldsymbol{\xi}}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = \frac{\exp\{\boldsymbol{\xi}'\boldsymbol{\psi}(\mathbf{s}_{t+1} - \mathbf{s}_t, \mathbf{a}_t)\}}{\sum_{\mathbf{s}\in\mathcal{S}} \exp\{\boldsymbol{\xi}'\boldsymbol{\psi}(\mathbf{s} - \mathbf{s}_t, \mathbf{a}_t)\}}.$$

The feature vector is constructed as follows. We select a set of p representative state differences $\mathcal{R} = \{(i_k, j_k) : i_k, j_k \in \{-1, 0, 1\},\ k = 1, \ldots, p\}$ and define feature functions:

$$\psi_{i_k,j_k}(\mathbf{z}, \mathbf{a}, \mathbf{b}) = \begin{cases} \frac{1}{\sqrt{2\pi}} e^{\frac{-\|\mathbf{z}-(i_k,j_k)\|^2}{2}}, & \text{if } \mathbf{b} = \mathbf{a}, \\ 0, & \text{otherwise}, \end{cases}$$

for $k = 1, \ldots, p$ and $\mathbf{b} \in \mathcal{A}$. Then, $\boldsymbol{\psi}(\mathbf{z}, \mathbf{a}) = (\psi_{i,j}(\mathbf{z}, \mathbf{a}, \mathbf{b});\ \forall (i, j) \in \mathcal{R},\ \forall \mathbf{b} \in \mathcal{A})$.

Similarly, assume that the original policy has the following form

$$\mu_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s}) = \frac{\exp\{\boldsymbol{\theta}'\boldsymbol{\phi}(\mathbf{s}, \mathbf{a})\}}{\sum_{\mathbf{b}\in\mathcal{A}} \exp\{\boldsymbol{\theta}'\boldsymbol{\phi}(\mathbf{s}, \mathbf{b})\}}.$$

In this case, the feature vector is constructed by selecting a set $\mathcal{B}$ of representative states from the state space and defining features

$$\phi_{\mathbf{u}}(\mathbf{s}, \mathbf{a}) = \sum_{\mathbf{q}} \mathrm{P}_{\boldsymbol{\xi}}(\mathbf{q}|\mathbf{s}, \mathbf{a}) f_{\mathbf{u}}(\mathbf{q}), \qquad \mathbf{u} \in \mathcal{B},$$

where

$$f_{\mathbf{u}}(\mathbf{q}) = \frac{1}{\sqrt{2\pi}} e^{\frac{-\|\mathbf{u}-\mathbf{q}\|^2}{2}}.$$

Then, the feature vector is $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a}) = (\phi_{\mathbf{u}}(\mathbf{s}, \mathbf{a});\ \forall \mathbf{u} \in \mathcal{B})$.

6.3. Performance Evaluation

For the two applications discussed in Sections 6.1.1 and 6.1.2, we solve the corresponding MDPs using value iteration and obtain an optimal policy. We use this policy to generate (state, action, next state) tuples. Given the observed tuples, we will estimate a policy $\mu_{\boldsymbol{\theta}}(\mathbf{a}_t|\mathbf{s}_t)$ parameterized by $\boldsymbol{\theta}$ and an induced transition probability model $\mathrm{P}_{\boldsymbol{\xi}}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, parameterized by $\boldsymbol{\xi}$ using the feature functions described in Section 6.2.

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003

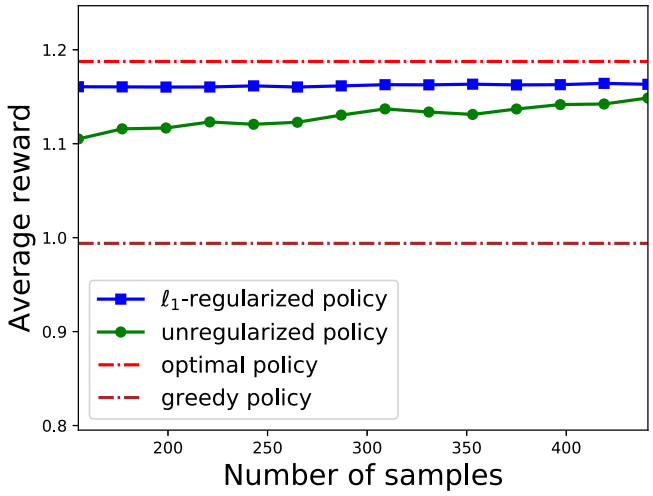

Fig. 1. Disease progression example: average rewards v.s. sample size.

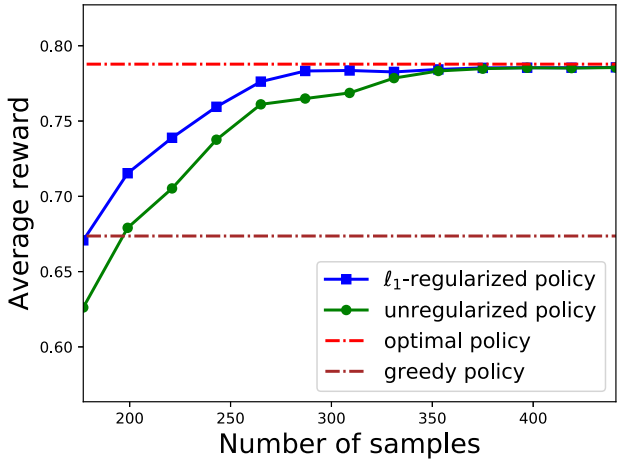

Fig. 2. Robot navigation example: average rewards v.s. sample size.

We compare the average rewards of the following four policies:

1. Optimal policy: obtained from the value iteration algorithm and used to generate (state, action, next state) tuples.
2. ℓ_1-regularized policy: trained using Algorithm 1 using ℓ_1-regularized logistic regression.
3. Unregularized policy: trained using Algorithm 1 to solve the logistic regression problems without regularization.
4. Greedy policy: at each state taking an action which maximizes the expected immediate reward.

To investigate how training sample size influences the performance of our proposed algorithm, we implemented Algorithm 1 using different sample sizes m. For each sample size, we conduct 10 independent runs, obtain a policy from each run, and then simulate this policy under the true MDP dynamics to assess its performance. We average over the 10 runs the per-run average reward obtained. These averages are plotted Figs. 1 and 2 for the healthcare and robot applications, respectively.

We can see that the ℓ_1-regularized policy achieves reward that is closer to the reward achieved by the unknown original policy, which is consistent with Theorem 5.3. In addition, the ℓ_1-regularized policy consistently outperforms the greedy and unregularized policies, especially in the case of small sample sizes. This illustrates the benefits of regularization and is consistent with our related comments in the Introduction.

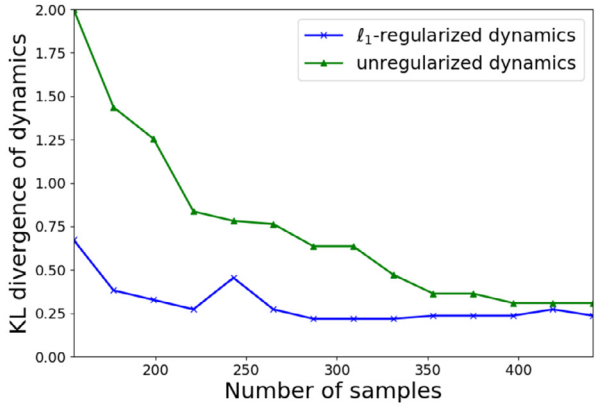

Fig. 3. Disease progression example: average KL divergence between the true and estimated conditional transition probabilities.

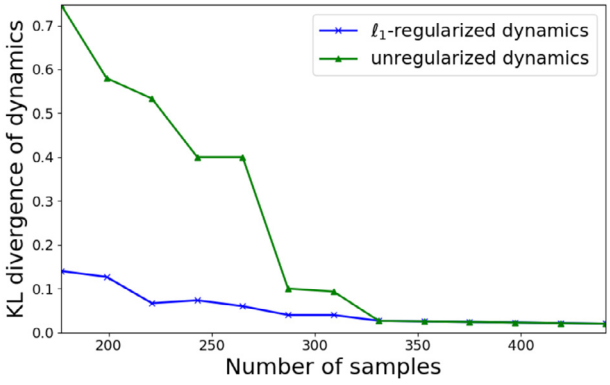

Fig. 4. Robot navigation example: average KL divergence between true and estimated conditional transition probabilities.

In addition, we compare two different estimates of the conditional transition probabilities as follows:

1. an ℓ_1-regularized estimate, where we use Algorithm 1 with the ℓ_1-regularized logistic regression to learn the conditional transition probability model; and
2. an unregularized estimate, where we use Algorithm 1 with unregularized logistic regression to learn the conditional transition probability model.

Specifically, we computed the KL-divergence between the true and estimated conditional transition probabilities $D_{\boldsymbol{\xi}^*,\boldsymbol{\theta}^*}(\mathbf{P}_{\boldsymbol{\xi}^*} \| \mathbf{P}_{\hat{\boldsymbol{\xi}}}) = \epsilon(\hat{\boldsymbol{\xi}}) - \epsilon(\boldsymbol{\xi}^*)$ as a function of sample size. The KL-divergence results for the healthcare and robot navigation examples are shown in Figs. 3 and 4, respectively. The lower KL-divergence is, the better the estimate is. Through all different sample sizes in the two examples, the ℓ_1-regularized estimate consistently outperforms its unregularized alternative, and more so in the case of small sample sizes. In addition, the regularization algorithm achieves accurate estimates with only a very small number of samples, which validates Corollary 4.4.

Note that even though the original policy and conditional transition probabilities do not follow the parametric forms in Eqs. (1) and (2), our algorithm is still able to achieve high accuracy.

7. Conclusion

This article investigates the problem of learning a policy and an associated transition probability model in an MDP based on observed (state, action, next state) tuples. We propose two

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003

regularized logistic regression models to estimate the true transition probabilities and original policy. We establish out-of-sample generalization bounds on log-loss for the policy and transition probability estimation. Further, we derive an upper bound on the regret of the estimated policy under the learned transition probability model. The theoretical results are validated in two different applications, one involving disease progression and the other robot navigation. The numerical results show satisfactory learning of both the policy and transition dynamics and illustrate the benefits of using robust learning techniques.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] B.D. Argall, S. Chernova, M. Veloso, B. Browning, A survey of robot learning from demonstration, Robotics and autonomous systems 57 (5) (2009) 469–483.
[2] D. Bertsimas, N. Kallus, A.M. Weinstein, Y.D. Zhuo, Personalized diabetes management using electronic medical records, Diabetes care 40 (2) (2017) 210–217.
[3] R. Chen, I.C. Paschalidis, A robust learning approach for regression models based on distributionally robust optimization, Journal of Machine Learning Research 19 (13) (2018).
[4] R. Chen, I.C. Paschalidis, Selecting optimal decisions via distributionally robust nearest-neighbor regression, in: H. Wallach, H. Larochelle, A. Beygelzimer, F.d. Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32 (NeurIPS), pages 748–758, Vancouver, Canada, December, Curran Associates, Inc., 2019.
[5] E. Choi, M.T. Bahadori, A. Schuetz, W.F. Stewart, J. Sun, Doctor AI: Predicting clinical events via recurrent neural networks, in: Machine Learning for Healthcare Conference, 2016, pp. 301–318.
[6] T.A. Cover, J.A. Thomas, Elements of Information Theory, John Wiley & Sons, 2006.
[7] T. Geller, Y.B. David, E. Khmelnitsky, I. Ben-Gal, A. Ward, D. Miller, N. Bambos, Learning health state transition probabilities via wireless body area networks, in: ICC 2019-2019 IEEE International Conference on Communications (ICC), pages 1–6, IEEE, 2019.
[8] L. Györfi, M. Kohler, Nonparametric estimation of conditional distributions, Information Theory, IEEE Transactions on 53 (06) (2007) 1872–1879.
[9] M.K. Hanawal, H. Liu, H. Zhu, I.C. Paschalidis, Learning policies for Markov decision processes from data, IEEE Transactions on Automatic Control 64 (6) (2019) 2298–2309.
[10] G. Iyengar, Robust dynamic programming, Math. Operations Research 30 (2) (2005) 1–21.
[11] D. Kent, S. Banerjee, S. Chernova, Learning sequential decision tasks for robot manipulation with abstract Markov decision processes and demonstration-guided exploration, in: 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pages 1–8, IEEE, 2018.
[12] Y.-Y. Liu, H. Ishikawa, M. Chen, G. Wollstein, J.S. Schuman, J.M. Rehg, Longitudinal modeling of glaucoma progression using 2-dimensional continuous-time hidden Markov model, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2013, pp. 444–451.
[13] L. Matignon, G.J. Laurent, N.L. Fort-Piat, Reward function and initial values: better choices for accelerated goal-directed reinforcement learning, in: International Conference on Artificial Neural Networks, pages 840–849, Springer, 2006.
[14] A. Nilim, L.E. Ghaoui, Robust solutions to Markov decision problems with uncertain transition matrices, Operations Research 53 (5) (2005) 780–798.
[15] S. Paternain, J.A. Bazerque, A. Small, A. Ribeiro, Learning policies for Markov decision processes in continuous spaces, in: 2018 IEEE Conference on Decision and Control (CDC), pages 4751–4758, IEEE, 2018.
[16] S. Shafieezadeh-Abadeh, P.M. Esfahani, D. Kuhn, Distributionally robust logistic regression, in: Advances in Neural Information Processing Systems, pages 1576–1584, 2015.
[17] J. Weiss, S. Natarajan, D. Page, Multiplicative forests for continuous-time processes, in: Advances in Neural Information Processing Systems, 2012, pp. 458–466.
[18] M. Wen, I. Papusha, U. Topcu, Learning from demonstrations with high-level side information, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017.
[19] W. Wiesemann, D. Kuhn, B. Rustem, Robust Markov decision processes, Mathematics of Operations Research 38 (1) (2013) 153–183.
[20] T. Xu, I.C. Paschalidis, Learning models for writing better doctor prescriptions, in: Proceedings of the European Control Conference pages, 2454–2459, 2019. Naples, Italy, June 25–28
[21] S. Yakowitz, Nonparametric estimation of Markov transition functions, Ann. Statist. 7 (3) (1979) 671–679. 05
[22] Z. Yu, Z. Xu, A.W. Black, A. Rudnicky, Strategy and policy learning for non–task-oriented conversational systems, in: Proceedings of the 17th annual meeting of the special interest group on discourse and dialogue, pages 404–412, 2016.
[23] H. Zhu, T. Xu, I.C. Paschalidis, Learning parameterized prescription policies and disease progression dynamics using Markov decision processes, in: American Control Conference, pages 3438–3443, Philadelphia, Pennsylvania, July 10–12, 2019.

Please cite this article as: T. Xu, H. Zhu and I.Ch. Paschalidis, Learning parametric policies and transition probability models of markov decision processes from data, European Journal of Control, https://doi.org/10.1016/j.ejcon.2020.04.003