A Data-Driven Policy Iteration Scheme based on Linear Programming

Goran Banjac and John Lygeros

***Abstract*— We consider the problem of learning discounted-cost optimal control policies for unknown deterministic discrete-time systems with continuous state and action spaces. We show that a policy evaluation step of the well-known policy iteration (PI) algorithm can be characterized as a solution to an infinite dimensional linear program (LP). However, when approximating such an LP with a finite dimensional program, the PI algorithm loses its nominal properties. We propose a data-driven PI scheme that ensures a certain monotonic behavior and allows for incorporation of expert knowledge on the system. A numerical example illustrates effectiveness of the proposed algorithm.**

I. Introduction

Optimal control problems are important tools of mathematical modeling that arise in many research areas. The dynamic programming (DP) techniques that have been developed for solving such problems rely on a comprehensive theoretical framework, and include methods such as value iteration (VI), policy iteration (PI), and the linear programming approach [1]–[4]. However, finding an exact solution to such problems is often computationally intractable, which motivates the development of approximation schemes, known collectively as approximate dynamic programming (ADP).

In many real-world control applications the dynamics of the system is unknown, and one needs to learn an optimal control policy from information obtained by interacting with the system. This problem has been well studied in the *reinforcement learning* literature [5], [6]. Although most of the literature studies stochastic problems with finite state and action spaces [1], [7], we will focus on deterministic optimal control problems where state and action spaces are continuous; see *e.g.* [8], [9].

Problem description

We consider a deterministic discrete-time system with dynamics

$$x_{t+1} = f(x_t, u_t),$$

where $x_t \in \mathbb{X}$ and $u_t \in \mathbb{U}$ are the state and control input (action) at time $t \in \mathbb{N}_0$, and $f : \mathbb{X} \times \mathbb{U} \mapsto \mathbb{X}$ is the dynamics function. We assume that $\mathbb{X}$ and $\mathbb{U}$ are continuous spaces. The control input u_t is chosen from a set $\mathbb{U}(x_t) \subseteq \mathbb{U}$. The cost of operating the system at time t is denoted by $l(x_t, u_t)$, where $l : \mathbb{X} \times \mathbb{U} \mapsto \mathbb{R}_+ \cup \{+\infty\}$ is the (nonnegative) *stage cost* function. We assume that for each $x_t \in \mathbb{X}$ there exists a $u_t \in \mathbb{U}(x_t)$ such that $l(x_t, u_t) < +\infty$.

This research was supported by the European Research Council under the ERC Advanced Grant agreement no. 787845 (OCAL).

The authors are with the Automatic Control Laboratory, ETH Zurich, Switzerland. {gbanjac,lygeros}@control.ee.ethz.ch

A decision rule for selecting the control input at a specific time t is referred to as a *policy*. We are interested in stationary state-dependent policies of the form $\mu : \mathbb{X} \mapsto \mathbb{U}$. The value of policy μ is given by the function $V_\mu : \mathbb{X} \mapsto \mathbb{R}_+$ defined as

$$V_\mu(x) \coloneqq \sum_{t=0}^{\infty} \gamma^t l(x_t, \mu(x_t)), \tag{1}$$

where $x_0 = x$, $x_{t+1} = f(x_t, \mu(x_t))$ for $t \in \mathbb{N}_0$, and $\gamma \in (0, 1)$ is a *discount factor*. The optimal value function is defined as

$$V^\star(x) \coloneqq \inf_{\mu \in \Pi} V_\mu(x), \tag{2}$$

where Π is the set of all stationary state-dependent policies. A policy $\mu^\star \in \Pi$ is said to be optimal if

$$V_{\mu^\star}(x) = \inf_{\mu \in \Pi} V_\mu(x) = V^\star(x), \quad \forall x \in \mathbb{X}.$$

Our goal here is to learn an optimal policy of a system whose dynamics function f and stage cost function l are unknown, but we can interact with the system by measuring its state x_t, applying control input u_t, and observing the resulting state $x_{t+1} = f(x_t, u_t)$ and cost $l_t = l(x_t, u_t)$.

Related work

A traditional approach in designing a controller for an unknown system consists of an offline procedure called *system identification* in which a model of the system is estimated from data, and then a controller is designed for the estimated model [10]. A similar procedure is used in [11] where the authors estimate both the model and uncertainty, and then design a robust controller that takes this uncertainty into account. The authors in [12] use system identification in each iteration of an online learning procedure. As an alternative, the authors in [13], [14] use a model-free framework to learn the optimal Q-function directly from data by employing the PI algorithm.

In this paper we adopt the model-free approach based on learning the optimal Q-function from which we can extract the optimal policy. The optimal value and Q-functions can be characterized as solutions to infinite dimensional linear programs (LPs). This characterization dates back to 1960's [15] and is used in [16]–[19] in the context of ADP where infinite-dimensional LPs are approximated by finite dimensional programs. We combine the PI algorithm with the linear programming approach to learn a sequence of Q-functions. The proposed scheme is computationally tractable, ensures a monotonic decrease of a certain optimality metric, and allows for inclusion of exploration strategies.

Notation

Let $\mathbb{N}$ denote the set of positive integers, $\mathbb{N}_0 \coloneqq \mathbb{N} \cup \{0\}$ the set of nonnegative integers, $\mathbb{R}$ the set of real numbers, and $\mathbb{R}_+$ the set of nonnegative real numbers. For some $N \in \mathbb{N}$, we define $\mathbb{N}_{[1,N]} \coloneqq \{1, \dots, N\}$. We denote by $\mathcal{F}(\mathbb{X} \times \mathbb{U})$ the vector space of real-valued measurable functions on $\mathbb{X} \times \mathbb{U}$ with a finite w-weighted norm, where w is a certain function that depends on the stage cost; see [2, Def. 6.3.2] for details. Similarly, we denote by $\mathcal{M}_+(\mathbb{X} \times \mathbb{U})$ the space of finite nonnegative measures on $\mathbb{X} \times \mathbb{U}$ with a finite w-weighted total variation. For $f \in \mathcal{F}(\mathbb{X} \times \mathbb{U})$ and $g \in \mathcal{F}(\mathbb{X} \times \mathbb{U})$, $f \leq g$ denotes pointwise inequality, *i.e.* $f(x,u) \leq g(x,u)$ for all $(x,u) \in \mathbb{X} \times \mathbb{U}$; a similar notation is used for pointwise equality.

II. Dynamic Programming Background

It is well known that a solution to the optimal control problem (2) can be characterized using DP [20]. In this section we introduce some definitions and results from the theory of DP that we require to analyze algorithms for computing an optimal policy; we refer the reader to [3], [4] for a comprehensive review.

For any stationary state-dependent policy μ, V_μ satisfies the following equation [3, Prop. 4.1.2]

$$V_\mu(x) = l(x, \mu(x)) + \gamma V_\mu\left(f(x, \mu(x))\right), \quad \forall x \in \mathbb{X}. \tag{3}$$

The optimal value function $V^\star$ is the unique solution to *Bellman's equation* given by [3, Prop. 4.1.1]

$$V^\star(x) = \inf_{u \in \mathbb{U}} \left\{ l(x,u) + \gamma V^\star\left(f(x,u)\right) \right\}, \quad \forall x \in \mathbb{X}.$$

Once $V^\star$ is known, an optimal policy can be evaluated as

$$\mu^\star(x) \in \operatorname*{argmin}_{u \in \mathbb{U}} \left\{ l(x,u) + \gamma V^\star\left(f(x,u)\right) \right\}.$$

Note, however, that evaluating an optimal policy requires not only availability of the optimal value function $V^\star$, but also the dynamics function f and stage cost function l. If f or l are unknown, then knowing $V^\star$ is not sufficient for evaluating an optimal policy. To overcome this issue, we will use the optimal Q-function to compute an optimal policy without knowing f and l explicitly.

A. Q-function

The *Q-function* associated to policy μ is defined as [21]

$$Q_\mu(x,u) \coloneqq l(x,u) + \gamma V_\mu(f(x,u)), \tag{4}$$

and can be interpreted as the cost of applying control input $u \in \mathbb{U}$ at state $x \in \mathbb{X}$, and following policy μ afterwards. Observe from (3) and (4) that V_μ can be expressed in terms of Q_μ as

$$V_\mu(x) = Q_\mu(x, \mu(x)),$$

which combined with (4) gives the following equation that holds for all $(x,u) \in \mathbb{X} \times \mathbb{U}$:

$$Q_\mu(x,u) = l(x,u) + \gamma Q_\mu\left(f(x,u), \mu(f(x,u))\right).$$

The optimal Q-function is given by

$$Q^\star(x,u) \coloneqq Q_{\mu^\star}(x,u) = l(x,u) + \gamma V^\star(f(x,u)). \tag{5}$$

Similarly as for V_μ, we can express $V^\star$ in terms of $Q^\star$, *i.e.*

$$V^\star(x) \coloneqq \inf_{u \in \mathbb{U}} Q^\star(x,u). \tag{6}$$

Observe from (5) that we can evaluate an optimal policy via

$$\mu^\star(x) \in \operatorname*{argmin}_{u \in \mathbb{U}} Q^\star(x,u), \tag{7}$$

which does not directly involve dynamics and stage cost functions. The characterization above means that the problem of finding an optimal policy boils down to learning the optimal Q-function.

Combining (5) and (6), we can express Bellman's equation in terms of the optimal Q-function, *i.e.* for all $(x,u) \in \mathbb{X} \times \mathbb{U}$ we have

$$Q^\star(x,u) = l(x,u) + \gamma \inf_{u' \in \mathbb{U}} Q^\star\big(f(x,u), u'\big). \tag{8}$$

We next show that a solution to the equation above can be characterized as a solution to an LP.

B. The linear programming formulation

Equation (8) characterizes the optimal Q-function as the fixed-point of *Bellman's operator*, which is defined for all $q \in \mathcal{F}(\mathbb{X} \times \mathbb{U})$ as

$$(\mathcal{T}q)(x,u) \coloneqq \left\{ l(x,u) + \gamma \inf_{u' \in \mathbb{U}} q\left(f(x,u), u'\right) \right\}.$$

The operator $\mathcal{T}$ satisfies *monotonicity* and *value iteration convergence* [18], [19], [22], which means that for any functions $q_1, q_2 \in \mathcal{F}(\mathbb{X} \times \mathbb{U})$,

$$q_1 \leq q_2 \quad \Longrightarrow \quad \mathcal{T}q_1 \leq \mathcal{T}q_2$$

and

$$Q^\star = \lim_{k \to \infty} \mathcal{T}^k q_1.$$

If $\hat{Q} \in \mathcal{F}(\mathbb{X} \times \mathbb{U})$ satisfies *Bellman's inequality*

$$\hat{Q} \leq \mathcal{T}\hat{Q}, \tag{9}$$

then the aforementioned properties of $\mathcal{T}$ imply

$$\hat{Q} \leq \mathcal{T}\hat{Q} \leq \ldots \leq \lim_{k \to \infty} \mathcal{T}^k \hat{Q} = Q^\star.$$

In other words, inequality (9) implies that $\hat{Q}$ is a pointwise lower bound to $Q^\star$. It is then natural to look for the greatest lower bound in the space $\mathcal{F}(\mathbb{X} \times \mathbb{U})$. Due to the infimum in the definition of $\mathcal{T}$, (9) is in general nonlinear in $\hat{Q}$, but can be represented equivalently as the following set of linear constraints

$$l(x,u) \geq (\mathcal{D}\hat{Q})(x,u,u'), \quad \forall (x,u,u') \in \mathbb{X} \times \mathbb{U}^2,$$

where $(\mathcal{D}q) \in \mathcal{F}(\mathbb{X} \times \mathbb{U}^2)$ is a function defined as

$$(\mathcal{D}q)(x,u,u') \coloneqq q(x,u) - \gamma q(f(x,u), u').$$

This leads to the following infinite dimensional LP:

$$\begin{aligned} \max_{q} \quad & \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x,\mathrm{d}u) \\ \text{s.t.} \quad & l(x,u) \geq (\mathcal{D}q)(x,u,u'), \quad \forall (x,u,u') \in \mathbb{X}\times\mathbb{U}^2 \\ & q \in \mathcal{F}(\mathbb{X}\times\mathbb{U}), \end{aligned} \tag{10}$$

where $c \in \mathcal{M}_+(\mathbb{X}\times\mathbb{U})$ is a finite measure that assigns positive mass to all open subsets of $\mathbb{X}\times\mathbb{U}$. Note that, due to the nonnegativity of l, the problem above is feasible since $\hat{Q} = 0$ satisfies the inequality in (10) for all $(x,u,u') \in \mathbb{X}\times\mathbb{U}^2$.

Proposition 1. *The solution to equation* (8) *coincides with a solution to LP* (10) *for c-almost all* $(x,u) \in \mathbb{X}\times\mathbb{U}$.

Proof. Due to (8), we have

$$\begin{aligned} Q^\star(x,u) &= l(x,u) + \gamma \inf_{u'\in\mathbb{U}} Q^\star\big(f(x,u),u'\big) \\ &\leq l(x,u) + \gamma Q^\star\big(f(x,u),u'\big) \end{aligned}$$

for all $(x,u,u') \in \mathbb{X}\times\mathbb{U}^2$, which means that $Q^\star$ is feasible for LP (10). Also, for any $q \in \mathcal{F}(\mathbb{X}\times\mathbb{U})$ feasible for (10), the following holds for all $(x,u,u') \in \mathbb{X}\times\mathbb{U}^2$

$$q(x,u) \leq l(x,u) + \gamma q(f(x,u),u'),$$

and thus the inequality holds for a specific $u' \in \mathbb{U}$ that minimizes $q(f(x,u),\cdot)$, *i.e.*

$$\begin{aligned} q(x,u) &\leq l(x,u) + \gamma \inf_{u'\in\mathbb{U}} q(f(x,u),u') \\ &= (\mathcal{T}q)(x,u). \end{aligned}$$

The inequality above implies that q is a lower bound to $Q^\star$ (see Section II-B). Since we are maximizing the objective function in (10), this means that $Q^\star$ is a maximizer of the LP, as well as all bounded functions that differ from $Q^\star$ on a subset of $\mathbb{X}\times\mathbb{U}$ with measure zero. □

The equivalence between (8) and (10) requires that the function space over which the decision variable is optimized contains an element $\hat{q} \in \mathcal{F}(\mathbb{X}\times\mathbb{U})$ for which $\hat{q}(x,u) \leq (\mathcal{T}\hat{q})(x,u)$ is satisfied with equality for all $(x,u) \in \mathbb{X}\times\mathbb{U}$ [19]. Note that the LP in (10) does not involve an additional optimization variable $v \in \mathcal{F}(\mathbb{X})$ as in [18], [19]. The elimination of v is possible in our case since we work in a deterministic setting.

C. Policy iteration

The PI algorithm aims to compute an optimal stationary policy $\mu^\star$ for problem (2) via an iterative procedure [4, §2.3]. It starts from an initial policy μ^1 and performs at each iteration $k \in \mathbb{N}$ the *policy evaluation* step, *i.e.* computes Q_{μ^k} satisfying

$$Q_{\mu^k}(x,u) = l(x,u) + \gamma Q_{\mu^k}\left(f(x,u),\mu^k(f(x,u))\right) \tag{11}$$

for all $(x,u) \in \mathbb{X}\times\mathbb{U}$, followed by the *policy improvement* step to obtain μ^{k+1} according to

$$\mu^{k+1}(x) \in \operatorname*{argmin}_{u\in\mathbb{U}} Q_{\mu^k}(x,u), \quad \forall x \in \mathbb{X}.$$

Under certain assumptions, it can be shown that the sequence $\{Q_{\mu^k}\}_{k\in\mathbb{N}}$ generated by the PI algorithm is nonincreasing and converges to $Q^\star$ [2, Thm. 4.4.1]. These properties have made the PI algorithm an attractive candidate for improving performance of a controlled dynamical system [9].

Let $\mathcal{T}_\mu$ be the operator defined for any stationary policy μ and for all $q \in \mathcal{F}(\mathbb{X}\times\mathbb{U})$ as

$$(\mathcal{T}_\mu q)(x,u) \coloneqq l(x,u) + \gamma q\left(f(x,u),\mu(f(x,u))\right).$$

Then Q_{μ^k} satisfying equation (11) can be seen as the fixed-point of $\mathcal{T}_{\mu^k}$, which satisfies both monotonicity and value iteration convergence [18]. Following similar arguments as in Section II-B, it is easy to show that Q_{μ^k} coincides with a solution to the following LP:

$$\begin{aligned} \max_{q} \quad & \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x,\mathrm{d}u) \\ \text{s.t.} \quad & l(x,u) \geq (\mathcal{D}q)\left(x,u,\mu^k(f(x,u))\right), \ \forall (x,u) \in \mathbb{X}\times\mathbb{U} \\ & q \in \mathcal{F}(\mathbb{X}\times\mathbb{U}), \end{aligned} \tag{12}$$

for c-almost all $(x,u) \in \mathbb{X}\times\mathbb{U}$.

We show in the sequel how infinite dimensional LPs introduced in this section can be approximated by finite dimensional programs.

D. Finite approximation

As already mentioned, evaluating an optimal policy can be done by computing the optimal Q-function via (10), and then implementing (7). However, this procedure is computationally intractable in general for the following reasons [19], [23]:

(i) q is an optimization variable in the infinite dimensional space $\mathcal{F}(\mathbb{X}\times\mathbb{U})$.
(ii) The number of constraints is infinite since the inequalities need to be satisfied for all $(x,u,u') \in \mathbb{X}\times\mathbb{U}^2$.
(iii) Evaluating the objective function in (10) involves computing a multidimensional integral.
(iv) Since $Q^\star$ can be any element of $\mathcal{F}(\mathbb{X}\times\mathbb{U})$, the optimization problem in (7) may be intractable.

To overcome these sources of intractability, the original optimization problem is approximated by a tractable one. This approach is referred to as ADP, and is often used to compute approximate value and Q-functions [17]–[19], [22]. As suggested in [16], [19], we restrict the Q-function in the span of a finite family of basis functions $\hat{q}_i \in \mathcal{F}(\mathbb{X}\times\mathbb{U})$ for $i \in \mathbb{N}_{[1,N]}$, and parameterize the restricted function space as

$$\hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U}) \coloneqq \left\{ \textstyle\sum_{i=1}^N \alpha_i \hat{q}_i \mid \alpha \in \mathbb{R}^N \right\}.$$

It is often the case that the basis functions are chosen so that evaluation of the objective function in the LP and implementing (7) is relatively easy [19], [24].

Problem (10) can thus be approximated by the following

semi-infinite LP:

$$\begin{aligned} \max_{q} \quad & \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x,\mathrm{d}u) \\ \text{s.t.} \quad & l(x,u) \geq (\mathcal{D}q)(x,u,u'),\ \forall (x,u,u') \in \mathbb{X}\times\mathbb{U}^2 \\ & q \in \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U}), \end{aligned} \tag{13}$$

and we denote its solution by $\hat{Q}^\star$. The authors in [17], [19] show that the quality of a policy obtained from $\hat{Q}^\star$ depends on the distance between $Q^\star$ and the space $\hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U})$. Also, while the specific choice of $c \in \mathcal{M}_+(\mathbb{X}\times\mathbb{U})$ in (10) does not affect the optimal solution $Q^\star$ when optimizing over the entire function space $\mathcal{F}(\mathbb{X}\times\mathbb{U})$, it plays an important role in determining the quality of $\hat{Q}^\star$ in the approximated problem.

When f and l are known and have a certain structure, and the basis functions are selected appropriately, the constraint set in (13) can sometimes be represented exactly or relaxed via the $\mathcal{S}$-procedure or sum-of-squares [22]–[24]. Since in our setup f and l are unknown, we cannot approximate the constraint set using these tools. However, it is possible to relax the constraints via sampling [25]–[28]. In particular, since we can measure state $x_i \in \mathbb{X}$, apply input $u_i \in \mathbb{U}$, observe the new state $x_i^+ = f(x_i,u_i)$ and the incurred cost $l_i = l(x_i,u_i)$, we can obtain a sequence of tuples $\left\{(x_i,u_i,x_i^+,l_i)\right\}_{i=1}^M$ through an experiment and replace the inequality constraint set in (13) by its sampled version

$$l_i \geq q(x_i,u_i) - \gamma q\left(x_i^+,u_i'\right), \quad i \in \mathbb{N}_{[1,M]},$$

for some $u_i' \in \mathbb{U}$, $i \in \mathbb{N}_{[1,M]}$. Note that we use index i instead of t to emphasize that the data tuples do not have to be consecutive in time.

The authors in [28], [29] use a similar constraint sampling approach and provide probabilistic bounds on the number of samples M needed to attain a certain level of approximation. A drawback of this approach is that the bounds on M are usually too conservative, which means that we need to sample a large number of constraints and solve a large-scale LP. Also, it is not clear how to select $\{u_i'\}_{i=1}^M$ to improve the performance of the method.

Instead of computing $\hat{Q}^\star$ directly from (13), we present in the next section a sampled version of the PI algorithm, where the policy evaluation step is computed by sampling constraints in (12).

III. Data-Driven Policy Iteration

In this section we explore how to use data obtained from interacting with a dynamical system to learn approximately a policy that minimizes (1). We consider a data-driven version of the PI algorithm in which we evaluate a policy by solving the following sampled version of (12) with N variables and M constraints:

$$\begin{aligned} \max_{q} \quad & \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x,\mathrm{d}u) \\ \text{s.t.} \quad & l_i \geq q(x_i,u_i) - \gamma q\left(x_i^+,\mu^k(x_i^+)\right), \quad i \in \mathbb{N}_{[1,M]} \\ & q \in \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U}). \end{aligned} \tag{14}$$

The authors in [13], [14] consider learning an optimal controller for an unknown deterministic system with linear dynamics and quadratic stage cost, known as the linear quadratic regulator (LQR). They use a sampled version of the PI algorithm in which Q_{μ^k} is restricted to the function space containing all quadratic functions on $\mathbb{X}\times\mathbb{U}$, and estimate it from equation (11) via recursive least-squares. Also, they rely on the fact that $Q_{\mu^k} \in \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U})$, which implies that it can be recovered exactly from $M \geq N$ data points as long as a certain persistent excitation condition is satisfied [14]. In this case one can use convergence analysis for the exact PI algorithm and prove that the sequence $\{Q_{\mu^k}\}_{k\in\mathbb{N}}$ is (pointwise) nonincreasing and convergent.

However, if $Q_{\mu^k} \notin \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U})$, then we cannot use the convergence results of the exact PI algorithm any more. In this case there will be no function q that satisfies all the constraints in (14) with equality, and a solution to the LP will depend on the choice of measure c.

Our goal is to design an algorithmic scheme based on data-driven PI that improves an estimate of $\hat{Q}^\star$ in each iteration. Also, we would like to allow inclusion of expert knowledge on the dynamical system in the learning process. Observe that in LP (13) we can select any $u' \in \mathbb{U}$, which gives us a certain freedom in exploring different control strategies, while in LP (14) u_i' is fixed to $\mu^k(x_i^+)$. We can combine these two approaches and solve instead the following LP:

$$\begin{aligned} \max_{q} \quad & \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x,\mathrm{d}u) \\ \text{s.t.} \quad & l_i \geq q(x_i,u_i) - \gamma q\left(x_i^+,u_i'\right), \quad i \in \mathbb{N}_{[1,M]} \\ & q \in \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U}), \end{aligned} \tag{15}$$

for some $u_i' \in \mathbb{U}$, $i \in \mathbb{N}_{[1,M]}$. Note that (14) can be seen as a special case of (15) where $u_i' = \mu^k(x_i^+)$ for all $i \in \mathbb{N}_{[1,M]}$. Moreover, we can reuse a data tuple (x_i,u_i,x_i^+,l_i) and add another constraint with different u_i'.

One way of ensuring the nonincreasing objective function when solving a sequence of LPs (15) is to keep all the constraints from previous iterations. However, this strategy would imply that the number of constraints in the LPs grows in each iteration, which would pose computational difficulties when the iteration counter k becomes large. Alternatively, we can keep only the *binding constraints* from iteration k and include them as additional constraints in the LP solved in iteration $k+1$; a similar strategy is used in [30] for solving semi-infinite convex programs. We define binding constraints as those with nonzero optimal Lagrange multiplier. An important property of such constraints is that removing all the other constraints does not change the optimal value of the problem. Moreover, for non-degenerate LPs the number of such constraints is not greater than N. Although degenerate LPs can have more than N binding constraints, it is always possible to select a subset of linearly independent constraints so that removing all the other constraints does not change the optimal objective value. Therefore, without loss of generality, we assume that the number of binding constraints is bounded above by N.

Alg. 1 Memory-efficient data-driven PI algorithm.

1: **given** parameters $M, N, P \in \mathbb{N}$, set of basis functions $\{\hat{q}_i\}_{i=1}^{N}$, and initial policy μ^1
2: Set $k = 1$ and $S^0 = \emptyset$
3: **repeat**
4: **for** $p = 1, \ldots, P$ **do**
5: Measure $x_p \in \mathbb{X}$
6: Apply $u_p \in \mathbb{U}$
7: Observe $x_p^+ \in \mathbb{X}$ and $l_p \in \mathbb{R}_+$
8: **for** $i = 1, \ldots, M$ **do**
9: Select $(x_i, u_i, x_i^+, l_i) \in \left\{(x_p, u_p, x_p^+, l_p)\right\}_{p=1}^{P}$
10: Select $u_i' \in \mathbb{U}$
11: $\hat{Q}_{\mu^k} \leftarrow \max_{q} \int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x, \mathrm{d}u)$
$\text{s.t. } l_i \geq q(x_i, u_i) - \gamma q\left(x_i^+, u_i'\right),\ \forall i \in \mathbb{N}_{[1,M]}$
$l_j \geq q(x_j, u_j) - \gamma q\left(x_j^+, u_j'\right),\ \forall j \in S^{k-1}$
$q \in \hat{\mathcal{F}}(\mathbb{X} \times \mathbb{U})$
12: $S^k \leftarrow$ binding constraints of the LP above
13: $\mu^{k+1}(x) \leftarrow \operatorname{argmin}_{u\in\mathbb{U}} \hat{Q}_{\mu^k}(x,u), \quad \forall x \in \mathbb{X}$
14: $k \leftarrow k + 1$
15: **until** convergence

The proposed algorithm is summarized in Alg. 1. It can be seen as an *actor-critic algorithm*, where the critic computes an estimate of Q_{μ^k}, which is then used by the actor to update the policy [31]. We assume that the restricted function space $\hat{\mathcal{F}}(\mathbb{X} \times \mathbb{U})$ is selected so that step 13 of Alg. 1 can be evaluated efficiently. Note that the number of constraints in the LP solved in each iteration is bounded above by $M + N$. We make a distinction between the number of data tuples P obtained through an experiment, and the number of derived constraints M, since we can use a single tuple (x_i, u_i, x_i^+, l_i) to generate multiple constraints by considering different values of $u_i' \in \mathbb{U}$, which usually depend on μ^k. Regardless of the choice of u_i', we have the following lemma:

Lemma 1. *Suppose that for $k = 1$ the optimal objective value of the LP in Alg. 1 is finite. Then the optimal objective values of LPs solved in Alg. 1 are monotonically nonincreasing.*

Proof. The optimal objective value of the LP solved in iteration k is the same as the optimal objective value of the LP in which we keep only binding constraints S^k. Since the constraint set of the LP solved in iteration $k+1$ includes the binding constraints S^k, along with additional constraints, the optimal objective value cannot increase. □

IV. Numerical Example

We illustrate performance of the proposed algorithm on an input-constrained LQR example from [22] with one state and one input. In particular, we consider a system with linear dynamics

$$x_{t+1} = x_t - 0.5u_t,$$

where $\mathbb{X} = \mathbb{R}$, $\mathbb{U} = [-1, 1]$, and stage cost $l(x,u) = x^2 + 0.1u^2$. We restrict the function space to the subspace of quadratic functions that are strongly convex in u, *i.e.*

$$\hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U}) = \{S_{11}x^2 + 2S_{12}xu + S_{22}u^2 + d_1x + d_2u + e \mid (S_{11}, S_{12}, d_1, d_2, e) \in \mathbb{R},\ S_{22} \geq 10^{-3}\},$$

and define

$$S \coloneqq \begin{bmatrix} S_{11} & S_{12} \\ S_{12} & S_{22} \end{bmatrix} \quad \text{and} \quad d \coloneqq \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}.$$

Due to the input constraints, the optimal Q-function is not quadratic and thus $Q^\star \notin \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U})$. Note that minimizing a function $q \in \hat{\mathcal{F}}(\mathbb{X}\times\mathbb{U})$ over $u \in \mathbb{U}$ results in a policy of the form

$$\mu(x) = \Pi_{[-1,1]}(Kx + k),$$

where $\Pi_{[-1,1]}$ denotes the projection onto the interval $[-1, 1]$. We set the initial policy to $\mu^1(x) = \Pi_{[-1,1]}(0.1x)$. The measure c is chosen as a probability measure with zero mean and covariance matrix $\Sigma = \mathrm{diag}(1, 0.1)$. Note that the objective function in LP (15) then takes the form [19]

$$\int_{\mathbb{X}\times\mathbb{U}} q(x,u)\, c(\mathrm{d}x, \mathrm{d}u) = \mathbb{E}_c\, q = \mathrm{Tr}(S\Sigma) + e.$$

In each iteration of Alg. 1 we generate 5 rollouts of length 7. For each rollout we set an initial state at $x_i \sim \mathcal{U}[-5, 5]$ and apply an action according to $u_i = \Pi_{[-1,1]}\left(\mu^k(x_i) + 0.25\varepsilon_i\right)$ where $\varepsilon_i \sim \mathcal{N}(0, 1)$. For each data tuple (x_p, u_p, x_p^+, l_p) we generate $u_p' = \Pi_{[-1,1]}\left(\mu^k(x_p^+)\right)$ and $u_p'' = \Pi_{[-1,1]}\left(\mu^k(x_p^+) + 0.1\delta_p\right)$ where $\delta_p \sim \mathcal{U}[-1, 1]$.

We consider two variants of Alg. 1. In Method 1 we construct constraints in the LPs using $\{(x_p, u_p, x_p^+, l_p, u_p')\}_{p=1}^{35}$, while in Method 2 we use both $\{(x_p, u_p, x_p^+, l_p, u_p')\}_{p=1}^{35}$ and $\{(x_p, u_p, x_p^+, l_p, u_p'')\}_{p=1}^{35}$. We observe that in both variants the sequence $\{Q_{\mu^k}\}_{k\in\mathbb{N}}$ converges to the solution of the unconstrained discounted-cost LQR

$$\hat{Q}^\star(x,u) = \begin{bmatrix} x & u \end{bmatrix} \hat{S}^\star \begin{bmatrix} x \\ u \end{bmatrix},$$

where $\hat{S}^\star$ has a closed-form expression [13].

Fig. 1 illustrates the case in which Method 2 benefits from including additional constraints in the LPs, and admits a faster convergence rate than Method 1. However, other numerical examples show that this is not always the case. A further investigation is needed to understand how to construct additional constraints so that the convergence rate of the method improves.

V. Conclusion and Discussion

We propose a method for learning discounted-cost optimal control policies for unknown deterministic systems with continuous state and action spaces. We combine ideas from the PI algorithm and the linear programming approach to learn the optimal Q-function via an iterative procedure. The proposed method is memory-efficient, ensures a certain monotonic behavior, and allows for incorporation of expert knowledge and exploration strategies.

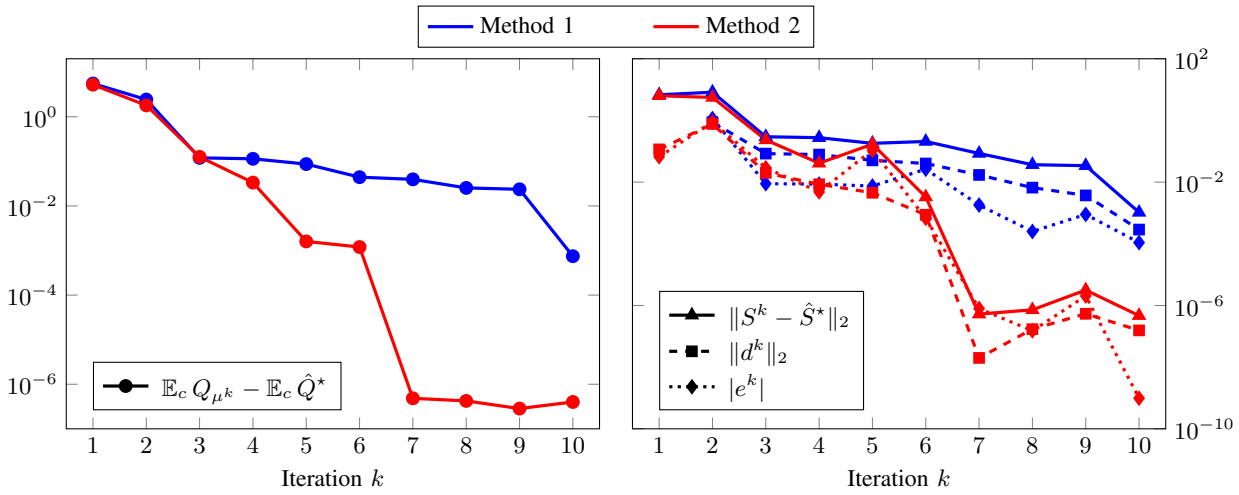

Fig. 1. Numerical performance of two variants of Alg. 1 for solving a simple input-constrained LQR example.

For future work, there are several interesting directions. Although we assume that the optimal objective values of LPs solved in Alg. 1 are finite, we do not provide exact conditions on how to select state-action pairs (x_i, u_i) to ensure this. Second, although the objective value of the LP solved in the first iteration of Alg. 1 does not increase by including additional constraints, the overall performance of the algorithm does not necessarily improve. Another open question is how to extend the proposed algorithm to learn optimal policies of stochastic systems.

REFERENCES

[1] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[2] O. Hernández-Lerma and J. B. Lasserre, *Discrete-Time Markov Control Processes: Basic Optimality Criteria*. Springer-Verlag, 1996.

[3] D. P. Bertsekas, *Dynamic Programming and Optimal Control: Approximate Dynamic Programming, Vol. II*, 4th ed. Athena Scientific, 2012.

[4] ——, *Dynamic Programming and Optimal Control, Vol. I*, 4th ed. Athena Scientific, 2017.

[5] S. Sutton, R and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.

[6] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: performance, stability, and deep approximators," *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[7] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. The MIT Press, 2012.

[8] D. P. Bertsekas, "Value and policy iterations in optimal control and adaptive dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 500–509, 2017.

[9] W. Guo, J. Si, F. Liu, and S. Mei, "Policy approximation in policy iteration approximate dynamic programming for discrete-time nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 2794–2807, 2018.

[10] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Prentice Hall, 1999.

[11] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu, "On the sample complexity of the linear quadratic regulator," *Foundations of Computational Mathematics (to appear)*, 2019.

[12] Y. Ouyang, M. Gagrani, and R. Jain, "Learning-based control of unknown linear systems with Thompson sampling," *arXiv:1709.04047*, 2017.

[13] S. Bradtke, "Reinforcement learning applied to linear quadratic regulation," in *Advances in Neural Information Processing Systems (NIPS)*, 1992.

[14] S. Bradtke, B. Ydstie, and A. Barto, "Adaptive linear quadratic control using policy iteration," in *American Control Conference (ACC)*, 1994.

[15] A. Manne, "Linear programming and sequential decisions," *Management Science*, vol. 6, no. 3, pp. 259–267, 1960.

[16] P. Schweitzer and A. Seidmann, "Generalized polynomial approximations in Markovian decision processes," *Journal of Mathematical Analysis and Applications*, vol. 110, no. 2, pp. 568–582, 1985.

[17] D. P. de Farias and B. Van Roy, "The linear programming approach to approximate dynamic programming," *Operations Research*, vol. 51, no. 6, pp. 850–865, 2003.

[18] R. Cogill, M. Rotkowitz, B. Van Roy, and S. Lall, "An approximate dynamic programming approach to decentralized control of stochastic systems," in *Control of Uncertain Systems: Modelling, Approximation, and Design*, 2006, pp. 243–256.

[19] P. Beuchat, A. Georghiou, and J. Lygeros, "Performance guarantees for model-based approximate dynamic programming in continuous spaces," *IEEE Transactions on Automatic Control (to appear)*, 2019.

[20] R. Bellman, "On the theory of dynamic programming," *Proceedings of the National Academy of Sciences*, vol. 38, no. 8, pp. 716–719, 1952.

[21] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[22] Y. Wang, B. O'Donoghue, and S. Boyd, "Approximate dynamic programming via iterated Bellman inequalities," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 10, pp. 1472–1496, 2015.

[23] T. Summers, K. Kunz, N. Kariotoglou, M. Kamgarpour, S. Summers, and J. Lygeros, "Approximate dynamic programming via sum of squares programming," in *European Control Conference (ECC)*, 2013.

[24] C. Savorgnan, J. B. Lasserre, and M. Diehl, "Discrete-time stochastic optimal control via occupation measures and moment relaxations," in *IEEE Conference on Decision and Control (CDC)*, 2009.

[25] D. P. de Farias and B. Van Roy, "On constraint sampling in the linear programming approach to approximate dynamic programming," *Mathematics of Operations Research*, vol. 29, no. 3, pp. 462–478, 2004.

[26] A. Petretti and M. Prandini, "An approximate linear programming solution to the probabilistic invariance problem for stochastic hybrid systems," in *IEEE Conference on Decision and Control (CDC)*, 2014.

[27] A. Falsone and M. Prandini, "An iterative scheme for the approximate linear programming solution to the optimal control of a Markov decision process," in *European Control Conference (ECC)*, 2015.

[28] P. Mohajerin Esfahani, T. Sutter, D. Kuhn, and J. Lygeros, "From infinite to finite programs: explicit error bounds with applications to approximate dynamic programming," *SIAM Journal on Optimization*, vol. 28, no. 3, pp. 1968–1998, 2018.

[29] T. Sutter, A. Kamoutsi, P. Mohajerin Esfahani, and J. Lygeros, "Data-driven approximate dynamic programming: a linear programming approach," in *IEEE Conference on Decision and Control (CDC)*, 2017.

[30] L. Zhang, S. Wu, and M. López, "A new exchange method for convex semi-infinite programming," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 2959–2977, 2010.

[31] V. Konda and J. Tsitsiklis, "On actor-critic algorithms," *SIAM Journal on Control and Optimization*, vol. 42, no. 4, pp. 1143–1166, 2003.