Safe Approximate Dynamic Programming via Kernelized Lipschitz Estimation

Ankush Chakrabarty, *Senior Member, IEEE*, Devesh K. Jha, Gregery T. Buzzard, Yebin Wang, *Senior Member, IEEE*, and Kyriakos G. Vamvoudakis, *Senior Member, IEEE*

***Abstract*— We develop a method for obtaining safe initial policies for reinforcement learning via approximate dynamic programming (ADP) techniques for uncertain systems evolving with discrete-time dynamics. We employ the kernelized Lipschitz estimation to learn multiplier matrices that are used in semidefinite programming frameworks for computing admissible initial control policies with provably high probability. Such admissible controllers enable safe initialization and constraint enforcement while providing exponential stability of the equilibrium of the closed-loop system.**

***Index Terms*— Approximate dynamic programming (ADP), data-driven Lipschitz constant estimation, incremental quadratic constraints, kernel density estimation (KDE), semidefinite programming.**

I. Introduction

RECENT advances in the field of deep and machine learning have led to a renewed interest in using learning for control of physical systems [1]. Reinforcement learning (RL) is a learning framework that handles sequential decision-making problems, in which an "agent" or decision maker learns a policy to optimize a long-term reward by interacting with the (unknown) environment. At each step, an RL agent obtains evaluative feedback (called reward or cost) about the performance of its action, allowing it to improve the performance of subsequent actions [2], [3]. While RL has witnessed huge success in recent times [4]–[7], there are several unsolved challenges which restricts the use of these algorithms for industrial systems. In most practical applications, control policies must be designed to satisfy operational constraints. This leads to the challenge that one has to guarantee constraint satisfaction during learning and policy optimization. Therefore, initializing with an unverified control policy is not "safe" (in terms of stability or constraint handling). In other words, using online RL for expensive equipment or safety-critical applications necessitates that the initial policy used for obtaining data for subsequently improved policies must be at least stabilizing and, generally, constraint enforcing. The work presented in this article is motivated by this challenge. We present a framework for deriving initial control policies from historical data that can be verified to satisfy constraints and guarantee stability while learning the optimal control policy online, from operational data.

A successful RL method needs to balance a fundamental tradeoff between exploration and exploitation. One needs to gather data safely (exploration) in order to best extract information from these data for optimal decision-making (exploitation). One way to solve the exploration and exploitation dilemma is to use optimistic initialization [8]–[11], but this assumes that the optimal policy is available until the data are obtained that proves otherwise. Such approaches have been applied to robotics applications, where systems with discrete and continuous state-action spaces [12], [13]. A limitation of these methods is that before the optimal policy is learned, the agent is quite likely to explore actions that lead to violation of the task-specific constraints as it aims to optimize the cumulative reward for the task. This shortcoming significantly limits such methods to apply to industrial applications since this could lead to irreparable hardware damage or harm human operators due to unexpected dynamics. Consequently, safe learning focuses on learning while enforcing safety constraints. There are primarily two types of approaches to safe RL and approximate/adaptive dynamic programming (ADP). These include modification of the optimization criterion with a safety component such as barrier functions by transforming the operational constraints into soft constraints [14], [15] and modifying the exploration process through the incorporation of external system knowledge or historical data [16]. Our method is among the latter class of methods because our operational constraints are hard constraints and softening them could lead to intermittent failure modes.

High-performance model-based control requires precise model knowledge for controller design. However, it is well known that for most applications, accurate model knowledge is practically elusive due to the presence of unmodeled dynamical interactions (e.g., friction and contacts). Recent efforts tackle this issue by learning control policies from operational (online) or archival data (off-line). Since the exact structure of the nonlinearity may be unknown or not amenable for analysis, researchers have proposed "indirect" data-driven controllers

Manuscript received June 26, 2019; revised October 30, 2019 and January 15, 2020; accepted March 2, 2020. The work of Gregery T. Buzzard was supported in part by the NSF under Grant CCF-1763896. The work of Kyriakos G. Vamvoudakis was supported in part by the NSF under Grant CPS-1851588 and Grant S&AS-1849198. *(Corresponding author: Ankush Chakrabarty.)*

Ankush Chakrabarty, Devesh K. Jha, and Yebin Wang are with Mitsubishi Electric Research Laboratories, Cambridge, MA 02139 USA (e-mail: achakrabarty@ieee.org; devesh.jha@merl.com; yebinwang@merl.com).

Gregery T. Buzzard is with the Department of Mathematics, Purdue University, West Lafayette, IN 47907 USA (e-mail: buzzard@purdue.edu).

Kyriakos G. Vamvoudakis is with the Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: kyriakos@gatech.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.2978805

that employ nonparametric learning methods, such as Gaussian processes to construct models from operational data [17], [18] to improve control policies online [19]. Conversely, "direct" methods, such as those proposed in [20]–[23], directly compute policies using a combination of archival/legacy and operational input–output data without constructing an intermediate model. For example, in [24], a human expert was introduced into the control loop to conduct initial experiments to ensure safety while generating archival data. A common assumption in many of these approaches is the availability of an initial control policy that is stabilizing and robust to uncertain dynamics. Designing such safe initial control policies in a computationally tractable manner remains an open challenge.

In this article, we present a formalism for synthesizing safe initial policies for uncertain nonlinear systems. We assume the presence of historical/archival/legacy system-related data, with which we estimate Lipschitz constants for the unmodeled system dynamics. The estimation of the Lipschitz constant is done via kernel density estimation (KDE). The estimated Lipschitz constant is used to design control policies using semidefinite programming methods that incorporate stability and constraint satisfaction while searching for policies. We show that the proposed approach can design feasible policies for different constrained tasks for several systems while respecting all active constraints.

Our key insight is that information regarding the structure of classes of unmodeled nonlinearities can be encapsulated using only a few parameters, without knowing the exact form of the nonlinearity. These classes include, for example, sector-bounded nonlinearities [25], Lipschitz and one-sided Lipschitz nonlinearities [26], and monotone nonlinearities [27], to name a few (see [25]–[28] and the references therein. It is well known that these nonlinearities can be represented by using multiplier matrices for linear matrix inequality (LMI)-based analysis and design [29]. Learning these multiplier matrices from data is an open problem, but if possible, would eliminate the need to model the unknown component perfectly in order to compute a safe control policy. For example, the class of Lipschitz nonlinearities (which constitute a large class of nonlinearities observed in applications) can be described using only a few parameters: the Lipschitz constants of the nonlinear components. This article is a first attempt at learning multiplier matrices for a class of Lipschitz nonlinearities in a data-driven manner.

Recent work has investigated the utility of Lipschitz properties in constructing controllers when an oracle is available [30], [31] or in designing models for prediction [32] with online data used for controller refinement [33]. In this article, we construct control policies that respect constraints and certify stability (with high probability) for applications where only off-line data are available and no oracle is present. We do so through the systematic use of multiplier matrices that enable the representation of nonlinear dynamics through quadratic constraints [29], [34] without requiring knowledge of the underlying nonlinearity. The control policies can then be obtained by solving semidefinite programs. However, the construction of multiplier matrices for Lipschitz systems requires knowledge of the Lipschitz constants, which are not always available and therefore must be estimated. We refer to the estimation of Lipschitz constants from data as Lipschitz learning. Historically, methods that estimate the Lipschitz constant [35]–[37] do not provide certificates on the quality of the estimate. Herein, we provide conditions that, if satisfied, enable us to estimate the Lipschitz constant of an unknown locally Lipschitz nonlinearity with high probability. To this end, we employ KDE, a nonparametric data-driven method that employs kernels to approximate smooth probability density functions to arbitrarily high accuracy. We refer to our proposed KDE-based Lipschitz constant estimation algorithm as kernelized Lipschitz learning.

A. Contributions

Compared with the existing literature on safe learning, the contributions of this article are threefold. First, we formulate an algorithm to construct stabilizing and constraint satisfying policies for nonlinear systems without knowing the exact form of the nonlinearity. Then, we leverage a kernelized Lipschitz learning mechanism to estimate Lipschitz constants of the uncertain dynamics with high probability, and finally, we use a multiplier-matrix-based controller design based on Lipschitz learning from legacy data that forces exponential stability on the closed-loop dynamics (with the same probability as the kernelized Lipschitz learner).

B. Structure

The rest of this article is structured as follows. We present the formal motivation of this article in Section II. Our kernelized Lipschitz learning algorithm is described in Section III, and benchmarking of the proposed learner on benchmark Lipschitz functions is performed. The utility of Lipschitz learning in policy design via multiplier matrices is elucidated in Section IV, and a numerical example demonstrating the potential of our overall formalism is provided in Section V. We provide the concluding remarks and discuss future directions in Section VI.

C. Notation

We denote $\mathbb{R}$ as the set of real numbers, $\mathbb{R}_+$ as the set of positive real, $\mathbb{N}$ as the set of natural numbers, and $\mathbb{N}_+ := \mathbb{N} \cup \{0\}$. The measure-based distance between two measurable subsets A and B of a metric space $\mathbb{R}^n$ equipped with the metric ρ_μ is given by $\rho_\mu(A, B) = \mu(A \triangle B)$, where μ is a measure on $\mathbb{R}^n$ and $A \triangle B$ is the symmetric difference $(A \backslash B) \cup (B \backslash A)$. We define a ball $\mathcal{B}_\epsilon(x) := \{y\colon \rho(x, y) \leq \epsilon\}$ and the sum $A \oplus \epsilon := \bigcup_{x \in A} \mathcal{B}_\epsilon(x)$. The complement of a set A is denoted by A^c. The indicator function of the set A is denoted by $\mathbf{1}_A$. A block diagonal matrix is denoted by blkdiag$(\cdot)$. For every $v \in \mathbb{R}^n$, we denote $\|v\| = (v^\top v)^{1/2}$, where $v^\top$ is the transpose of v. The sup-norm or ∞-norm is defined as $\|v\|_\infty \triangleq \sup_{t \in \mathbb{R}} \|v(t)\|$. We denote by $\lambda_{\min}(P)$ and $\lambda_{\max}(P)$ as the smallest and largest eigenvalues of a square, symmetric matrix P, respectively. The symbol $\succ$ ($\prec$) indicates positive (negative) definiteness and $A \succ B$ implies $A - B \succ 0$ for A and B of appropriate dimensions. Similarly, $\succeq$ ($\preceq$) implies positive

(negative) semidefiniteness. The operator norm is denoted $\|P\|$ and is defined as the maximum singular value of P. For a symmetric matrix, we use the $\star$ notation to imply symmetric terms, that is, $\left[\begin{smallmatrix} a & b \\ b^\top & c \end{smallmatrix}\right] \equiv \left[\begin{smallmatrix} a & b \\ \star & c \end{smallmatrix}\right]$. The symbol **Pr** denotes the probability measure.

II. Problem Formulation

Consider the following discrete-time nonlinear system:

$$x_{t+1} = F(x_t, u_t), \quad t \in \mathbb{N}$$
$$q_t = C_q x_t$$

where $x_t \in \mathbb{R}^{n_x}$ and $u = u_t \in \mathbb{R}^{n_u}$ denote the state and the control input of the system, respectively.

For simplicity of exposition, we consider input-affine systems of the form

$$x_{t+1} = Ax_t + Bu_t + G\phi(q_t), \quad t \in \mathbb{N} \tag{1a}$$
$$q_t = C_q x_t \tag{1b}$$

where the system matrices A, B, G, and C_q have appropriate dimensions. Denote by $\phi \in \mathbb{R}^{n_\phi}$ the system's uncertainty or unmodeled nonlinearity, whose argument $q = q_t \in \mathbb{R}^{n_q}$ is represented by a linear combination of the state. The origin is an equilibrium state for (1), that is, $\phi(0) = 0$.

We use the following assumptions.

Assumption 1: The matrices B and G are known, and G has a full column rank. The matrix C_q and function ϕ are unknown. The matrix A is unknown. □

Remark 1: The assumption on knowledge of B and G is mild. Indeed, the input matrix B is known classically in many ADP frameworks [3], [38] and G can be assumed to be the identity if the vector field through which the nonlinearity acts is unknown. □

We require the following definition to describe the class of nonlinearities considered in this article.

Definition 1: A function $f : \mathbb{X} \to \mathbb{R}^{n_x}$ is Lipschitz continuous in the domain $\mathbb{X} \subset \mathbb{R}^{n_f}$ if

$$\|f(x_1) - f(x_2)\| \leq \mathfrak{L}_f \|x_1 - x_2\| \tag{2}$$

for some $\mathfrak{L}_f > 0$ and all $x_1, x_2 \in \mathbb{X}$. We define the scalar

$$\mathfrak{L}_f^* = \inf_{\mathbb{R}_+} \big\{ \mathfrak{L}_f : \text{condition (2) holds} \big\} \tag{3}$$

as the Lipschitz constant of f in $\mathbb{X}$. A function is globally Lipschitz if (2) holds for $\mathbb{X} \equiv \mathbb{R}^{n_f}$. □

Assumption 2: The nonlinearity ϕ is globally Lipschitz continuous, that is

$$\|\phi(q_1) - \phi(q_2)\| \leq \mathfrak{L}_\phi^* \|q_1 - q_2\| \tag{4}$$

for any $q_1, q_2 \in \mathbb{R}^{n_q}$, and the global Lipschitz constant $\mathfrak{L}_\phi^*$ is unknown. □

Remark 2: While the matrix A is typically known in practice or can easily be estimated from data by using standard system identification methods, we do not require that A is known. Instead, one can choose any matrix A_0 of appropriate dimensions such that (A_0, B) is a stabilizable pair and use our proposed method on the effective nonlinearity $G\phi(q) + (A - A_0)x$, which is Lipschitz, since $G\phi$ and $(A - A_0)x$ are each Lipschitz. □

Given a control policy $u(x)$, we define an infinite-horizon cost functional given an initial state $x_0 \in \mathbb{R}^{n_x}$ as

$$\mathcal{J}(x_0, u) = \sum_{t=0}^{\infty} \gamma^t \mathcal{U}(x_t, u(x_t)) \tag{5}$$

where $\mathcal{U}$ is a function with nonnegative range, $\mathcal{U}(0, 0) = 0$, and $\{x_k\}$ denotes the sequence of states generated by the closed-loop system

$$x_{t+1} = Ax_t + Bu(x_t) + G\phi\big(C_q x_t\big). \tag{6}$$

The scalar $\gamma \in (0, 1]$ is a forgetting/discount factor intended to enable the cost to be emphasized more by current state and control actions and lend less credence to the past.

Before formally stating our objective, we need to introduce the following standard definition [1].

Definition 2: A continuous control policy $u(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ is admissible on $X \subset \mathbb{R}^{n_x}$ if $u(0) = 0$ and $u(\cdot)$ stabilizes the closed-loop system (6) on X and $\mathcal{J}(x_0, u)$ is finite for any $x_0 \in X$. □

We want to design an optimal control policy that achieves the optimal cost

$$\mathcal{J}_\infty(x_0) = \inf_{u \in \mathfrak{U}} \mathcal{J}(x_0, u) \tag{7}$$

for any $x_0 \in \mathbb{R}^{n_x}$. Here, $\mathfrak{U}$ denotes the set of all admissible control policies. In other words, we wish to compute an optimal control policy

$$u_\infty = \arg\inf_{u \in \mathfrak{U}} \mathcal{J}(x_0, u). \tag{8}$$

Directly constructing such an optimal controller is very challenging for general nonlinear systems; this is further complicated because system (1) contains unmodeled/uncertain dynamics. Therefore, we shall use ADP: a class of iterative, data-driven algorithms that generate a convergent sequence of control policies whose limit is provably the optimal control policy $u_\infty(x)$.

Recall from [38] and [39] that a necessary condition for convergence of policy iteration methods (a subclass of ADP) is the availability of an initial admissible control policy $u_0(x)$, which is nontrivial to derive for systems with some uncertain dynamics. Therefore, our objective in this article is to systematically derive an initial admissible control policy using only partial model information via kernelized Lipschitz learning and semidefinite programming. We also extend this idea to handle the case when the control input is constrained. In such cases, along with an admissible controller, we also derive a domain of attraction of the controller within which the control policy is guaranteed to satisfy input constraints and the closed-loop system remains stable. We refer to the derivation of admissible control policies with guaranteed stabilizability and/or constraint enforcement as safe initialization for ADP: a crucial property required for ADP algorithms to gain traction in expensive industrial applications.

We invoke the assumption in [21] and [22] regarding the availability of legacy/archival/historical data generated by the

system during prior experiments, that is, at design time, we have a data set $\mathcal{D}$ consisting of unique triples: state-input pairs along with the corresponding state update information. Concretely, we have access to $\mathcal{D} = \{x_j, u_j, x_j^+\}_{j=1}^N$. For each $\{x_j, u_j, x_j^+\} \in \mathcal{D}$, we estimate the nonlinear term using (1), that is

$$\phi(q_j) = G^\dagger \left(x_j^+ - Ax_j - Bu_j \right) \tag{9}$$

where $G^\dagger$ exists by Assumption 1. Note that we also need to estimate the matrix C_q [see (1)] so that q_j can be calculated from x_j. While estimating the exact elements of these matrices is quite challenging, we can estimate the nonzero elements in the matrices, which is enough to design safe initial control policies, because the exact elements of C_q will be subsumed within the Lipschitz constant.

Remark 3: The problem of estimating the sparsity pattern of C_q is analogous to the problem of feature selection and sparse learning, known as automatic relevance determination (ARD) [40]. The basic idea in ARD is to give feature weights some parametric prior densities; these densities are subsequently refined by maximizing the likelihood of the data [40], [41]. For example, one can define hyperparameters that explicitly represent the relevance of different inputs to a machine learning algorithm with respect to the desired output (e.g., a regression problem). These relevance hyperparameters determine the range of variation of parameters relating to a particular input. ARD can then determine these hyperparameters during learning to discover which inputs are relevant. □

We need the following assumption on the data $\{q_j, \phi(q_j)\}$, without which one cannot attain the global Lipschitz constant of the nonlinearity $\phi(\cdot)$ with high accuracy.

Assumption 3: Let $\mathcal{Q}$ denote the convex hull of the samples $\{q_j\}$. The Lipschitz constant of $\phi(\cdot)$ in the domain $\mathcal{Q}$ is identical to the global Lipschitz constant $\mathfrak{L}_\phi^*$. □

Assumption 3 ensures that the samples obtained from the archival data are contained in a subregion of $\mathbb{R}^{n_q}$ where the nonlinearity $\phi(\cdot)$'s local Lipschitz constant is the same as its global Lipschitz constant.

Example 1: Suppose $\phi(q) = 1.5\sin(q)$. As long as the convex hull of the samples $\{q\}$ contains zero, the Lipschitz constant of ϕ on the convex hull $\mathcal{Q}$ and on $\mathbb{R}$ is identical. □

In Section III, we will leverage the data set $\mathcal{D}$ to estimate the Lipschitz constant of $\phi(\cdot)$ using the kernelized Lipschitz learning/estimation and consequently design an initial admissible linear control policy $u_0 = K_0 x$ via semidefinite programs. We will demonstrate how such an initial admissible linear control policy fits into a neural-network-based ADP formulation (such as policy iteration) to asymptotically generate the optimal control policy $u_\infty(x)$.

Remark 4: The control algorithm proposed in this article is a direct data-driven controller because no model of $\phi(\cdot)$ is identified in the controller design step. □

Remark 5: Although we focus only on discrete-time systems, our results hold for continuous-time systems with slight modifications. □

Remark 6: If $n_\phi > 1$, our proposed Lipschitz learning algorithm will yield n_ϕ Lipschitz constant estimates, one for each dimension of $\phi(\cdot)$. To avoid notational complications, we proceed (without loss of generality) with $n_\phi = 1$. For larger n_ϕ, our algorithm can be used componentwise. □

III. Kernelized Lipschitz Learning

In this section, we provide a brief overview of KDE and provide a methodology for estimating Lipschitz constants from data.

A. Empirical Density of Lipschitz Estimates

With the data $\{\phi(q_j), q_j\}_{j=1}^N$, we obtain $n \in \mathbb{N}$ underestimates of the global Lipschitz constant $\mathfrak{L}_\phi^*$ using

$$\ell_{jk} = \frac{|\phi(q_j) - \phi(q_k)|}{\|q_j - q_k\|} \tag{10}$$

where $k \in \{1, \ldots, N\} \backslash j$. The sequence $\{\ell_{jk}\}$ is empirical sample drawn from an underlying univariate distribution L. Note that the true distribution L has finite support: its support is bounded below (componentwise) by zero since all $\ell_{jk} \geq 0$ and bounded above by the true Lipschitz constant $\mathfrak{L}_\phi^*$. This leads us to the key idea of our approach that is to identify the support of the distribution L to yield an estimate of the true Lipschitz constant of $\phi(\cdot)$.

Remark 7: Variants of the estimator (10), such as $\max_k \ell_{jk}$, have been widely used in the literature to construct algorithms for determining Lipschitz constants (see [35], [36], [42]). □

In the literature, common methods of tackling the support estimation problem are by assuming prior knowledge about the exact density of Lipschitz estimates [42] or using Strongin overestimates of the Lipschitz constant [36]. However, we avoid these overestimators because they are sometimes unreliable, even for globally Lipschitz functions [37, Th. 3.1]. Instead, we try to fit the density directly from local estimates and the data in a nonparametric manner using the KDE and characteristics of the estimated density.

B. Plug-in Support Estimation

With a set of n underestimates $\{\ell_r\}_{r=1}^n$, we generate an estimate $\hat{L}_n$ of the true density L using a kernel density estimator

$$\hat{L}_n(\ell) = \frac{1}{nh_n} \sum_{r=1}^{n} \mathcal{K}\left(\frac{\ell - \ell_r}{h_n} \right) \tag{11}$$

where $\mathcal{K} : \mathbb{R} \to \mathbb{R}$ is a smooth function called the kernel function and $h_n > 0$ is the kernel bandwidth. A plug-in estimate of the support S of the true density L is

$$\hat{S}_n := \left\{ \ell \in \mathbb{R}_{\geq 0} | \hat{L}_n(\ell) \geq \beta_n \right\} \tag{12}$$

where β_n is an element of a sequence $\{\beta_n\}$ that converges to zero as $n \to \infty$; this plug-in estimator was proposed in [43].

C. Implementation

Implementing the plug-in estimator involves first constructing a KDE of L with n samples. Then, if one picks $\beta \equiv \beta_n$ small enough, one can easily compute $\hat{S}$ from (12). Then

$$\hat{\mathfrak{L}}_\phi := \max\left(\hat{S}_n \right). \tag{13}$$

Algorithm 1 Kernelized Lipschitz Estimation

Require: Initial data set, $\{x_k, \phi(C_q x_k)\}_{k=1}^N$
Require: Confidence parameter, $0 < \beta \ll 1$
1: $\{q_k, \phi(q_k)\} \leftarrow$ Estimate C_q via ARD
2: **for** k in $1, \ldots, N$ **do**
3: **for** j in $\{1, \ldots, N\} \setminus k$ **do**
4: $\ell \leftarrow$ append ℓ_{jk} computed by (10)
5: $\hat{L}_n \leftarrow$ KDE with cross-validated $\mathcal{K}$ and h using $\{\ell_r\}$
6: $\hat{S}_n \leftarrow$ compute using (12)
7: $\hat{\mathfrak{L}}_\phi \leftarrow \max(\hat{S}_n)$.

This is a very straightforward operation with the availability of tools, such as `ksdensity` (MATLAB) and the `KernelDensity` tool in `scikit-learn` (Python). The pseudocode is detailed herein in Algorithm 1.

Remark 8: Note that the true support S is a subset of $\mathbb{R}_{\geq 0}$. Therefore, when computing the density estimate, this information should be fed into the tool being used. For example, in MATLAB, one has the option {`'support'`, `'positive'`}. Essentially, this subroutine transforms the data into the log scale and estimates the log density so that, upon returning to linear scale, one preserves positivity. □

D. Theoretical Guarantees

We formally describe the density L. We consider that the samples $q \in \mathcal{Q}$ are drawn according to some probability distribution μ_0 with support $\mathbb{X}$. For any set S, suppose that μ_0 can be written as $\mu_0(S) = \int_S \Omega(q)\, \mathrm{d}\mu(q)$, where μ is the Lebesgue measure and Ω is continuous and positive on $\mathbb{X}$. Let μ_X denote the product measure $\mu_0 \times \mu_0$ on $\mathbb{X} \times \mathbb{X}$. Since μ_0 is absolutely continuous with respect to the Lebesgue measure, μ_X assigns zero mass on the diagonal $\{(q, q) : q \in \mathbb{X}\}$. The cumulative distribution function for L is then given by

$$\tilde{L}(\lambda) = \mu_X\left(\left\{(q_1, q_2) : q_1 \neq q_2, \frac{|\phi(q_1) - \phi(q_2)|}{\|q_1 - q_2\|} \leq \lambda\right\}\right).$$

Since $\tilde{L}$ is nondecreasing, L exists almost everywhere by Lebesgue's theorem for differentiability of monotone functions, and L's support is contained within $[0, \mathfrak{L}_\phi^*]$ because of (10).

We investigate the worst case sample complexity involved in overestimating $\mathfrak{L}_\phi^*$ under the following mild assumption.

Assumption 4: The nonlinearity $\phi(\cdot)$ is twice continuously differentiable, that is, $\phi(\cdot) \in \mathcal{C}^2$. □

Lemma 1: Suppose that Assumptions 3 and 4 hold. Then, there exists some $q^* \in \mathcal{Q}$ such that $\|\nabla\phi(q^*)\| = \mathfrak{L}_\phi^*$.

Proof: Suppose that $\{(q_1^k, q_2^k)\}_{k=1}^\infty$ denotes a sequence of paired samples in $\mathcal{Q}$ such that $|\phi(q_1^k) - \phi(q_2^k)|/\|q_1^k - q_2^k\| \to \mathfrak{L}_\phi^*$ as $k \to \infty$. Since $\mathcal{Q}$ is the convex hull of finitely many samples, it is compact, so we can choose a subsequence of $\{q_1^k, q_2^k\}_{k=1}^\infty$ that converges to (q_1^∞, q_2^∞) where both limits are in $\mathcal{Q}$. If $q_1^\infty = q_2^\infty$, then a Taylor expansion estimate implies $\|\nabla\phi(q_1^\infty)\| \geq \mathfrak{L}_\phi^*$. Since $\mathfrak{L}_\phi^*$ is an upper bound of $\|\nabla\phi\|$ at any sample in $\mathcal{Q}$, $\|\nabla\phi(q_1^\infty)\| = \mathfrak{L}_\phi^*$ and $q^* = q_1^\infty$. If $q_1^\infty \neq q_2^\infty$, then the result follows by applying the mean value theorem to

$$\varphi(t) = \phi\big(q_1^\infty + t\big(q_2^\infty - q_1^\infty\big)\big) - \phi\big(q_1^\infty\big)$$

for $t \in [0, 1]$, for which $\varphi(0) = 0$ and $\varphi(1) = \mathfrak{L}_\phi^*\|q_1^\infty - q_2^\infty\|$. Also, $\mathrm{d}\varphi/\mathrm{d}t = (\nabla\phi(q_1^\infty + t(q_2^\infty - q_1^\infty)))^\top(q_2^\infty - q_1^\infty)$. Since $\|\nabla\phi\| \leq \mathfrak{L}_\phi^*$, this implies $|\mathrm{d}\varphi/\mathrm{d}t| \leq \mathfrak{L}_\phi^*\|q_2^\infty - q_1^\infty\|$. Reordering q_1^∞ and q_2^∞ if needed, we have

$$\mathfrak{L}_\phi^*\big\|q_2^\infty - q_1^\infty\big\| = \varphi(1) = \int_0^1 (\mathrm{d}\varphi/\mathrm{d}t)\, \mathrm{d}s \leq \mathfrak{L}_\phi^*\big\|q_2^\infty - q_1^\infty\big\|.$$

Hence, the rightmost inequality must be an equality, which implies that $d\varphi(s)/dt = \mathfrak{L}_\phi^*\|q_1^\infty - q_2^\infty\|$ for all s, that is, if the Lipschitz constant is attained with $q_1^\infty \neq q_2^\infty$, then $\phi(\cdot)$ restricted to the segment connecting q_1^∞ and q_2^∞ is linear with slope $\mathfrak{L}_\phi^*$. This concludes the proof. ■

Lemma 1 enables the worst case complexity result described in the following theorem.

Theorem 1: Let $\varphi'(q_1, q_{-1}) = |\phi(q_1) - \phi(q_{-1})|/\|q_1 - q_{-1}\|$, and suppose that Assumptions 3 and 4 hold. There exists $C_0 > 0$ such that for all sufficiently small $\delta > 0$ and any set $\{q_j\}_{j=1}^n$ of n uniform random samples in $\mathbb{X}$, the probability that some pair $q_+, q_- \in \{q_j\}$ gives the Lipschitz estimate

$$\phi'(q_+, q_-) \geq (1 - \delta)\mathcal{L}_\phi^* - C_0\delta$$

is at least $1 - \epsilon(n, \delta)$. Here, $\epsilon(n, \delta) \leq 3\exp(-n\kappa\delta^{2\ n_q - 1})$, where κ is a constant depending on n_q.

Proof: By Lemma 1, there exists at least one q^* such that $\|\nabla\phi(q^\star)\| = \mathfrak{L}_\phi^*$. For the worst case analysis, suppose that this occurs only at a single sample, $q^\star$. A Taylor expansion at $q^\star$ yields

$$\phi\big(q^\star + q\big) = \phi\big(q^\star\big) + \nabla\phi\big(q^\star\big)^\top q + \mathfrak{R}(q) \tag{14}$$

where $\mathfrak{R}$ is a remainder term with $|\mathfrak{R}(q)| \leq C_{\mathfrak{R}}\|q\|^2$ when $\|q\| \leq \eta$, for some $C_{\mathfrak{R}} > 0$ and $\eta > 0$. Note that

$$\nabla\phi(q^\star)^\top q = \|\nabla\phi(q^\star)\|\|q\|\cos\theta$$

where θ is the angle between $\nabla\phi(q^\star)$ and q. To obtain a good estimate of $\|\nabla\phi(q^\star)\|$, one needs to sample two points in this neighborhood: one point q_+ with $\cos\theta \approx 1$ and a second point q_- with $\cos\theta \approx -1$. Each of these conditions defines a cone. Regarding one of these cones in cylindrical coordinates $0 \leq \ell \leq \eta$ and $\|y\| \leq \chi\ell$ for $\chi = \tan\theta$, we can integrate the $n_q - 1$ dimensional volume to get the volume of this cone as $C_0\chi^{n_q - 1}\eta^{n_q}$ for some dimension-dependent constant C_0. A calculation shows that $\cos\theta \geq 1 - \chi^2/2$ for small θ. With q_+ and q_- as one sample from each cone, we have that $q_+ - q_-$ is contained in the cone $\|y\| \leq \chi\ell$, and so we can use (14) to approximate

$$\begin{aligned}
&\big|\phi\big(q^\star + q_+\big) - \phi\big(q^\star + q_-\big)\big| \\
&\quad = \Big|\nabla\phi\big(q^\star\big)^T(q_+ - q_-) + \mathfrak{R}(q_+) - \mathfrak{R}(q_-)\Big| \\
&\quad \geq \big\|\nabla\phi\big(q^\star\big)\big\|\|q_+ - q_-\|\left(1 - \frac{\chi^2}{2}\right) - |\mathfrak{R}(q_+) - \mathfrak{R}(q_-)|.
\end{aligned}$$

Dividing by $\|q_+ - q_-\|$ and using the defining property of $q^\star$ gives

$$\phi'(q_+, q_-) \geq \left(1 - \frac{\chi^2}{2}\right)\mathcal{L}_\phi^* - \frac{|\mathfrak{R}(q_+) - \mathfrak{R}(q_-)|}{\|q_+ - q_-\|}. \tag{15}$$

Subsequently, one can decompose $q_+ = q_+^\parallel + q_+^\perp$, with $q_+^\parallel$ parallel to $\nabla\phi(q^\star)$ and $q_+^\perp$ perpendicular and satisfying

$\|q_+^\perp\| \leq \chi|q_+^\parallel|$. Identical arguments can be used to infer $\|q_-^\perp\| \leq \chi|q_-^\parallel|$, and hence, $\|q_+ - q_-\| \geq \|q_+^\parallel - q_-^\parallel\| - \|q_+^\perp - q_-^\perp\|$. Since q_+ and q_- are chosen from opposite cones, we have $\|q_+^\parallel - q_-^\parallel\| = \|q_+^\parallel\| + \|q_-^\parallel\|$. Using $\|q^\perp\| \leq \chi\|q^\parallel\|$ and $\cos\theta \geq 1 - \chi^2/2$, we have $\|q_+ - q_-\| \geq (\|q_+\| + \|q_-\|)(1-\chi)(1-\chi^2/2)$. Hence

$$\begin{aligned}\frac{|\mathfrak{R}(q_+) - \mathfrak{R}(q_-)|}{\|q_+ - q_-\|} &\leq \frac{C_{\mathfrak{R}}\big(\|q_+\|^2 + \|q_-\|^2\big)}{(\|q_+\| + \|q_-\|)(1-\chi)\big(1-\chi^2/2\big)} \\ &\leq \frac{2\ C_{\mathfrak{R}} \max\{\|q_+\|, \|q_-\|\}^2}{\max\{\|q_+\|, \|q_-\|\}(1-\chi)\big(1-\chi^2/2\big)} \\ &\leq \frac{2\ C_{\mathfrak{R}}\eta\big(1+\chi^2\big)^{1/2}}{(1-\chi)\big(1-\chi^2/2\big)}.\end{aligned}$$

By combining the aforementioned inequality with (15), one obtains

$$\varphi'(q_+, q_-) \geq \Big(1 - \frac{\chi}{2}\Big)\mathfrak{L}_\phi^* - \frac{2\ C_{\mathfrak{R}}\eta\big(1+\chi^2\big)^{1/2}}{(1-\chi)\big(1-\chi^2/2\big)}.$$

Set $\delta = \chi/2$ and take $\eta = \delta$. Then, there exists $C_0 > 0$ such that for all sufficiently small δ

$$\varphi'(q_+, q_-) \geq (1-\delta)\mathfrak{L}_\phi^* - C_0\delta, \tag{16}$$

which implies that $\varphi' \to \mathfrak{L}_\phi^*$ as $\delta \to 0$.

From the assumption on uniformly drawn samples, the probability of sampling in one of the (χ, η) cones is

$$\int_0^\eta \kappa(\chi r)^{n_q - 1}\,\mathrm{d}r = \frac{\kappa\chi^{n_q-1}\eta^{n_q}}{n_q}$$

for some $\kappa > 0$ that depends on n_q. Using $\delta = \eta = \chi/2$ and absorbing the factors of 2 and $1/n_q$ into κ yields $\kappa\delta^{2n_q-1}$.

Let $\mathfrak{X}_{1\pm}$ be the event of sampling at least one point in the (χ, η) cone as above, and let $\mathfrak{X}_{0\pm}$ be the event of sampling nothing in the (χ, η) cone. The probability of sampling at least one of each of the points q_+ and q_- just described is

$$\begin{aligned}1 - P(\mathfrak{X}_{0+} \cap \mathfrak{X}_{0-}) - P(\mathfrak{X}_{0+} \cap \mathfrak{X}_{1-}) - P(\mathfrak{X}_{1+} \cap \mathfrak{X}_{0-}) \\ \geq 1 - \big(1 - 2\kappa\delta^{2\ n_q-1}\big)^n - 2\big(1 - \kappa\delta^{2\ n_q-1}\big)^n\end{aligned}$$

where the factor 2 before κ in the second term comes from the fact that both cones are excluded and they are disjoint, and the 2 before the third term comes by combining the final two terms in the first expression.

By using the fact that for any $\epsilon' \in (0,1)$ and $n > 0$, the inequality $(1-\epsilon')^n \leq \exp(-n\epsilon')$ holds, we can conclude that

$$\begin{aligned}1 - P(\mathfrak{X}_{0+} \cap \mathfrak{X}_{0-}) - P(\mathfrak{X}_{0+} \cap \mathfrak{X}_{1-}) - P(\mathfrak{X}_{1+} \cap \mathfrak{X}_{0-}) \\ \geq 1 - \exp\big(-2\ n\kappa\delta^{n_q-1}\big) - 2\exp\big(-n\kappa\delta^{n_q-1}\big) \geq 1 - \epsilon\end{aligned}$$

for any given $\epsilon > 0$. The latter can be ensured by choosing n large enough. This gives a lower bound on the probability of obtaining (16) and, hence, the desired result. ■

TABLE I

KERNELIZED LIPSCHITZ LEARNING OF BENCHMARK FUNCTIONS

n	$\log_{10}\beta$	Time [s]	$\hat{\mathfrak{L}}_\phi$ (mean $\pm$ stdev)	OE‡?
$\phi_1 = \lvert\cos(\pi x)\rvert$, $\mathfrak{L}_\phi^* = 3.141$ on $[-\pi, \pi]$				
100	-2	0.596 $\pm$ 0.127	3.361 $\pm$ 0.093	✓
100	-4	0.610 $\pm$ 0.122	3.528 $\pm$ 0.177	✓
500	-2	2.463 $\pm$ 0.432	3.252 $\pm$ 0.083	✓
500	-4	2.438 $\pm$ 0.427	3.369 $\pm$ 0.158	✓
$\phi_2 = x - x^3/3$, $\mathfrak{L}_\phi^* = 1.000$ on $[-1, 1]$				
100	-2	0.455 $\pm$ 0.123	1.018 $\pm$ 0.011	✓
100	-4	0.459 $\pm$ 0.114	1.030 $\pm$ 0.020	✓
500	-2	1.656 $\pm$ 0.218	1.005 $\pm$ 0.004	✓
500	-4	1.585 $\pm$ 0.208	1.010 $\pm$ 0.009	✓
$\phi_3 = \sin(x) + \sin(2x/3)$, $\mathfrak{L}_\phi^* = 1.667$ on $[3.1, 20.4]$				
100	-2	0.556 $\pm$ 0.121	1.780 $\pm$ 0.074	✓
100	-4	0.547 $\pm$ 0.124	1.923 $\pm$ 0.166	✓
500	-2	1.826 $\pm$ 0.224	1.684 $\pm$ 0.010	✓
500	-4	1.821 $\pm$ 0.221	1.720 $\pm$ 0.002	✓
$\phi_4 =$ Hansen test function from [35], $\mathfrak{L}_\phi^* = 8.378$ on $[0, 1]$				
100	-2	0.450 $\pm$ 0.117	8.969 $\pm$ 0.262	✓
100	-4	0.488 $\pm$ 0.123	9.401 $\pm$ 0.476	✓
500	-2	1.921 $\pm$ 0.138	8.507 $\pm$ 0.046	✓
500	-4	1.923 $\pm$ 0.130	8.707 $\pm$ 0.103	✓
$\phi_5 = \max\{1 - 3\sin(x), \exp(-\sin(x))\}$, $\mathfrak{L}_\phi^* = 3.0$ on $[-10, 10]$				
100	-2	0.546 $\pm$ 0.061	3.139 $\pm$ 0.051	✓
100	-4	0.612 $\pm$ 0.066	3.200 $\pm$ 0.087	✓
500	-2	1.893 $\pm$ 0.245	3.043 $\pm$ 0.014	✓
500	-4	1.989 $\pm$ 0.230	3.104 $\pm$ 0.024	✓

‡ OE indicates an overestimate of the true Lipschitz constant, that is, $\min\{\hat{\mathfrak{L}}_\phi\} > \mathfrak{L}_\phi^*$.

E. Benchmarking the Lipschitz Estimator

Our Lipschitz estimator is tested on well-studied benchmark examples studied previously in [32] and [35]; the benchmark functions are described in Table I along with their domains and true local Lipschitz constants. Note that all the functions are not globally Lipschitz (e.g., ϕ_2), not differentiable everywhere (e.g., ϕ_1 and ϕ_4), and, in the special case of ϕ_4, specifically constructed to ensure that naive overestimation of $\mathfrak{L}_\phi^*$ using Strongin methods provably fails [37]. To evaluate the proposed Lipschitz estimator, we vary the number of data points N and the confidence parameter β. Over 100 runs, we report mean $\pm$ one standard deviation of the following quantities: the time required by our learning algorithm, the estimated Lipschitz constant $\hat{\mathfrak{L}}_\phi$, and the error $\hat{\mathfrak{L}}_\phi - \mathfrak{L}_\phi^*$ (which should be positive when we overestimate $\mathfrak{L}_\phi^*$).

The final column of Table I reveals an important empirical detail; all our estimates of $\mathfrak{L}_\phi^*$ are overestimates for $\beta \leq 0.01$ and $n \geq 100$. This is a critical advantage of our proposed approach because an overestimate will enable us to provide stability and constraint satisfaction guarantees about the data-driven controller, in Section IV. Furthermore, the estimation error is small for every test run, and as expected, the error increases as β decreases because a smaller value of β indicates the need for greater confidence, which results in more conservative estimates.

IV. SAFE INITIALIZATION VIA LEARNED MULTIPLIER MATRICES

In this section, we begin by reviewing a general ADP procedure and then explain how to safely initialize unconstrained,

as well as input-constrained ADP. A key insight is that estimating the Lipschitz constant is equivalent to learning a multiplier matrix in a purely data-driven manner. This enables the use of semidefinite programs to compute safe and admissible initial control policies for ADP.

A. Unconstrained ADP

Recall the optimal value function given by (7) and the optimal control policy (8). From the Bellman optimality principle, we know that the discrete-time Hamilton–Jacobi–Bellman (HJB) equations are given by

$$J_\infty(x_t) = \inf_u (\mathcal{U}(x_t, u(x_t)) + \gamma\, J_\infty(x_{t+1})) \tag{17a}$$

$$u_\infty(x_t) = \arg\inf_u (\mathcal{U}(x_t, u(x_t)) + \gamma\, J_\infty(x_{t+1})) \tag{17b}$$

where $J_\infty(x_t)$ is the optimal value function and $u_\infty(x_t)$ is the optimal control policy. The key operations in ADP methods [38] involve setting an admissible control policy $u_0(x)$ and then iterating the policy evaluation step

$$\mathcal{J}_{k+1}(x_t) = \mathcal{U}(x_t, u_k(x_t)) + \gamma\, \mathcal{J}_{k+1}(x_{t+1}) \tag{18a}$$

and the policy improvement step

$$u_{k+1}(x_t) = \arg\min_{u(\cdot)} (\mathcal{U}(x_t, u(x_t)) + \gamma\, \mathcal{J}_{k+1}(x_{t+1})) \tag{18b}$$

until convergence.

1) Semidefinite Programming for Safe Initial Control Policy: Recall the following definition.

Definition 3: The equilibrium point $x = 0$ of the closed-loop system (6) is globally exponentially stable with a decay rate α if there exist scalars $C_0 > 0$ and $\alpha \in (0, 1)$ such that $\|x_t\| \leq C_0 \alpha^{(t-t_0)} \|x_0\|$ for any $x_0 \in \mathbb{R}^{n_x}$. □

Conditions for global exponential stability (GES) of the equilibrium state, adopted from [28], are provided next.

Lemma 2: Let $V(\cdot, \cdot) : [0, \infty) \times \mathbb{R}^{n_x} \to \mathbb{R}$ be a continuously differentiable function such that

$$\gamma_1 \|x\|^2 \leq V(t, x_t) \leq \gamma_2 \|x\|^2 \tag{19a}$$

$$V(t+1, x_{t+1}) - V(t, x_t) \leq -\left(1 - \alpha^2\right) V(t, x_t) \tag{19b}$$

for any $t \geq t_0$ and $x \in \mathbb{R}^{n_x}$ along the trajectories of the system

$$x^+ = \varphi(x) \tag{20}$$

where γ_1, γ_2, and α are positive scalars, and $\varphi(\cdot)$ is a nonlinear function. Then, the equilibrium state $x = 0$ for system (20) is GES with decay rate α. □

The following design theorem provides a method to construct an initial linear stabilizing policy $u_0(x) = K_0 x$ such that the origin is a GES equilibrium state of the closed-loop system (6).

Theorem 2: Suppose that Assumptions 1 and 2 hold and that there exist matrices $P = P^\top \succ 0 \in \mathbb{R}^{n_x \times n_x}$, $K_0 \in \mathbb{R}^{n_u \times n_x}$, and scalars $\alpha \in (0, 1)$, $\nu > 0$ such that

$$\Psi + \Gamma^\top \mathcal{M} \Gamma \preceq 0 \tag{21}$$

where

$$\Psi = \begin{bmatrix} (A + BK_0)^\top P (A + BK_0) - \alpha^2\, P & \star \\ G^\top P (A + BK_0) & G^\top P G \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} C_q & 0 \\ 0 & I \end{bmatrix}, \quad \text{and } \mathcal{M} = \begin{bmatrix} \nu^{-1} \left(\mathfrak{L}_\phi^*\right)^2 I & 0 \\ 0 & -\nu^{-1} I \end{bmatrix}.$$

Then, the equilibrium $x = 0$ of the closed-loop system (6) is GES with decay rate α.

Proof: Let $V = x^\top P x$. Then, (19a) in Lemma 2 is satisfied with $\gamma_1 = \lambda_{\min}(P)$ and $\gamma_2 = \lambda_{\max}(P)$. Let $\Delta V = V^+ - V$. Note that

$$\begin{aligned} V^+ &= \left(x^+\right)^\top P x^+ \\ &= ((A + BK_0)x + G\phi)^\top P ((A + BK_0)x + G\phi) \\ &= x^\top (A + BK_0)^\top P (A + BK_0) x \\ &\quad + 2x^\top (A + BK_0)^\top P G \phi + \phi^\top G^\top P G \phi. \end{aligned}$$

Therefore

$$\begin{aligned} \begin{bmatrix} x \\ \phi \end{bmatrix}^\top \Psi \begin{bmatrix} x \\ \phi \end{bmatrix} &= x^\top (A + BK_0)^\top P (A + BK_0) x - \alpha^2\, x^\top P x \\ &\quad + 2x^\top (A + BK_0)^\top P G \phi + \phi^\top G^\top P G \phi \\ &= V^+ - \alpha^2\, V = \Delta V + \left(1 - \alpha^2\right) V \end{aligned}$$

and

$$\begin{bmatrix} x \\ \phi \end{bmatrix}^\top \Gamma^\top \mathcal{M} \Gamma \begin{bmatrix} x \\ \phi \end{bmatrix} = \begin{bmatrix} q \\ \phi \end{bmatrix}^\top \mathcal{M} \begin{bmatrix} q \\ \phi \end{bmatrix} = \nu \left(\left(\mathfrak{L}_\phi^*\right)^2\, q^\top q - \phi^\top \phi \right).$$

Thus, premultiplying and postmultiplying (21) with $\begin{bmatrix} x & \phi \end{bmatrix}^\top$ and its transpose, respectively, we get

$$\Delta V + \left(1 - \alpha^2\right) V + \nu \left(\left(\mathfrak{L}_\phi^*\right)^2\, q^\top q - \phi^\top \phi \right) \leq 0.$$

By inequality (4) in Assumption 2 and recalling that $\phi(0) = 0$, we get $(\mathfrak{L}_\phi^*)^2\, q^\top q - \phi^\top \phi \geq 0$. Since $\nu > 0$, this implies $\Delta V + (1 - \alpha^2) V \leq 0$, which is identical to (19b). ■

Note that we do not need to know $\phi(\cdot)$ to satisfy conditions (21). Instead, Theorem 2 provides the conditions that leverage matrix multipliers similar to those described in [29]. Note that the estimation of the Lipschitz constant is the key step in learning a multiplier matrix $\mathcal{M}$ since this matrix is parameterized solely by $\mathfrak{L}_\phi^*$ and ν is an optimization variable.

We shall now provide LMI-based conditions for computing the initial control policy K_0 and the initial domain of attraction P and ν via convex programming.

Theorem 3: Fix $\alpha \in (0, 1)$ and let $\hat{\mathfrak{L}}_\phi$ be obtained via (13). If there exist matrices $S = S^\top \succ 0$, Y, and a scalar $\nu > 0$ such that the LMI conditions

$$\begin{bmatrix} -\alpha^2\, S & \star & \star & \star \\ 0 & -\nu I & \star & \star \\ AS + BY & GS & -S & \star \\ \hat{\mathfrak{L}}_\phi C_q S & 0 & 0 & -\nu I \end{bmatrix} \preceq 0 \tag{24}$$

are satisfied, then the matrices $K_0 = YS^{-1}$, $P = S^{-1}$, and scalar ν satisfy conditions (21) with the same α and $\mathfrak{L}_\phi^*$ replaced by $\hat{\mathfrak{L}}_\phi$.

Proof: A congruence transformation of (24) with the matrix blkdiag ($[P\ \nu^{-1}I\ P\ I]$) and substituting S with P^{-1} and Y with K_0P^{-1} yields

$$\left[\begin{array}{cc|cc} -\alpha^2 P & \star & \star & \star \\ 0 & -\nu^{-1}I & \star & \star \\ \hline A+BK_0 & G & -P & \star \\ \hat{\mathfrak{L}}_\phi C_q & 0 & 0 & -\nu I \end{array}\right] \preceq 0.$$

Taking the Schur complement with the submatrices shown by the guidelines in the abovementioned inequality, we get (22), as shown at the bottom of the page. Since $\nu > 0$, taking the Schur complement again yields (23), as shown at the bottom of this page, which can be rewritten as

$$\Psi - \begin{bmatrix} 0 \\ I \end{bmatrix} \nu^{-1} I \begin{bmatrix} 0 \\ I \end{bmatrix}^\top + \begin{bmatrix} C_q \\ 0 \end{bmatrix} \left(\hat{\mathfrak{L}}_\phi\right)^2 \nu^{-1} I \begin{bmatrix} C_q \\ 0 \end{bmatrix}^\top \preceq 0$$

which is exactly (21) if $\hat{\mathfrak{L}}_\phi = \mathfrak{L}_\phi^*$. Thus, conditions (21) and (24) are equivalent when $\hat{\mathfrak{L}}_\phi = \mathfrak{L}_\phi^*$. ■

Based on Theorem 1, we will get a perfect estimate of the Lipschitz constant only with infinite data, which is impractical. However, a much more practical result is provided next. This is based on the observation that our proposed kernelized Lipschitz learner typically provides the overestimates of $\mathfrak{L}_\phi^*$. In such cases, that is, when $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, an advantage of our method is that any feasible solution of the LMI (24) is guaranteed to be an admissible control policy. This is demonstrated by the following result.

Theorem 4: Let (P, K_0, ν, α) be a feasible solution to conditions (21) with an overestimate of the Lipschitz constant $\hat{\mathfrak{L}}_\phi > \mathfrak{L}_\phi^*$. Then, (P, K_0, ν, α) is also a feasible solution to conditions (21).

Proof: Let $\delta L = \hat{\mathfrak{L}}_\phi - \mathfrak{L}_\phi^*$. Since $\hat{\mathfrak{L}}_\phi$ is an overestimator of $\mathfrak{L}_\phi^*$, $\delta L > 0$. Since (P, K, ν, α) is a feasible solution to (21) with $\hat{\mathfrak{L}}_\phi$, it satisfies

$$\Psi + \Gamma^\top \begin{bmatrix} -\nu^{-1}\left(\mathfrak{L}_\phi^* + \delta L\right)^2 I & 0 \\ 0 & I \end{bmatrix} \Gamma \preceq 0$$

which can be written as

$$\Psi + \Gamma^\top \mathcal{M} \Gamma + \Gamma^\top \underbrace{\begin{bmatrix} -\nu^{-1}\left(2\mathfrak{L}_\phi^* \delta L + \delta L^2\right) I & 0 \\ 0 & 0 \end{bmatrix}}_{:=\delta\mathcal{M}} \Gamma \preceq 0.$$

As $\nu > 0$, we infer that $\delta\mathcal{M} \preceq 0$ and, hence, $\Gamma^\top \delta\mathcal{M} \Gamma \preceq 0$. Therefore, $\Psi + \Gamma^\top \mathcal{M} \Gamma \preceq 0$. Since the other conditions in (21) are independent of $\mathfrak{L}_\phi^*$, the other conditions are automatically satisfied. This concludes the proof. ■

Theorem 4 indicates that if our learned $\hat{\mathfrak{L}}_\phi$ is an overestimate of $\mathfrak{L}_\phi^*$, and we use $\hat{\mathfrak{L}}_\phi$ to obtain a safe stabilizing control policy, then this is also a safe stabilizing control policy for the true system (1). Having a feasible solution to (21) with an underestimator of $\mathfrak{L}_\phi^*$ is not sufficient to guarantee a feasible solution for the true Lipschitz constant because $\delta\mathcal{M}$ may not be negative semi-definite in that case. Of course, extremely conservative overestimates of $\hat{\mathfrak{L}}_\phi$ will result in conservative control policies or result in infeasibility. In our proposed approach, we have observed that the confidence parameter β dictates the conservativeness of the overestimate, that is, $\beta \to 1$ makes the estimate $\hat{\mathfrak{L}}_\phi$ more conservative.

2) Safely Initialized PI: We begin by proving the following critical result.

Theorem 5: Let $\mathcal{U}(x, u)$ be defined as in (5). If K_0 is obtained by solving (24) for $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, then the initial control policy $u_0 = K_0 x$ is an admissible control policy on $\mathbb{R}^{n_x}$.

Proof: Clearly, u_0 is continuous and (by Theorems 2 and 3) is a stabilizing control policy for (1). It remains to show that the cost induced by u_0 is finite. Since u_0 is stabilizing and $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, we know that $\|x_t\| \to 0$ as $t \to \infty$, which implies $u_0 \to 0$ and, therefore, $\mathcal{U}(x_t, u_t) \to 0$ as $t \to \infty$. Since $\mathcal{U}(x_t, u_t)$ converges to a finite limit, $\mathcal{U}(x_t, u_t)$ is bounded for all $t \geq 0$. Therefore, any partial sum $\sum_{t=0}^{t'} \mathcal{U}(x_t, u_t)$ is bounded and monotonic, that is, $\mathcal{J}$ converges to a finite limit. ■

Admissibility of u_0 for the specific linear quadratic regulator (LQR) cost function follows directly from Theorem 5.

Corollary 1: Let

$$\mathcal{U}(x_t, u_t) = x_t^\top Q x_t + u_t^\top R u_t \tag{25}$$

for some matrices $Q = Q^\top \succeq 0$ and $R = R^\top \succ 0$. Then, the initial control policy $u_0 = K_0 x$ obtained by solving (24) is an admissible control policy on $\mathbb{R}^{n_x}$. □

Now, we know that $u_0 = K_0 x$ is an admissible control policy, and we are ready to proceed with the policy iteration steps (18). Typically, an analytical form of $\mathcal{J}_k$ is not known *a priori*, so we resort to a shallow neural approximator/truncated basis expansion for fitting this function, assuming that $\mathcal{J}_k$ is smooth for every $k \in \mathbb{N} \cup \{\infty\}$. Concretely, we represent the value function and cost functions as

$$\mathcal{J}_k(x) := \omega_k^\top \psi(x) \tag{26}$$

where $\psi_0(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_0}$ denotes the set of differentiable basis functions (equivalently, hidden-layer neuron activations) and $\omega : \mathbb{R}^{n_0}$ is the corresponding column vector of basis coefficients (equivalently, neural weights).

It is not always clear how to initialize the weights of the neural approximators (26). Commonly, small random numbers drawn from a uniform distribution are used [44], but there is no safety guarantee associated with random initialization. An alternative initialization method used in the literature employs proportional–integral–derivative (PID)

$$\begin{bmatrix} -\alpha^2 P & 0 \\ 0 & -\nu^{-1} I \end{bmatrix} + \begin{bmatrix} (A+BK_0)^\top & C_q^\top \\ G^\top & 0 \end{bmatrix} \begin{bmatrix} P & 0 \\ 0 & \hat{\mathfrak{L}}_\phi^2 \nu^{-1} I \end{bmatrix} \begin{bmatrix} (A+BK_0)^\top & C_q^\top \\ G^\top & 0 \end{bmatrix}^\top \preceq 0 \tag{22}$$

$$\begin{bmatrix} (A+BK_0)^\top P (A+BK_0) - \alpha^2 P & (A+BK_0)^\top P G \\ G^\top P (A+BK_0) & -\nu^{-1} I + G^\top P G \end{bmatrix} + \begin{bmatrix} C_q \\ 0 \end{bmatrix} \left(\hat{\mathfrak{L}}_\phi\right)^2 \nu^{-1} I \begin{bmatrix} C_q \\ 0 \end{bmatrix}^\top \preceq 0 \tag{23}$$

controllers [45]. While PID can be tuned to stabilize systems without a model, it is considerably harder to enforce constraints while ensuring that stability and regions of constraint satisfaction are not considered during initialization. In addition, many PID initialization frameworks in the literature are used for data generation and collection; these data are then used to generate neural weights by least-squares fitting the neural approximator. While these neural weights are indeed the optimal weights based on the collected data, the induced control policy is rarely guaranteed to be constraint-satisfying or even stable.

To address these issues, we propose initializing the weights as follows. Since our initial Lyapunov function is quadratic, we include the quadratic terms of the components of x to be on the basis $\psi(x)$. Then, we can express the initial Lyapunov function $x^\top P x$ obtained by solving (24) with appropriate weights in $\psi(x)$, setting all other weights to be zero. With the approximator initialized as above, the policy evaluation step (18a) is replaced by

$$\omega_{k+1}^\top(\psi(x_t) - \gamma\,\psi(x_{t+1})) = \mathcal{U}(x_t, u_k(x_t)) \tag{27a}$$

from which one can solve for ω_{k+1} recursively via

$$\omega_{k+1} = \omega_k - \eta_k \varphi_k\big(\omega_k^\top \varphi_k - \mathcal{U}(x_t, u_k(x_t))\big)$$

where η_k is a learning rate parameter that is usually selected to be an element from the sequence $\{\eta_k\} \to 0$ as $k \to \infty$, and $\varphi_k = \psi(x_t) - \gamma\,\psi(x_{t+1})$. Subsequently, the policy improvement step (18b) is replaced by

$$u_{k+1} = \arg\min_{u(\cdot)} \big(\mathcal{U}(x_t, u(x_t)) + \gamma\,\omega_{k+1}^\top \psi(x_{t+1})\big).$$

This minimization problem is typically nonconvex and, therefore, challenging to solve to optimality. In some specific cases, one of which is that the cost function is quadratic as described in (25), the policy improvement step becomes considerably simpler to execute, namely

$$u_{k+1}(x) = -\frac{\gamma}{2} R^{-1} B^\top \nabla\psi(x)^\top \omega_{k+1}. \tag{27b}$$

This can be evaluated as R and B are known, and ψ is differentiable and chosen by the user, so $\nabla\psi$ is computable. A pseudocode for ease of implementation is provided in Algorithm 2.

Since we prove that u_0 is an admissible control policy, we can use arguments identical to [44] to claim that if the optimal value function and the optimal control policy are dense in the space of functions induced by the basis function expansions (26), then the weights of the neural approximator employed in the PI steps (27) converge to the optimal weights, that is, the optimal value function $\mathcal{J}_\infty$ and the optimal control policy u_∞ are achieved asymptotically. This is encapsulated in the following theorem.

Theorem 6: Let $\mathcal{J}_k$ and u_k be obtained by the updates (27). If K_0 is obtained by solving (24) for $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, then we have that the value function $\mathcal{J}_k(x) \to \mathcal{J}_\infty(x)$ and the control policy $u_k(x) \to u_\infty(x)$ as $k \to \infty$, where $\mathcal{J}_\infty$ and u_∞ are optimal solutions of the discrete-time HJB equations described in (17).

Proof: This follows from Theorem 5 and [44, Th. 3.2]. ■

Algorithm 2 Safely Initialized PI for Discrete-Time Systems

Require: Termination condition constant ϵ_{ac}
Require: Historical data $\mathcal{D}$
1: Estimate Lipschitz constant $\hat{\mathfrak{L}}_\phi$ using Algorithm 1
Require: Compute stabilizing control gain K_0 via SDP (24)
2: Fix admissible control policy $u_0(x) = K_0 x$
3: **while** $\|\mathcal{J}_k - \mathcal{J}_{k-1}\| \geq \epsilon_{\text{ac}}$ **do**
4: Solve for the value $\mathcal{J}_k(x)$ using

$$\mathcal{J}_{k+1}(x_t) = \mathcal{U}(x_t, u_k(x_t)) + \gamma\,\mathcal{J}_{k+1}(x_{t+1}).$$

5: Update the control policy $u_{(k+1)}(x)$ using

$$u_{k+1}(x_t) = \arg\min_{u(\cdot)} (\mathcal{U}(x_t, u_k(x_t)) + \gamma\,\mathcal{J}_{k+1}(x_{t+1})).$$

6: $k := k + 1$

B. Input-Constrained ADP With Safety Enhancements

Herein, we tackle the case when the control input is to be constrained, which is very common in practical applications. We make the following assumption on the constraints.

Assumption 5: The control input $u \in \mathbb{U}$, where

$$\mathbb{U} = \big\{u \in \mathbb{R}^{n_u} : \xi_i^\top u \leq 1\big\} \tag{28}$$

for $i = 1, \ldots, n_c$, where n_c is the number of input constraints, and $\xi_i \in \mathbb{R}^{n_u}$ for every i. □

Remark 9: The matrix inequality (28) defines a polytopic input constraint set. Clearly, constraints of the form $|u| \leq \bar{u}$ can be written as

$$\begin{bmatrix} \xi_i \\ \xi_{i+1} \end{bmatrix} u = \begin{bmatrix} 0 & \cdots & 1/\bar{u} & \cdots & 0 \\ 0 & \cdots & -1/\bar{u} & \cdots & 0 \end{bmatrix} u \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

which is of the form (28). □

Note that with any control policy $u_0 = K_0 x$, the constraint set described in (28) is equivalent to the set

$$\mathcal{X} = \big\{x \in \mathbb{R}^{n_x} : \xi_i^\top K_0 x \leq 1\big\} \tag{29}$$

for $i = 1, \ldots, n_c$. Before we state the main design theorem, we require the following result from [46, p. 69].

Lemma 3: The ellipsoid

$$\mathcal{E}_P = \big\{x \in \mathbb{R}^{n_x} : x^\top P x \leq 1\big\} \tag{30a}$$

is a subset of $\mathcal{X}$ if and only if

$$\xi_i K_0^\top P^{-1} K_0 \xi_i^\top \leq 1 \tag{30b}$$

for $i = 1, \ldots, n_c$. □

1) Constrained Admissible Initial Control Policy and Invariant Set Estimation: Since the control input is constrained, we need to characterize an invariant set of the form $\mathcal{E}_P$ within which all control actions satisfy (28) and the following stability certificate holds.

Definition 4: The equilibrium point $x = 0$ of the closed-loop system (6) is locally exponentially stable with a decay rate α and a domain of attraction $\mathcal{E}_P$ if there exist scalars $C_0 > 0$ and $\alpha \in (0, 1)$ such that $\|x_t\| \leq C_0 \alpha^{(t-t_0)} \|x_0\|$ for any $x_0 \in \mathcal{E}_P$. □

A standard result for testing local exponential stability of the equilibrium point adopted from [47] is provided next.

Lemma 4: Let $V : [0, \infty) \times \mathcal{E}_P \to \mathbb{R}$ be a continuously differentiable function such that the inequalities (19) hold for any $t \geq t_0$ and $x \in \mathcal{E}_P$ along the trajectories of system (6), where γ_1, γ_2, and α are positive scalars. Then, the equilibrium $x = 0$ for system (6) is locally exponentially stable with a decay rate α and a domain of attraction $\mathcal{E}_P$. □

The following design theorem provides a method to construct a stabilizing policy such that the origin is a locally exponentially stable equilibrium of the closed-loop system and constraint satisfaction is guaranteed within a prescribed ellipsoid $\mathcal{E}_P \subset \mathcal{X}$ without knowing the nonlinearity $\phi(\cdot)$.

Theorem 7: Fix $\alpha \in (0, 1)$ and $\hat{\mathfrak{L}}_\phi$. Suppose that $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, and there exist matrices $S = S^\top \succ 0$, Y, and a scalar $\nu > 0$ such that the LMI conditions (24) and

$$\begin{bmatrix} 1 & \xi_i^\top Y \\ \star & S \end{bmatrix} \succeq 0 \tag{31}$$

for every $i = 1, \ldots, n_c$. Then, the equilibrium $x = 0$ of the closed-loop system (6) is locally exponentially stable with a decay rate α and a domain of attraction $\mathcal{E}_P$ defined in (30a). Furthermore, given that the initial state $x_0 \in \mathcal{E}_P$, then the control actions u_t satisfy the constraints (28) for all $t \geq 0$.

Proof: From Theorem 2, we know that (19) holds. Taking Schur complements of (31) yields (30b), which, by Lemma 3, implies that $\mathcal{E}_P \subset \mathcal{X}$ and, hence, the input constraints are satisfied for all $t \geq 0$ by the closed-loop system with policy $u = K_0 x$ because $x_0 \in \mathcal{E}_P$. Thus, all the conditions of Lemma 4 are satisfied, which concludes the proof. ■

Remark 10: Note that conditions (24) and (31) are LMIs in S, Y, and ν for a fixed $\hat{\mathfrak{L}}_\phi$ value. Therefore, one can maximize the volume of $\mathcal{E}_P$ by solving a constrained convex program with cost function $-\log|S|$ (the log determinant of S) subject to the constraints (24) and (31) while line searching for α. This will reduce the conservativeness of the domain of attraction. □

2) Safely Initialized Input-Constrained PI: By adopting the work of [48]–[52] for input-constrained/actuator saturated ADP, we choose a cost function of the form

$$\mathcal{U}(x, u) = Q(x) + 2\int_0^u \left(\bar{u}\tanh^{-1}(\upsilon/\bar{u})\right)^\top R \, \mathrm{d}\upsilon \tag{32}$$

where $Q(x) : \mathbb{R}^{n_x} \to \mathbb{R}$ is a positive-definite function satisfying $Q(0) = 0$ and $R \succ 0$.

We begin by demonstrating that the constrained policy is an admissible policy on its domain of attraction.

Theorem 8: Let $\mathcal{U}$ be defined as in (32). Then, the initial control policy $u_0 = K_0 x$ obtained by solving (24) and (31) is an admissible control policy on $\mathcal{E}_P$.

Proof: By definition $Q(0) = 0$. Also, the integrand in (32) is zero when the upper limit is zero. Therefore, $\mathcal{U}(0, 0) = 0$. For any $x_0 \in \mathcal{E}_P$, u_0 is a stabilizing constrained control policy, and therefore, $\|x_t\| \to 0$ and $\|u_t\| \to 0$ as $t \to \infty$. Hence, $\mathcal{U} \to 0$ as $t \to \infty$. The rest of the proof follows identically as in the proof of Theorem 5. ■

Since the control policy is constrained, we can initialize ADP safely using the neural approximator (26), as discussed in Section IV-B1. The policy evaluation step is given by

$$\omega_{k+1}^\top(\psi(x_t) - \gamma\,\psi(x_{t+1})) = Q(x_t) + 2\int_0^{u_k(x_t)} \left(\bar{u}\tanh^{-1}(\upsilon/\bar{u})\right)^\top R \, \mathrm{d}\upsilon. \tag{33a}$$

Some algebraic manipulation yields

$$\begin{aligned}\omega_{k+1}^\top(\psi(x_t) - \gamma\,\psi(x_{t+1})) &= Q(x_t) + 2\bar{u}u^\top R\tanh^{-1}(u/\bar{u}) \\ &\quad + \bar{u}^2\mathrm{diag}(R)^\top \begin{bmatrix} \ln\left(1 - u_1^2/\bar{u}^2\right) \\ \ln\left(1 - u_2^2/\bar{u}^2\right) \\ \vdots \\ \ln\left(1 - u_{n_u}^2/\bar{u}^2\right) \end{bmatrix}\end{aligned} \tag{33b}$$

where $u_1,\ u_2,\ u_3, \ldots, u_{n_u}$ are the individual components of the vector u. Subsequently, the policy improvement step is given by

$$u_{k+1} = -\bar{u}\tanh\left(\frac{\gamma}{2\bar{u}} R^{-1} B^\top \nabla\psi(x_{t+1})^\top \omega_{k+1}\right). \tag{33c}$$

Since the initial control policy is constrained and admissible, one can use [49] to prove the convergence of the value function and the control policy to the optimal using the constrained policy iteration steps (33b) and (33c). This is summarized in the following theorem.

Theorem 9: Let $\mathcal{J}_k$ and u_k be obtained by the updates (33). If K_0 is obtained by solving (24) and (31) for $\hat{\mathfrak{L}}_\phi \geq \mathfrak{L}_\phi^*$, then we have that the value function $\mathcal{J}_k(x) \to \bar{\mathcal{J}}_\infty(x)$ and the control policy $u_k(x) \to \bar{u}_\infty(x)$ as $k \to \infty$, where $\bar{\mathcal{J}}_\infty$ and $\bar{u}_\infty$ are the optimal value function and optimal constrained control policy, respectively. Furthermore, $u_k \in \mathbb{U}$ for every $k \in \mathbb{N}_+$.

Proof: This follows directly from Theorem 8 and [49, Th. 2]. The constraint admissibility of u_0 is a direct consequence of (31), and all subsequent policies $u_k \in \mathbb{U}$ because of the update (33c) since $\|\tanh(\cdot)\|_\infty \leq 1$. ■

C. Discussion

The value iteration algorithm (see Algorithm 3) does not generally require an admissible control policy in order to converge optimally using data. Although this is true in off-policy implementations (that is, when the updated control policy is not used online), in on-policy implementations, a lack of stabilizing initial policies could result in unsafe transient behavior unless the underlying system is open-loop stable, leading to unsafe exploration during the initial data collection phase. Of course, if the underlying system is stable or bounded, then the VI algorithm does not require an admissible initial control policy.

Q-learning is a provably convergent direct optimal adaptive control algorithm and model-free reinforcement learning technique [53]–[56]. Q-learning can be used to find an optimal action-selection policy based on measurements of the previous state and action observations controlled using a suboptimal policy. In most of the existing works, the reward/cost function is manipulated to guarantee the correction of the unsafe actions in the learning phase. Our proposed method does not require a corrective modification of the reward/cost function online for safety. Instead, historical data and solving SDPs based on

Algorithm 3 Safely Initialized VI for Discrete-Time Systems

Require: Termination condition constant ϵ_{ac}

Require: Historical data $\mathcal{D}$

1: Estimate Lipschitz constant $\hat{\mathfrak{L}}_\phi$ using Algorithm 1

Require: Compute stabilizing control gain K_0 via SDP (24)

2: Fix safe initial control policy $u_0(x) = K_0 x$

3: **while** $\|\mathcal{J}_k - \mathcal{J}_{k-1}\| \geq \epsilon_{\text{ac}}$ **do**

4: Solve for the value $\mathcal{J}_k(x)$ using

$$\mathcal{J}_{k+1}(x_t) = \mathcal{U}(x_t, u_k(x_t)) + \gamma\, \mathcal{J}_k(x_{t+1}).$$

5: Update the control policy $u_{(k+1)}(x)$ using

$$u_{k+1}(x_t) = \arg\min_{u(\cdot)} \left(\mathcal{U}(x_t, u_k(x_t)) + \gamma\, \mathcal{J}_{k+1}(x_{t+1})\right).$$

6: $k := k + 1$

Lipschitz estimation are used to generate safe control policies that enable safe data collection during on-policy Q-learning implementation because the states are guaranteed not to diverge with the initial policy (conversely, this divergence could occur if the initial policy was unsafe).

V. Numerical Examples

A. Example 1: Nonlinear Torsional Pendulum

We demonstrate our proposed approach using the torsional pendulum, which is modeled by discretizing the system

$$\dot{\theta} = \omega \tag{34a}$$

$$J\dot{\omega} = u - Mgl\sin\theta - f_d\omega \tag{34b}$$

with mass $M = 0.333$ kg, length $l = 0.667$ m, acceleration due to gravity $g = 0.981$ m/s^2, friction factor $f_d = 0.2$, and moment of inertia $J = 0.1975$ kg·m^2. With Euler discretization and a sampling time of $\tau = 0.01$ s, we get a discrete-time model of the form (1) with

$$x = \begin{bmatrix}\theta\\ \omega\end{bmatrix},\quad A = I + \tau\begin{bmatrix}0 & 1\\ 0 & -f_d\end{bmatrix},\quad B = \tau\begin{bmatrix}0\\1\end{bmatrix},\quad G = \tau\begin{bmatrix}0\\-1\end{bmatrix}.$$

We assume that the nonlinearity $\phi = Mgl\sin\theta/J$ is completely unknown; clearly, $\phi(\cdot)$ has a Lipschitz constant $\mathfrak{L}_\phi^* = Mgl/J = 11.038$, which is also unknown to us.

In the data collection phase, we initialize system (34) from ten different initial conditions in the space $[-\pi, \pi] \times [-2, 2]$ and collect data each 0.1 s, leading to a total data set of $N = 50$ samples. Note that the initialization procedure mentioned in [49] requires 400 data points, which is considerably more than ours, and in that procedure, the original policy in the pretraining phase is not guaranteed to be admissible. ARD reveals that the nonlinearity only acts through the second state, and the argument of the nonlinearity is $q = \theta$. Proceeding as in Algorithm 1, we perform cross validation using an Epanechnikov kernel with bandwidth $h_n = 0.05$ and choose $\beta = 0.01$. This yields the overestimate $\hat{\mathfrak{L}}_\phi = 11.511 > \mathfrak{L}_\phi^*$. Using this Lipschitz estimate, we solve (24) with $\alpha = 0.95$ and $\nu = 1$ for an initial value function $x^\top P x$ and control policy estimate $K_0 x$.

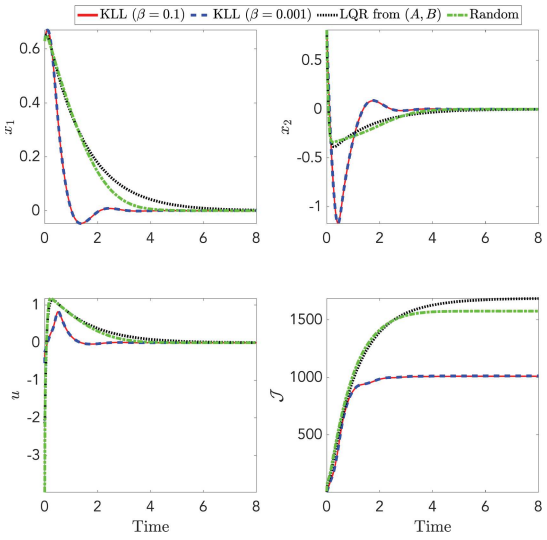

Fig. 1. Comparison of states x_t, inputs u_t, and cost function values for unconstrained ADP with safe initialization for Lipschitz estimates with increasing confidence $\hat{\mathfrak{L}}_\phi(\beta = 0.1) = 11.14$ and $\hat{\mathfrak{L}}_\phi(\beta = 0.001) = 12.29$. We also compare this article to an LQR controller that is known to work for linear systems.

We construct a $2 - 11 - 1$ (input layer-hidden layer-output layer) value function neural approximator with a set of polynomial basis functions

$$\psi(x_1, x_2) = \left\{\frac{x_1^2}{2}, \frac{x_2^2}{2}, x_1x_2, \frac{x_1^2x_2}{2}, \frac{x_1x_2^2}{2}, \frac{x_1^4}{4}, \frac{x_2^4}{4}, \frac{x_1^3}{3}, \frac{x_2^3}{3}, \frac{x_1^2x_2^2}{2}, \frac{x_1^4x_2^4}{4}\right\} \tag{35}$$

where x_1 and x_2 denote the first and second components of x, respectively. Our initial weight vector is set to

$$\omega_0 = \begin{bmatrix}2P_{11} & 2P_{22} & P_{12} + P_{21} & 0 & \cdots & 0\end{bmatrix}^\top$$

where P_{ij} is the (i, j)th element of P. We fix the learning rate at $\eta = 10^{-4}$ and the forgetting factor $\gamma = 0.95$.

1) Unconstrained Scenario: We first test the unconstrained scenario, where the cost function is $\sum \|Qx\|_1 + u^\top Ru$, with $Q = I_2$ and $R = 0.5$, and compare four initial policies and value functions obtained through: 1) kernelized Lipschitz learning with $\beta = 0.1$; 2) kernelized Lipschitz learning with $\beta = 0.001$; 3) solving an algebraic Riccati equation and ignoring the nonlinearity; and 4) randomly initializing with small weights from a normal distribution with small variance and zero mean as in [44], which is the most common initializer. The comparison study results are shown in Fig. 1. We observe that all the methods listed earlier work and result in stabilizing control policies that result in the state of the torsional pendulum to converge to its equilibrium. Interestingly, both the Lipschitz constant estimates result in similar trajectories, implying that the SDPs (24) are not extremely sensitive to the Lipschitz estimate. However, based on the $\mathcal{J}$ subplot that shows the variation of $\sum x^\top Qx + u^\top Ru$ with time, there is a slight improvement of performance in the $\beta = 0.1$ (continuous red line) case compared to the $\beta = 10^{-3}$ (blue dashed line) case since the Lipschitz estimate in the former is closer to

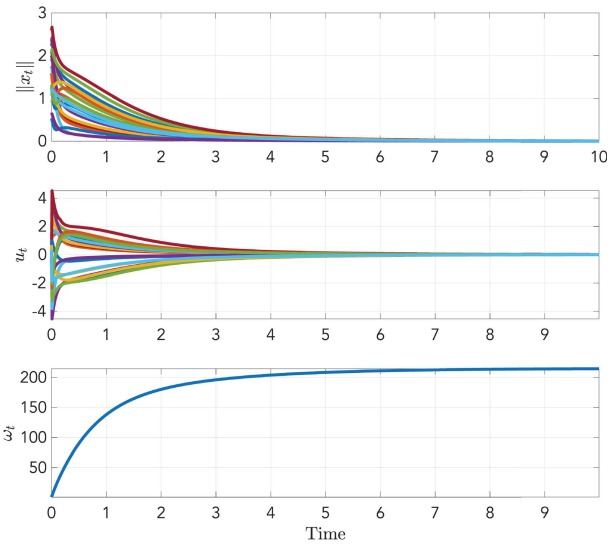

Fig. 2. Illustration of constrained-input value-iteration-based ADP with safe initialization. The top plot shows the variation of $\|x\|$ over time. The middle plot demonstrates that input constraints are satisfied for all t, and the bottom plot demonstrates that the initial control policy was close to the optimal, but learning was necessary to change the weights to the optimal values.

the true Lipschitz constant. As expected, the cost incurred by the control policy ignoring the nonlinearity (black dotted line) is by far the worst since the control actions required early on are of larger magnitude and the tracking performance is severely compromised. Randomly selecting weights also results in worse performance than our proposed method, as the cost incurred is increased due to oscillatory behavior in the states and poor tracking in the initial time frame. Summarily, this experiment demonstrates the effectiveness of the proposed approach and its robustness to Lipschitz estimate conservatism.

In Fig. 2, we demonstrate the on-policy value iteration algorithm with safe initialization. All initial conditions converge to the origin using our proposed approach. In contrast, randomly initializing a policy and value as is typical in on-policy value iteration results in the states initially diverging (not shown in the plot) and poor performance before the rank condition is reached for determining a least-squares solution to update the neural weights.

2) Constrained Scenario: We also test the scenario where the control actions are constrained by $|u| \leq 1$. In this case, we use the cost functional defined in (32) with $\bar{u} = 1$, $Q = I_2$, and $R = 0.5$. We begin by solving (24) and (31) with $\nu = 1$ and $\alpha = 0.95$ to get P and K_0, as in Section V-A1. We also select the same basis functions (35). We randomly initialize (using 20 random initial conditions) system (34) from within the domain of attraction of the initial control policy, that is, from within the set $\{x^\top P x \leq 1\}$. We know from Theorem 7 that this ensures that the initial control policy will satisfy input constraints. Consequently, because the policy improvement step is also guaranteed to satisfy input constraints and the initial policy is stabilizing, there are no constraint violations, and the initialization is deemed safe. The performance of the proposed algorithm is provided in Fig. 3. The convergence of $\|x_t\|$ to zero and the satisfaction of input bounds are illustrated.

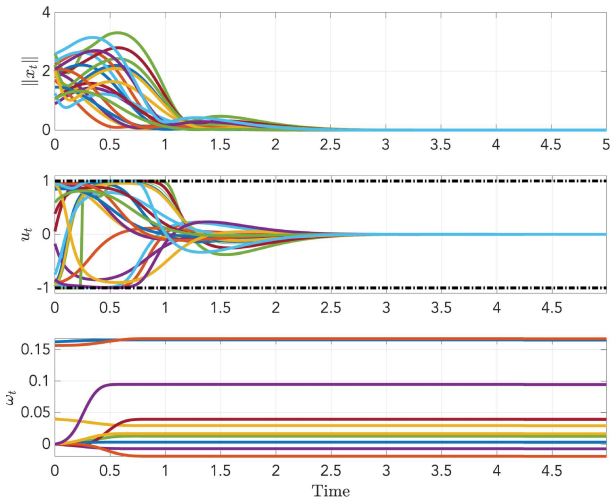

Fig. 3. Illustration of constrained-input policy-iteration-based ADP with safe initialization. The top plot shows the variation of $\|x\|$ over time. The middle plot demonstrates that input constraints are satisfied for all t, and the bottom plot demonstrates that the initial control policy was close to the optimal, but learning was necessary to change the weights to the optimal values.

TABLE II
MODEL PARAMETERS AND NOMINAL VALUES

Param	Value	Param	Value	Param	Value
δ_0	0.2351	ω_0	100π	E_{q0}	1.7827
H	8.74	x_d	1.0354	x_d'	0.36
x_q	0.7194	V_s	1.00	x_T	0.041
x_L	0.0485	P_m	0.7	E_{f0}	3.00
η_1	0.4495	η_2	0.8089	V_{s0}	0.4942

Finally, we demonstrate the convergence of the neural weights ω_t, noting that learning did occur, that is, the weights were not static throughout the simulation (which would indicate that the initial policy was optimal).

B. Example 2: Excitation Control of Hydrogenerator

Developing excitation control systems without complete model information is an important open challenge in smart grids and multimachine power systems [57]. To this end, we demonstrate the potential of our proposed framework on a practical system: a 300-MW hydrogenerator in the Northeastern China power grid. A complete description of the power grid topology and its components is provided in [57]. A state-space model of the hydrogenerator dynamics is given by

$$\begin{aligned}
\dot{\delta} &= \omega - \omega_0 \\
\frac{H}{\omega_0}\dot{\omega} &= P_m - \frac{V_s}{\eta_1}E_q \sin\delta - \frac{D}{\omega_0}(\omega - \omega_0) - V_{s0}\sin\ 2\delta \\
T_{d0}\dot{E}_q &= E_f - \left(1 + \frac{x_d - x_d'}{\eta_1}\right)E_q + \frac{x_d - x_d'}{\eta_1}V_s\cos\delta
\end{aligned}$$

where $V_{s0} = V_s^2(x_q - x_d')/2\eta_1\eta_2$, $\eta_1 = x_d' + x_T + x_L$, and $\eta_2 = x_q + x_T + x_L$. The equilibrium state of this model is given by $x_\infty := [\delta_0, \omega_0, E_{q0}]^\top$ with the corresponding equilibrium control $u_\infty := E_{f0}$. All model parameter values and equilibrium values are provided in Table II; physical meanings of the parameters are discussed in [57].

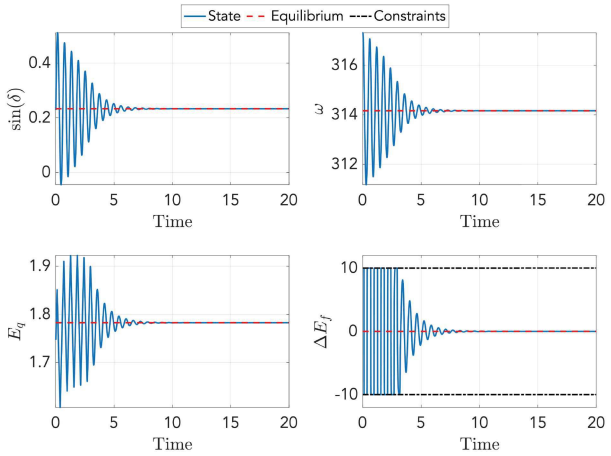

Fig. 4. Temporal variation of states (δ, ω, E_q) and control input (ΔE_f) for Example 2.

Our objective is to design a constrained control action $u = E_f$, where $|u - u_\infty| \leq 10$, such that the state of the system is driven to the equilibrium state x_∞ while minimizing a quadratic cost function given by $(x - x_\infty)^\top Q(x - x_\infty) + R(u - u_\infty)^2$, with $Q = \text{diag}[100, 10000, 100]$ and $R = 1$. With Euler discretization and a sampling time of $\tau = 1$ ms, we get a discrete-time model of the form (1). We choose

$$x = \begin{bmatrix} \delta \\ \omega \\ E_q \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 1 \\ 0 & -0.0034 & 2 \\ 0 & 0 & -0.2402 \end{bmatrix}$$
$$B = \begin{bmatrix} 0 \\ 0 \\ 0.0960 \end{bmatrix}$$

by fitting a linear model to $N = 1000$ data points generated from the true system with small random control actions near the equilibrium control; (A, B) is ensured to be a stabilizable pair. Note that even though the residual model is not globally Lipschitz, we can leverage the sampled data to compute a local Lipschitz constant (in a neighborhood of x_∞) of $\hat{\mathfrak{L}}_\phi = 0.9910$ with 99% confidence using the KDE method. Using this Lipschitz estimate, we solve (24) with $\alpha = 1$ and $\nu = 1$ for an initial value function $x^\top Px$ and control policy estimate $K_0 x$. The initial admissible control is computed to be

$$K_0 = [-997.43 \ 488.01 \ -4999.97].$$

We randomly initialize the system from within 5% of the equilibrium state; this is a considerably more difficult scenario than that studied in [57] because we allow our initial angular frequency to be different from ω_0, which causes strong oscillatory initial behavior and instability if the initial control policy is constructed using small random neural network weights instead of our proposed method. Furthermore, in our cases, different to those studied before, the initial control policy is constrained, making it even more challenging to stabilize the power system.

We run the constrained policy iteration using a neural approximator with basis functions involving polynomials of x up to degree 4, with higher order polynomials having smaller coefficient weights to ensure stable numerical conditioning. A learning rate of 10^{-4} is chosen, and the simulation is run for 50 s. We observe from Fig. 4, the proposed ADP method works well, and the states converge to the equilibrium state without control constraint violation despite severely oscillatory dynamics until $|\omega - \omega_0|$ becomes small.

VI. Conclusion and Future Work

This article provides a methodology for constructing admissible initial control policies for ADP methods using multiplier matrix learning by using KDE and semidefinite programming for a class of nonlinear systems with uncertain Lipschitz dynamics. Such admissible controllers enable safe initialization, that is, with constraint satisfaction using only historical data, which is necessary not only in policy iteration methods but also in value iteration and Q-learning for safely obtaining initial data online for on-policy learning when the underlying system is not open-loop stable. Simulations on a discretized torsional pendulum model and a high-dimensional linear system are provided to show the efficiency of our approach. Future research efforts will focus on more general costs and uncertain nonlinear safety constraints while ensuring feasibility with a high probability in terms of regret.

In addition, while we have not explicitly considered robustness in the article, the robustness properties of optimal controllers and ADP for nonlinear systems have been well studied in the literature (see [6], [58], [59], and the references therein). In addition, a robust initial control policy and a corresponding robust constraint-admissible set can be constructed with multiplier matrix learning as discussed in this article via semidefinite programming; we consign to find the optimal policy with respect to a worst case disturbance (H∞ control or zero-sum game) of this method to future work.

References

[1] K. Vamvoudakis *et al.*, "Autonomy and machine intelligence in complex systems," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2015, pp. 5062–5079.
[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[3] D. Vrabie, K. G. Vamvoudakis, and F. L. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. London, U.K.: Institution of Engineering and Technology, 2013.
[4] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.
[5] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.
[6] D. Wang, H. He, and D. Liu, "Adaptive critic nonlinear robust control: A survey," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3429–3451, Oct. 2017.
[7] Y. Yang, Z. Guo, H. Xiong, D.-W. Ding, Y. Yin, and D. C. Wunsch, "Data-driven robust control of discrete-time uncertain linear systems via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3735–3747, Dec. 2019.
[8] M. Gregor and J. Spalek, "The optimistic exploration value function," in *Proc. IEEE 19th Int. Conf. Intell. Eng. Syst. (INES)*, Sep. 2015, pp. 119–123.
[9] R. I. Brafman and M. Tennenholtz, "R-max—A general polynomial time algorithm for near-optimal reinforcement learning," *J. Mach. Learn. Res.*, vol. 3, pp. 213–231, Oct. 2002.
[10] P. Thomas, G. Theocharous, and M. Ghavamzadeh, "High confidence policy improvement," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2380–2388.

[11] D. K. Jha, M. Zhu, Y. Wang, and A. Ray, "Data-driven anytime algorithms for motion planning with safety guarantees," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 5716–5721.

[12] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, 2015.

[13] Y. Duan *et al.*, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[14] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 22–31.

[15] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A Lyapunov-based approach to safe reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8092–8101.

[16] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.

[17] D. Romeres, M. Zorzi, R. Camoriano, and A. Chiuso, "Online semi-parametric learning for inverse dynamics modeling," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 2945–2950.

[18] D. Romeres, D. K. Jha, A. DallaLibera, B. Yerazunis, and D. Nikovski, "Semiparametrical Gaussian processes learning of forward dynamical models for navigating in a circular maze," in *Proc. Int. Conf. Robot. Automat. (ICRA)*, 2019, pp. 3195–3202.

[19] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," *IEEE Trans. Control Syst. Technol.*, to be published, doi: 10.1109/TCST.2019.2949757.

[20] Y. Jiang, Y. Wang, S. A. Bortoff, and Z.-P. Jiang, "Optimal codesign of nonlinear control systems based on a modified policy iteration method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 2, pp. 409–414, Jan. 2015.

[21] M. Tanaskovic, L. Fagiano, C. Novara, and M. Morari, "Data-driven control of nonlinear systems: An on-line direct approach," *Automatica*, vol. 75, pp. 1–10, Jan. 2017.

[22] D. Piga, S. Formentin, and A. Bemporad, "Direct data-driven control of constrained systems," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1422–1429, Jul. 2018.

[23] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2042–2062, Dec. 2018.

[24] L. Fagiano and C. Novara, "Automatic crosswind flight of tethered wings for airborne wind energy: A direct data-driven approach," *IFAC Proc. Volumes*, vol. 47, no. 3, pp. 4927–4932, 2014.

[25] Z. Wang, Y. Liu, and X. Liu, "$\mathcal{H}_\infty$ filtering for uncertain stochastic time-delay systems with sector-bounded nonlinearities," *Automatica*, vol. 44, no. 5, pp. 1268–1277, 2008.

[26] X. Zhang, P. Niu, Y. Ma, Y. Wei, and G. Li, "Global Mittag-Leffler stability analysis of fractional-order impulsive neural networks with one-side Lipschitz condition," *Neural Netw.*, vol. 94, pp. 67–75, Oct. 2017.

[27] J. A. K. Suykens, J. Vandewalle, and B. De Moor, "An absolute stability criterion for the Lur'e problem with sector and slope restricted nonlinearities," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 45, no. 9, pp. 1007–1009, Sep. 1998.

[28] H. K. Khalil, *Nonlinear Control*. New York, NY, USA: Pearson, 2015.

[29] A. Chakrabarty, M. J. Corless, G. T. Buzzard, S. H. Zak, and A. E. Rundell, "State and unknown input observers for nonlinear systems with bounded exogenous inputs," *IEEE Trans. Autom. Control*, vol. 62, no. 11, pp. 5497–5510, Nov. 2017.

[30] A. Chakrabarty, V. Dinh, M. J. Corless, A. E. Rundell, S. H. Zak, and G. T. Buzzard, "Support vector machine informed explicit nonlinear model predictive control using low-discrepancy sequences," *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 135–148, Jan. 2017.

[31] A. Raha, A. Chakrabarty, V. Raghunathan, and G. T. Buzzard, "Embedding approximate nonlinear model predictive control at ultrahigh speed and extremely low power," *IEEE Trans. Control Syst. Technol.*, early access, Mar. 1, 2019, doi: 10.1109/TCST.2019.2898835.

[32] J.-P. Calliess, "Conservative decision-making and inference in uncertain dynamical systems," Ph.D. dissertation, Dept. Eng. Sci., Univ. Oxford, Oxford, U.K., 2014.

[33] D. Limon, J. Calliess, and J. M. Maciejowski, "Learning-based nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7769–7776, 2017.

[34] X. Xu, B. Acikmese, M. Corless, and H. Sartipizadeh, "Observer-based output feedback control design for systems with incrementally conic nonlinearities," in *Proc. Annu. Amer. Control Conf. (ACC)*, Jun. 2018, pp. 1364–1369.

[35] G. R. Wood and B. P. Zhang, "Estimation of the lipschitz constant of a function," *J. Global Optim.*, vol. 8, no. 1, pp. 91–103, Jan. 1996.

[36] R. G. Strongin, "On the convergence of an algorithm for finding a global extremum," *Eng. Cybern.*, vol. 11, pp. 549–555, 1973.

[37] P. Hansen, B. Jaumard, and S. H. Lu, "On using estimates of Lipschitz constants in global optimization," *J. Optim. Theory Appl.*, vol. 75, no. 1, pp. 195–200, Oct. 1992.

[38] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers," *IEEE Control Syst.*, vol. 32, no. 6, pp. 76–105, Nov. 2012.

[39] K. G. Vamvoudakis, H. Modares, B. Kiumarsi, and F. L. Lewis, "Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online," *IEEE Control Syst. Mag.*, vol. 37, no. 1, pp. 33–52, Feb. 2017.

[40] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Sep. 2001.

[41] W. Chu and Z. Ghahramani, "Preference learning with Gaussian processes," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 137–144.

[42] J.-P. Calliess, "Lipschitz optimisation for Lipschitz interpolation," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 3141–3146.

[43] A. Cuevas and R. Fraiman, "A plug-in approach to support estimation," *Ann. Statist.*, vol. 25, no. 6, pp. 2300–2312, Dec. 1997.

[44] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 621–634, Mar. 2014.

[45] Q. Yang and S. Jagannathan, "Reinforcement learning controller design for affine nonlinear discrete-time systems using online approximators," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 377–390, Sep. 2011.

[46] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, vol. 15. Philadelphia, PA, USA: SIAM, 1994.

[47] V. C. Aitken and H. M. Schwartz, "On the exponential stability of discrete-time systems with applications in observer design," *IEEE Trans. Autom. Control*, vol. 39, no. 9, pp. 1959–1962, Sep. 1994.

[48] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, no. 5, pp. 779–791, May 2005.

[49] Q. Lin, Q. Wei, and B. Zhao, "Optimal control for discrete-time systems with actuator saturation," *Optim. Control Appl. Methods*, vol. 38, no. 6, pp. 1071–1080, Mar. 2017.

[50] H. Zhang, Y. Luo, and D. Liu, "Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints," *IEEE Trans. Neural Netw.*, vol. 20, no. 9, pp. 1490–1503, Sep. 2009.

[51] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1513–1525, Oct. 2013.

[52] K. G. Vamvoudakis, M. F. Miranda, and J. P. Hespanha, "Asymptotically stable adaptive-optimal control algorithm with saturating actuators and relaxed persistence of excitation," *IEEE Trans. Neural Nets. Learn. Sys.*, vol. 27, no. 11, pp. 2386–2398, Nov. 2016.

[53] C. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, May 1992.

[54] J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Mach. Learn.*, vol. 16, no. 3, pp. 185–202, Sep. 1994.

[55] P. Mehta and S. Meyn, "Q-learning and Pontryagin's minimum principle," in *Proc. 48th IEEE Conf. Decis. Control (CDC)*, Dec. 2009, pp. 3598–3605.

[56] K. G. Vamvoudakis, "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Syst. Control Lett.*, vol. 100, pp. 14–20, Feb. 2017.

[57] W. Guo, J. Si, F. Liu, and S. Mei, "Policy approximation in policy iteration approximate dynamic programming for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2794–2807, Jun. 2017.

[58] M. Ikeda and D. Šiljak, "Optimality and robustness of LQ control for nonlinear systems," *Automatica*, vol. 26, no. 3, pp. 499–511, 1990.

[59] H.-N. Wu and B. Luo, "Neural network based online simultaneous policy update algorithm for solving the hji equation in nonlinear $\mathcal{H}_\infty$ control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 12, pp. 1884–1895, Oct. 2012.

Ankush Chakrabarty (Senior Member, IEEE) received the B.E. degree (Hons.) in electrical engineering from Jadavpur University, Kolkata, India, in 2011, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 2016.

He was a Post-Doctoral Fellow with Harvard University, Cambridge, MA, USA. Since 2018, he has been a Researcher with Mitsubishi Electric Research Laboratories (MERL), Cambridge. His research interests are in data-driven control, constrained control, machine learning, and unknown input estimation for applications, including factory automation, buildings, and healthcare.

Devesh K. Jha received the M.S. degrees in mechanical engineering and mathematics from Pennsylvania State, State College, PA, USA, and the Ph.D. degree in mechanical engineering from Pennsylvania State in December 2016.

He is currently a Research Scientist with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. His research interests are in the areas of machine learning, robotics, and deep learning.

Dr. Jha was a recipient of several best paper awards, including the Kalman Best Paper Award from the Dynamic Systems and Control Division, American Society of Mechanical Engineers (ASME), in 2019.

Gregery T. Buzzard received the B.S. degree in computer science and the B.Mus. degree in violin performance, the M.S. degree in mathematics from Michigan State University, East Lansing, MI, USA, and the Ph.D. degree in mathematics from the University of Michigan, Ann Arbor, MI, in 1989, 1991, and 1995, respectively.

He is currently a Professor and the Head of the Department of Mathematics, Purdue University, West Lafayette, IN, USA. His current research interests include dynamical systems, inverse problems in imaging, and methods for approximation and optimization.

Dr. Buzzard is a member of SIAM and a winner of the SIAM Imaging Sciences Best Paper Prize for 2020.

Yebin Wang (Senior Member, IEEE) received the B.Eng. degree in mechatronics engineering from Zhejiang University, Hangzhou, China, in 1997, the M.Eng. degree in control theory and control engineering from Tsinghua University, Beijing, China, in 2001, and the Ph.D. degree in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2008.

He has been with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, since 2009, where he is currently a Senior Principal Research Scientist. From 2001 to 2003, he was a Software Engineer, the Project Manager, and the Research and Development Manager in automation industries. His research interests include nonlinear control and estimation, optimal control, adaptive and learning systems, and their applications, including mechatronic systems.

Kyriakos G. Vamvoudakis (Senior Member, IEEE) was born in Athens, Greece. He received the Diploma degree (a five-year degree, equivalent to a Master of Science) (Hons.) in electronic and computer engineering from the Technical University of Crete, Chania, Greece, in 2006, and the M.S. and Ph.D. degree in electrical engineering from The University of Texas at Arlington, Arlington, TX, USA, in 2008 and 2011, respectively.

From 2012 to 2016, he was a Project Research Scientist with the Center for Control, Dynamical Systems and Computation, University of California at Santa Barbara, Santa Barbara, CA, USA. He was an Assistant Professor with the Kevin T. Crofton Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA, USA, until 2018. He is currently an Assistant Professor with the Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, USA. His research interests include approximate dynamic programming, game theory, cyber–physical security, networked control, smart grid, and safe autonomy.

Dr. Vamvoudakis is a member of the Technical Chamber of Greece and a Senior Member of the American Institute of Aeronautics and Astronautics (AIAA). He is also a member of the IEEE Control Systems Society Conference Editorial Board. He was a recipient of the 2019 ARO YIP Award, the 2018 NSF CAREER Award, and several international awards, including the 2016 International Neural Network Society Young Investigator (INNS) Award, the Best Paper Award for Autonomous/Unmanned Vehicles at the 27th Army Science Conference in 2010, the Best Presentation Award at the World Congress of Computational Intelligence in 2010, and the Best Researcher Award from the Automation and Robotics Research Institute in 2011. He is also an Associate Editor of *Automatica*, the *IEEE Computational Intelligence Magazine*, the *Journal of Optimization Theory and Applications*, *Neurocomputing*, and the IEEE CONTROL SYSTEMS LETTERS. He is a registered Electrical/Computer Engineer (PE).