# Continuous-Time Reinforcement Learning Control: A Review of Theoretical Results, Insights on Performance, and Needs for New Designs

Brent A. Wallace and Jennie Si, *Fellow, IEEE*

***Abstract*— This exposition discusses continuous-time reinforcement learning (CT-RL) for the control of affine nonlinear systems. We review four seminal methods that are the centerpieces of the most recent results on CT-RL control. We survey the theoretical results of the four methods, highlighting their fundamental importance and successes by including discussions on problem formulation, key assumptions, algorithm procedures, and theoretical guarantees. Subsequently, we evaluate the performance of the control designs to provide analyses and insights on the feasibility of these design methods for applications from a control designer's point of view. Through systematic evaluations, we point out when theory diverges from practical controller synthesis. We, furthermore, introduce a new quantitative analytical framework to diagnose the observed discrepancies. Based on the analyses and the insights gained through quantitative evaluations, we point out potential future research directions to unleash the potential of CT-RL control algorithms in addressing the identified challenges.**

***Index Terms*— Adaptive/approximate dynamic programming (ADP), continuous-time (CT), optimal control, policy iteration (PI), reinforcement learning (RL), value iteration (VI).**

## I. Introduction

THE origins of modern approaches to optimal control problems are rooted in the 1960s with the inception of dynamic programming (DP) by Bellman [1]. As the first conceptualization of solving challenging nonlinear control problems using recursive methods readily implementable on digital computers, DP has inspired influential works from numerous authors [2], [3], [4], [5], [6]. While researchers recognize the great potential of optimal control, the central "curse of dimensionality" has plagued the field and limited applications. Reinforcement learning (RL) emerged as a systematic method in the early 1980s [4], [7] with the potential to combat the curse of dimensionality. RL has since become a major breakthrough for addressing key challenges in complex nonlinear control problems. The two original solution approaches to solving Bellman's principle of optimality, namely, the policy iteration (PI) and value iteration (VI) algorithms [4], [8], were developed in the RL setting in the context of Markov decision processes (MDPs), and as a result, many of the historic and current RL results have been developed for MDP problems. The characteristic of these formulations is to treat optimal decision and control problems stochastically in discrete state/action spaces [9].

By contrast, the continuous state/action spaces involved in continuous-time (CT) and discrete-time (DT) dynamical control problems give rise to unique challenges. An important branch of research work on RL for decision and control is covered under the scope of adaptive/approximate DP (ADP) [2], [6], [10], [11], which focuses on using approximation and learning to solve the optimal control problem. The works of Werbos [11], [12], [13] represent some of the earliest and most influential conceptualizations of RL in the controls setting. As has become convention throughout the community, we will henceforth use the terms RL and ADP interchangeably.

Classical approaches to solving optimal control problems [14], [15], [16], [17], [18] developed in the 1960s to 1990s were largely model-based off-line methods. With increasing computational capacity in the 2000s came the transition to partially model-free or model-free RL algorithms—a trend seen in both the DT-RL and CT-RL literature. The survey of the field reveals that four main CT-RL algorithm works are fundamentally central: 1) integral RL (IRL) [19]; 2) synchronous PI (SPI) [20]; 3) robust ADP (RADP) [21]; and 4) CT-VI [22]. We, thus, focus our CT-RL study on these works.

## II. Motivation

### A. DT-RL Theoretical and Application Successes

DT-RL algorithms (cf. [23], [24], [25] for review) have demonstrated excellent stability, convergence, and approximation guarantees. They have also substantively addressed a variety of control design requirements, such as stability robustness [26], input saturation [27], and fault tolerance [28]. Additional representative theoretical works include [29], [30], [31], [32], [33], [34] that are based on the PI framework and [35], [36], [37], [38], [39] that are based on the VI framework. Collectively, these results address important properties of learning convergence, solution optimality, and system stability for DT nonlinear systems.

Successful applications of DT-RL algorithms have provided further validation of RL as, perhaps, the most promising and potentially powerful solution to complex control problems. These include DT deep RL and policy gradient methods [40], [41], [42], [43], [44], as well as deep Q-networks (DQNs) [45]. Deep RL methods have

Manuscript received 19 September 2022; revised 1 December 2022 and 7 February 2023; accepted 13 February 2023. This work was supported in part by the NSF under Grant 1563921, Grant 1808752, and Grant 2211740. The work of Brent A. Wallace was supported in part by the NSF through the Graduate Research Fellowship under Grant 026257-001.

The authors are with the Department of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: bawalla2@asu.edu; si@asu.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2023.3245980.

Digital Object Identifier 10.1109/TNNLS.2023.3245980

successfully tackled problems in robotics applications [46], Atari games [45], and the game of Go [47], [48], to name a few. DT-RL methods have also demonstrated great success in addressing complex continuous state and control problems. These results include energy-efficient data centers [49], aggressive ground robot position control [50], [51], power system stability enhancement [52], [53], [54], industrial process control [55], [56], Apache helicopter stabilization, tracking, and reconfiguration control [57], [58], [59], waste water treatment [60], and wearable robots to enable continuous and stable walking [61], [62], [63], [64], [65], [66].

### *B. Relatively Few CT-RL Theoretical Results and Fewer With Quantitative Evaluations*

On the other hand, CT-RL algorithms [19], [20], [21], [22], [67], [68], [69], [70] have seen fewer further theoretical developments compared to their DT-RL counterparts. A survey of these leading CT-RL methods reveals that most address weight convergence, uniform value/policy approximation, and closed-loop stability. However, results in each of these three areas are often developed at a high level and are only qualitative in nature (e.g., results usually require "sufficiently many" basis functions to approximate the value and policy functions), which limits their practical utility.

A handful of recent works, however, have introduced some ideas different from those discussed above. Kim et al. [71] introduce a semidiscrete version of the Hamilton–Jacobi–Bellman (HJB) equation, which allows for a $Q$-learning algorithm to use data collected in discrete time without discretizing or approximating the system dynamics. The proposed method was tested on some relatively simple OpenAI GYM benchmarks. The method may be promising, but the results still need improvement to be comparable to the state of the art, especially since hyperparameter tuning has a significant impact on the results. Lee and Sutton [72] focus on developing PI-based solutions to solve the HJB equation. Their theoretical results may be comprehensive, but they are based on quite stringent assumptions. A robust optimal control formulation is developed in [73], where it is shown applicable to cart-pole and real pendulum control problems. However, even as a model-based approach, it still faces state distribution mismatch challenges and the associated issue when scaling up to high-dimensional problems. Reference [74] is another model-based approach, which is based on learning arbitrary time differentials of environment dynamics in order to learn optimal policy. The results are limited to when there is no environment noise.

Finally, the diverse performance measures that are vital to the control systems' community are seldom addressed by CT-RL although there are occasional exceptions (e.g., stability robustness in RADP [21]). Typical CT-RL theoretical works usually include only simple examples for demonstration purposes (e.g., linear or second-order nonlinear systems with exact solutions). The absence of partnering follow-up analyses leaves a major gap between theory and application.

### *C. CT-RL Follow-Up Works Incrementally Revolve Around Four Central Algorithms, Still Few Evaluation Studies*

Upon survey of the state of the art in the CT-RL community, we observe two main features.

1) The majority of CT-RL works draw heavily from at least one of the four central works introduced above, including their problem formulation, fundamental assumptions, algorithm procedures, and theoretical results.
2) The majority of CT algorithm citations fall into the following three categories.

The first, and the most numerous, comprises works that make a brief citation to the major CT references without providing a substantive connection to the work under development. For instance, SPI [20] and RADP [21] are mentioned in passing in [75] and [76], respectively.

The second category contains incremental works. For instance, [77] is a result of combing the features of SPI and IRL. The work in [78] develops a system identification recurrent neural network (RNN), but the associated critic and actor tuning laws used subsequent to the RNN learning are taken directly from SPI [20], with the RNN features substituting the usual system dynamics $(f, g)$. Similarly, [79] tunes its critic and actor networks simultaneously with gradient descent-based tuning laws reminiscent of those of SPI [20] and uses similar theoretical machinery. Reference [80] is largely a reproduction of IRL [19] with adding a discount factor to the cost functional.

The third, and least common, category consists of the works that attempt to substantively implement CT-RL algorithms on real-world systems. We observe that these works generally fall short of implementing the CT-RL algorithms directly. For example, Liu et al. [81] propose a model-free CT-RL algorithm for attitude control of multiple quadrotors, but the quadrotor model used neglects motor dynamics and assumes an algebraic relationship between the rotor speeds and body torques. Similarly, Cui et al. [82] implement a VI method for wheeled ground robots, but the model has to be reduced to that of a wheel inverted pendulum system and then linearized in order to accommodate the VI algorithm. Finally, Jiang and Jiang [83], [84] implement RADP [21] on power systems and sensorimotor control, respectively, but the system dynamics are assumed to be partially or fully linear so that the nonlinear RADP algorithm [21] simplifies to solving a sequence of linear equations.

### *D. Compelling CT Applications Demand CT Control Solutions*

There is no shortage of motivation to effectively tackle the CT nonlinear optimal control problem. From an applications' standpoint, there is a wealth of well-motivated real-world systems that are inherently CT in nature. In the central fields of robotics and autonomous vehicles, for instance, systems are naturally modeled by the mechanics of Euler–Lagrange, which are coupled second-order ordinary differential equations (ODEs). In wastewater treatment, influent and effluent flows are modeled as continuous fluid dynamical processes. Bacteria and substrate concentration models are based on the continuous decay regeneration theory. Many chemical processes, such as distillation columns, are also fundamentally continuous in nature and can be modeled by PDEs or nonlinear ODEs under appropriate assumptions.

### E. Heritage of CT Theoretical Results and Challenges in Data-Driven Approaches

From a theoretical standpoint, historically, control systems' literature has been developed with a focus on the CT setting [3], [85], [86], [87]. Indeed, it was the CT applications (specifically, the development of feedback amplifiers), which inspired Bode [88] and Nyquist [89] to devise the field of classical control, an analog discipline by inception. Subsequently, the optimal [3], adaptive [90], and robust [91] control disciplines were conceptualized in the CT setting, alongside the various CT works on feedback linearization [87], and input-output stability and the small gain theorem [92]. Indeed, many existing ADP approaches rely on classical adaptive control techniques [3], [90] when solving optimal control problems, as both aim at learning to control unknown systems using data measurements along system trajectories. Consequently, many of the performance guarantees of ADP were derived from the existing adaptive control results, such as [93], where the first DT-RL performance analysis was introduced. To do so, the neural network (NN) weights used in approximating the value function and the control law are treated as adaptation parameters. Lyapunov-based techniques applied to the weight tuning are used to guarantee convergence and closed-loop stability. However, these theoretical approaches may be inadequate. First, they have not been shown effectively solving realistic application problems. Second, it is not clear how to incorporate the diverse weight convergence or termination criteria within a Lyapunov construct. These criteria have become widely accepted common practice in training (deep) NNs, and they have shown profound successes in applications. Another fundamental issue of concern is how the Lyapunov-based CT-RL algorithms may scale up, as NNs rely on a vast number of weights to make approximations.

### F. Fundamental Questions to be Addressed

The illustrated gaps in knowledge between theory and application motivate the following fundamental questions: 1) why have CT-RL methodologies lagged so far behind their DT-RL counterparts, both in terms of theoretical/algorithm development and potential adoption by real-world applications? 2) why are substantial CT-RL applications works yet to be demonstrated in spite of longstanding theoretical results? and 3) what fundamental insights may be provided to designers to make the existing CT-RL methods more capable/implementable in practical applications? It is, therefore, pivotal to have a comprehensive analysis focusing on CT-RL algorithm insights and inherent limitations.

## III. Scope and Contributions

### A. Scope

In this work, we have chosen to focus on CT-RL in the optimal regulation context. Of course, CT-RL successfully addresses many other control problems. Recent works in event-based control [94], [95], [96] show promising theoretical results, but, at this point, these algorithms have not matured sufficiently to synthesize designs for meaningful real-world systems. An even more substantial body of work has been developed for multiagent dynamic game problems; see [97], [98], [99]. However, multiagent control is fundamentally different from single-agent control in terms of problem structure, theoretical results, and implementation in the field, so we believe that a separate, self-standing work devoted to these algorithms is required. Similarly, we have decided not to address CT-RL optimal tracking here although we realize that several important CT-RL works have been developed [78], [100], [101]. More fundamentally, we restrict the scope of this work to the regulation problem because we need a ground-up approach to diagnose the CT-RL theory/application knowledge gap. Moving on to event-based control, game problems, tracking, and so on, while eventually necessary, would, at this point, convolute the underlying issues and, thereby, detract from the analysis.

### B. Contributions

The contributions of this work are threefold.

1) We provide a comprehensive review of four foundational CT-RL control algorithms, deciphering key theoretical assumptions/results, and highlighting their significance to the following:
   a) the development of RL solutions to optimal control problems;
   b) impact on related literature.
2) We conduct in-depth, designer-focused quantitative analyses revealing gaps between CT-RL theoretical promises and practical synthesis.
3) We outline the needs of future innovative research in CT-RL, pointing out directions for potential new results based on realistic assumptions and system characteristics.

### C. Organization

We first formally define the CT-RL optimal control problem in Section IV. NNs, training, and theoretical results for each of the four methods are discussed in Sections V–VII, respectively. Sections VIII–X provide comprehensive and quantitative performance evaluations of the four methods. Finally, we conclude this study with a discussion and possible future directions of research in Section XI.

## IV. Problem Formulation

### A. Notation

Throughout this work, $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{Z}$, and $\mathbb{N}$ denote the sets of reals, nonnegative reals, integers, and naturals, respectively. For $n \in \mathbb{N}$, we denote $\mathbb{R}^n$ as the $n$-dimensional Euclidean space. $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^n$ or operator norm for matrices, unless decorated otherwise. We refer the reader to [86, p. 117] for the definition of positive (semi)definite functions and [86, p. 144] for class $\mathcal{K}$, $\mathcal{K}_\infty$, and $\mathcal{KL}$ functions. In this work, the set $\Omega \subset \mathbb{R}^n$ is assumed to be compact and contain the origin $x = 0$ in its interior.

### B. Problem Formulation: Optimal Control Problem

The background provided here follows largely from the works [17], [18]. Consider the nonlinear time-invariant affine system

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector, $u : \mathbb{R}_+ \to \mathbb{R}^m$ is a measurable locally essentially bounded control, $f : \mathbb{R}^n \to \mathbb{R}^n$, and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$. We make the following assumptions on the system (1).

*Assumption 1 (Dynamical Assumptions [18]):* $f$ and $g$ are Lipschitz on $\Omega$, and $f(0) = 0$.

We denote $x(t)$ as the solution at time $t \geq 0$ to the ODE (1) with initial condition $x(0) = x_0 \in \mathbb{R}^n$ evolving under the control $u(t) = \mu(x(t))$ given by the feedback control law $\mu : \mathbb{R}^n \to \mathbb{R}^m$. Define the infinite horizon performance index

$$J(x_0, \mu) = \int_0^\infty r(x(\tau), \mu(x(\tau)))d\tau \tag{2}$$

where $r : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_+$, $r(x, u) = Q(x) + u^T Ru$ is the running cost and is assumed to satisfy the following:

*Assumption 2 (Cost Structure Assumptions [18]):* $Q : \mathbb{R}^n \to \mathbb{R}_+$ is a positive definite, monotonically increasing function, and the system (1) is zero-state observable through $Q$. $R \in \mathbb{R}^{m\times m}$ satisfies $R = R^T > 0$.

*Definition 1 (Admissible Policies [18]):* A control policy $\mu : \mathbb{R}^n \to \mathbb{R}^m$ is admissible with respect to the cost (2) on $\Omega$, denoted $\mu \in \mathcal{A}(\Omega)$, if $\mu$ is continuous on $\Omega$, $\mu(0) = 0$, $\mu$ stabilizes the system (1) on $\Omega$ (cf. [17, Definition 3.1.2]), and $J(x_0, \mu)$ in (2) is finite for all $x_0 \in \Omega$.

The optimal regulation problem is to find the optimal control $\mu^* \in \mathcal{A}(\Omega)$ and its associated optimal cost function $V^*$ (if they exist) such that

$$\begin{aligned} V^*(x_0) &= \min_{\mu\in\mathcal{A}(\Omega)} J(x_0, \mu) \\ \mu^*(x_0) &= \arg\min_{\mu\in\mathcal{A}(\Omega)} J(x_0, \mu) \\ &= -\frac{1}{2}R^{-1}g^T(x_0)\nabla V^*(x_0) \quad \forall x_0 \in \Omega \end{aligned} \tag{3}$$

subject to the dynamics (1). Define the Hamiltonian function $H : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{1\times n} \to \mathbb{R}$ as

$$H(x, u, p) = p\big[f(x) + g(x)u\big] + r(x, u). \tag{4}$$

*Definition 2 (Generalized HJB (GHJB) Equation [18]):* For an admissible control $\mu \in \mathcal{A}(\Omega)$, the function $V \in \mathcal{C}^1(\Omega)$ satisfies the GHJB equation, written $\text{GHJB}(V, \mu) = 0$, if

$$H\big(x, \mu(x), (\nabla V(x))^{\mathrm{T}}\big) = 0 \quad \forall x \in \Omega, \ V(0) = 0. \tag{5}$$

In the affine case, (5) is given by

$$\nabla V^T\big[f + g\mu\big] + Q + \mu^T R\mu = 0, \quad V(0) = 0. \tag{6}$$

We next list the key properties of the GHJB equation.

*Lemma 1 (GHJB Equation Properties [18, Lemma 8]):* Suppose that the system (1) satisfies Assumption 1, and the cost structure (2) satisfies Assumption 2. Then, for each admissible policy $\mu \in \mathcal{A}(\Omega)$, there exists a unique $\mathcal{C}^1$ solution $V$ to the equation $\text{GHJB}(V, \mu) = 0$ (5). $V$ is a Lyapunov function on $\Omega$ for the closed-loop system comprised of (1), and $u = \mu(x)$ (in particular, $V$ is positive definite). Furthermore, $\text{GHJB}(V, \mu) = 0$ if and only if $V(x) = J(x, \mu)$, where $J$ is the performance index given in (2).

We next discuss the HJB equation and its fundamental importance to the optimal control problem.

*Definition 3 (HJB Equation [18]):* The function $V^* \in \mathcal{C}^1(\Omega)$ satisfies the HJB equation, written $\text{HJB}(V^*) = 0$, if

$$\text{HJB}(V^*) = \text{GHJB}\left(V^*, -\frac{1}{2}R^{-1}g^T\nabla V^*\right) = 0, \quad V^*(0) = 0. \tag{7}$$

In the affine case, (7) is given by

$$(\nabla V^*)^{\mathrm{T}} f - \frac{1}{4}(\nabla V^*)^{\mathrm{T}} g R^{-1} g^T \nabla V^* + Q = 0, \quad V^*(0) = 0. \tag{8}$$

The following theorem establishes sufficient conditions for the existence and uniqueness of solutions to the HJB equation (8). It also ties the solutions of the HJB equation to the optimal control problem.

*Theorem 1 (HJB Equation Properties [17]):* Suppose that Assumptions 1 and 2 hold. Then, there exists a unique positive definite $\mathcal{C}^1$ solution $V^*$ to the HJB equation (8), and $V^*$ is the optimal value function in (3), i.e., the associated control $\mu^*$ given by (3) is admissible and uniquely minimizes the performance index (2) over the admissible controls $\mathcal{A}(\Omega)$.

Through the classical PI Algorithm 1, the GHJB equation (a first-order linear PDE) may be solved successively to search for the solution of the HJB equation (a first-order nonlinear PDE). Solving the GHJB PDE, although simpler than the HJB PDE, is still a challenging problem. Subsequent sections outline numerically tractable methods by means of NN approximation and ADP.

**Algorithm 1** PI Algorithm

1: **Hyperparameters:** Initial admissible policy $\mu_0 \in \mathcal{A}(\Omega)$.
2: **for** $i = 0, 1, \ldots$ **do**
3: *Policy Evaluation:* Evaluate the performance index (2) for the policy $\mu_i$ by solving $\text{GHJB}(V_i, \mu_i) = 0$ (5).
4: *Policy Improvement:* Update the control by

$$\begin{aligned} \mu_{i+1}(x) &= \arg\min_{v\in\mathcal{A}(\Omega)} \left\{H\big(x, v(x), (\nabla V_i(x))^{\mathrm{T}}\big)\right\} \\ &= -\frac{1}{2}R^{-1}g^T(x)\nabla V_i(x). \end{aligned} \tag{9}$$

5: **end for**

*Theorem 2 (PI Algorithm Properties [17]):* Let all hypotheses be as in Theorem 1. Suppose that $\mu_0 \in \mathcal{A}(\Omega)$ is admissible, and consider the sequences $\{\mu_i\}_{i=0}^\infty$ and $\{V_i\}_{i=0}^\infty$ generated by the PI Algorithm 1. Then, the following holds:

1) $\mu_i \in \mathcal{A}(\Omega)$ for all $i \geq 0$.
2) $V_{i+1}(x) \leq V_i(x)$ for all $x \in \Omega$ and $i \geq 0$.
3) $V_i \to V^*$ and $\mu_i \to \mu^*$ uniformly on $\Omega$.
4) If $\Upsilon_i$ denotes the basin of attraction of policy $\mu_i$ ($i = 0, 1, \ldots$), and if $\Upsilon^*$ denotes the basin of attraction of $\mu^*$, then, for each $i \geq 0$, we have $\Omega \subset \Upsilon_i \subset \Upsilon_{i+1} \subset \Upsilon^*$.

## V. NN Structures Considered

In what follows, let $\{\phi_j\}_{j=1}^{\infty}$, $\{\psi_j\}_{j=1}^{\infty}$, $\{\theta_j\}_{j=1}^{\infty}$, and $\{\tilde{\psi}_j\}_{j=1}^{\infty}$ be sequences of linearly independent $\mathcal{C}^1$ basis functions that vanish at the origin, with $\phi_j, \psi_j, \theta_j : \mathbb{R}^n \to \mathbb{R}$ and $\tilde{\psi}_j : \mathbb{R}^n \to \mathbb{R}^{1\times m}$ for $j \in \mathbb{N}$.

### A. Critic Network

Consider the critic network

$$V^*(x) = \hat{V}(x, c) + \epsilon_1(x), \quad \hat{V}(x, c) = c^T \Phi(x) \tag{10}$$

where $N_1 \in \mathbb{N}$, $\Phi : \mathbb{R}^n \to \mathbb{R}^{N_1}$, $\Phi(x) = [\,\phi_1(x) \;\cdots\; \phi_{N_1}(x)\,]^{\mathrm{T}}$, $c \in \mathbb{R}^{N_1}$ is the critic weight vector, and $\epsilon_1 : \Omega \to \mathbb{R}$ is the critic NN approximation error function. For the sake of brevity, we shall whenever possible denote the critic (10) by $\hat{V}(x)$.

### B. Actor Network

Consider the actor network

$$\mu^*(x) = \hat{\mu}(x) + \epsilon_2(x) \tag{11}$$

where $\epsilon_2 : \Omega \to \mathbb{R}^m$ is the actor NN approximation error function. The considered methodologies (IRL [19], SPI [20], RADP [21], and CT-VI [22]) use three distinct actor networks $\hat{\mu}$ (11). The first structure, used by RADP [21], is given by

$$\hat{\mu}(x, W) = W^T \Psi(x) \tag{12}$$

where $N_2 \in \mathbb{N}$, $\Psi : \mathbb{R}^n \to \mathbb{R}^{N_2}$, $\Psi(x) = [\,\psi_1(x) \;\cdots\; \psi_{N_2}(x)\,]^{\mathrm{T}}$, and $W \in \mathbb{R}^{N_2\times m}$ is a weight matrix. The second structure, used by IRL [19] and SPI [20], is motivated by the structural form of $\mu^*$ assumed in (3)

$$\hat{\mu}(x, w) = -\frac{1}{2} R^{-1} g^T(x) \nabla \Phi^T(x) w \tag{13}$$

where $w \in \mathbb{R}^{N_2}$ is the actor weight vector. We note, for this structure, that $N_2 \leftarrow N_1$ is imposed, and knowledge of the input dynamics $g$ is required. This is in contrast to the structure (12), which is model-free. The third structure, used by CT-VI [22], is discussed next.

### C. Hamiltonian Network

Consider the Hamiltonian network

$$\begin{aligned} H^*(x, u) &\triangleq H\big(x, u, (\nabla V^*(x))^{\mathrm{T}}\big) \\ &= \hat{H}(x, u, \overline{v}) + \epsilon_3(x, u) \end{aligned} \tag{14}$$

where the Hamiltonian function $H$ is defined in (4), and $\epsilon_3 : \Omega \times \mathbb{R}^m \to \mathbb{R}$ is the Hamiltonian NN approximation error function. We consider the following approximation structure for the Hamiltonian NN:

$$\hat{H}(x, u, \overline{v}) = \overline{v}^T \Sigma(x, u) + \big(Q(x) + u^T R u\big) \tag{15}$$

where $N_3 \in \mathbb{N}$, $v \in \mathbb{R}^{N_3}$, $w \in \mathbb{R}^{N_2}$, $\overline{v} \triangleq [\,w^T \;\; v^T\,]^{\mathrm{T}}$, $\tilde{\Psi} : \Omega \to \mathbb{R}^{N_2\times m}$, $\tilde{\Psi}(x) = [\,\tilde{\psi}_1^T(x) \;\cdots\; \tilde{\psi}_{N_2}^T(x)\,]^{\mathrm{T}}$, and $\Sigma : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{N_2+N_3}$ is defined as

$$\Sigma(x, u) = \begin{bmatrix} \tilde{\Psi}(x) u \\ \Theta(x) \end{bmatrix}. \tag{16}$$

Here, $\Theta : \mathbb{R}^n \to \mathbb{R}^{N_3}$, $\Theta(x) = [\,\theta_1(x) \;\cdots\; \theta_{N_3}(x)\,]^{\mathrm{T}}$. We, thus, choose the following basis for the Hamiltonian NN:

$$\{\sigma_j(x, u)\}_{j=1}^{N_2+N_3+1} = \big\{\tilde{\psi}_j(x) u\big\}_{j=1}^{N_2} \cup \big\{\theta_j(x)\big\}_{j=1}^{N_3} \cup \{r(x, u)\}. \tag{17}$$

The selection of the basis functions (17), together with the definition of $H^*(x, u)$ (14), that $V^*(x)$ satisfies the HJB equation (8), and that, given $x \in \mathbb{R}^n$, the approximation (14) must hold for any $u \in \mathbb{R}^m$, implies that the following approximations are to be made:

$$w^T \tilde{\Psi}(x) \approx (\nabla V^*(x))^{\mathrm{T}} g(x) \in \mathbb{R}^{1\times m} \tag{18}$$

$$\begin{aligned} v^T \Theta(x) &\approx (\nabla V^*(x))^{\mathrm{T}} f(x) \\ &= \frac{1}{4} (\nabla V^*(x))^{\mathrm{T}} g(x) R^{-1} g^T(x) \nabla V^*(x) - Q(x). \end{aligned} \tag{19}$$

We are now ready to define the actor network adopted by CT-VI [22], which is given by

$$\hat{\mu}(x, w) = -\frac{1}{2} R^{-1} \big(w^T \tilde{\Psi}(x)\big)^{\mathrm{T}}. \tag{20}$$

*Remark 1 (CT-VI Basis Selection):* In [22, Sec. IV-B], the authors include the control penalty function $u^T R u$ in the Hamiltonian network basis $\{\sigma_j(x, u)\}_{j=1}^{N_2+N_3+1}$ (17) instead of the full running cost $r(x, u)$. This basis selection, in turn, makes the term $-Q(x)$ in (19) disappear, which, in principle, reduces the complexity of the required approximation. In spite of this intuition, we find the selection of basis (17) to be more numerically reliable in practice. Therefore, we employ (17) in our evaluations of Sections IX and X.

*Remark 2 (Notation):* In the single-input ($m = 1$) case, the RADP actor weight matrix $W \in \mathbb{R}^{N_2\times m}$ (12) becomes a weight vector, which we naturally denote $w \in \mathbb{R}^{N_2}$, as it is for the other three methods. Comparison of the RADP and CT-VI actor implementations (12) and (20), respectively, reveals these two structures to be identical modulo multiplication by the scalar term $-1/2R$. Thus, for CT-VI, we use the RADP activation functions $\{\psi_j\}_{j=1}^{N_2}$ in place of the activation functions $\{\tilde{\psi}_j\}_{j=1}^{N_2}$. We adopt these conventions in the evaluations of Sections IX and X, which focuses on single-input systems.

## VI. Algorithms and Training

We begin by defining some common notation. Each of these algorithms requires a CT-RL learning time $t_f > 0$ and associated learning window $t \in [0, t_f]$ over which to collect state-action data. For the two PI-based algorithms (IRL and RADP), we denote $i$ as the iteration index and $i^* \in \mathbb{N}$ as the final iteration. The PI-based algorithms require the collection of $l \in \mathbb{N}$ data samples per iteration. RADP reuses the same data for each iteration, so we denote its sample times as $\{t_k\}_{k=0}^{l}$ (i.e., $0 = t_0 < t_1 < \cdots < t_l = t_f$). IRL requires new data at each iteration $0 \le i \le i^*$, so we denote its sample instants as $\{\{t_k^i\}_{k=0}^{l}\}_{i=0}^{i^*}$ (i.e., $0 = t_0^0 < t_1^0 < \cdots < t_l^0 = t_0^1 < \cdots < t_l^{i^*} = t_f$). The algorithms have various termination criteria, so, to unify notation, we use the subscript "$f$" to denote an algorithm's final output value of the respective parameter (e.g., final critic weights $c_f$ and final critic network output $\hat{V}_f$).

### A. Integral Reinforcement Learning [19]

Given a state trajectory $\{x(t)\}_{t\in\mathbb{R}_+}$ of the system (1), define $\Delta\phi_j : \mathbb{R}_+^2 \to \mathbb{R}$ $(j = 1, \ldots, N_1)$ as $\Delta\phi_j(t_0, t_1) = \phi_j(x(t_1)) - \phi_j(x(t_0))$. Next, define $\Delta\Phi : \mathbb{R}_+^2 \to \mathbb{R}^{N_1}$ by

$$\Delta\Phi(t_0, t_1) = \begin{bmatrix} \Delta\phi_1(t_0, t_1) \\ \vdots \\ \Delta\phi_{N_1}(t_0, t_1) \end{bmatrix}. \tag{21}$$

For a strictly increasing sequence $\{t_k\}_{k=0}^{l}$, define $A_\phi : \mathbb{R}_+^{l+1} \to \mathbb{R}^{l\times N_1}$ by

$$A_\phi(t_0, \ldots, t_l) = \begin{bmatrix} \Delta\Phi^T(t_0, t_1) \\ \vdots \\ \Delta\Phi^T(t_{l-1}, t_l) \end{bmatrix}. \tag{22}$$

Next, for an admissible policy $\mu \in \mathcal{A}(\Omega)$, define the integral reinforcement function $\xi : \mathbb{R}_+^2 \times \mathcal{A}(\Omega) \to \mathbb{R}_+$ by

$$\xi(t_0, t_1, \mu) = \int_{t_0}^{t_1} \big(Q(x) + \mu^T(x)R\mu(x)\big)d\tau. \tag{23}$$

Similarly, define the function $\Xi : \mathbb{R}_+^{l+1} \times \mathcal{A}(\Omega) \to \mathbb{R}_+^{l}$ by

$$\Xi(t_0, \ldots, t_l, \mu) = \begin{bmatrix} \xi(t_0, t_1, \mu) \\ \vdots \\ \xi(t_{l-1}, t_l, \mu) \end{bmatrix}. \tag{24}$$

At iteration $i$ $(i = 0, 1, \ldots)$, IRL collects state trajectory data $\{x(t_k^i)\}_{k=0}^{l}$ at the time instants $\{t_k^i\}_{k=0}^{l}$ under the control $u = \hat{\mu}_i(x)$. At $t = t_l^i$, it updates its weights by solving the least-squares solution $c_i \in \mathbb{R}^{N_1}$ to the system of equations

$$\begin{aligned} \mathbf{A}_{\mathrm{IRL}}^i\, c_i &= -\Xi\big(t_0^i, \ldots, t_l^i, \hat{\mu}_i\big) \\ \mathbf{A}_{\mathrm{IRL}}^i &\triangleq A_\phi\big(t_0^i, \ldots, t_l^i\big) \in \mathbb{R}^{l\times N_1} \end{aligned} \tag{25}$$

where $A_\phi$ and $\Xi$ are defined in (22) and (24), respectively.

**Algorithm 2** IRL Algorithm [19]

1: **Hyperparameters:** $t_f$, $i^*$, $l$, sample times $\{\{t_k^i\}_{k=0}^{l}\}_{i=0}^{i^*}$, $\mu_0 \in \mathcal{A}(\Omega)$, and IC $x_0 \in \Omega$.
**Initialization:** Let $\hat{\mu}_0 \leftarrow \mu_0$.
2: **for** $i = 0 : i^*$ **do**
3: Apply control $u = \hat{\mu}_i(x)$ to system (1), collecting data $\{x(t_k^i)\}_{k=0}^{l}$ and $\{\xi(t_k^i, t_{k+1}^i, \hat{\mu}_i)\}_{k=0}^{l-1}$ (23).
4: Perform weight update $c_i$ (25) and policy update $\hat{\mu}_{i+1}(x) \leftarrow \hat{\mu}(x, c_i)$ (13).
5: **end for**
6: Apply final policy $\hat{\mu}_f = \hat{\mu}_{i^*+1}$.

### B. Synchronous Policy Iteration [20]

SPI updates its critic weights $\{c(t)\}_{t\in[0,t_f]}$ over the learning window $[0, t_f]$ dynamically via the tuning law

$$\dot{c} = -\alpha_1 \frac{\sigma_2}{m_s^2}\big[\sigma_2^T c + r(x, \hat{\mu}(x, w))\big] \tag{26}$$

where $\alpha_1 > 0$ is a tuning gain, $\hat{\mu}(x, w)$ is given by the actor network (13), $\sigma_2(x) = \nabla\Phi(x)[f(x)+g(x)\hat{\mu}(x, w)]$, and $m_s(x) = (\sigma_2^T(x)\sigma_2(x) + 1)$ is a normalization term.

*Remark 3 (SPI Actor Tuning):* Vamvoudakis and Lewis [20] prescribe the following actor tuning law:

$$\dot{w} = -\alpha_2\left[\left(F_2 w - F_1 \frac{\sigma_2^T}{m_s} c\right) - \frac{1}{4}D(x)w\frac{\sigma_2^T}{m_s^2}c\right] \tag{27}$$

where $\alpha_2 > 0$, $F_1 \in \mathbb{R}^{N_1} > 0$, $F_2 \in \mathbb{R}^{N_1\times N_1}$, $F_2 = F_2^T > 0$ are tuning parameters, and $D(x) = \nabla\Phi(x)g(x)R^{-1}g^T(x)\nabla\Phi^T(x)$. After extensive exploration, we were unable to find parameter values $\alpha_2$, $F_1$, and $F_2$, which yield stable state trajectory and weight responses for the examples studied in this work. In the code (found at [102]) for a previous rendition [103] of the SPI algorithm [20], the authors implement the following modified tuning law, which we observe to function properly and hence use throughout this work instead of (27):

$$\dot{w} = -\alpha_2\left[(F_2 w - F_2 c) - \frac{1}{4}D(x)w\frac{\sigma_2^T}{m_s^2}c\right]. \tag{28}$$

We note that, after adopting the modified tuning law (28), the closed-loop stability and convergence results for SPI [20] (cf. Theorems 5 and 6, respectively) are no longer guaranteed. Examining the update (28) qualitatively, we note that the rightmost terms in (28) vanish as $\|x\| \to 0$ and $\|x\| \to \infty$. Thus, in these regimes, (28) can be approximated by $\dot{w} \approx -\alpha_2 F_2(w - c)$, which resembles a linear tracking control law, whereby the actor weights $w(t)$ track the critic weights $c(t)$.

**Algorithm 3** SPI Algorithm [20]

1: **Hyperparameters:** $t_f$, tuning gains $\alpha_1, \alpha_2 > 0$, $F_1 > 0$, $F_2 = F_2^T > 0$ (26), $e$, $\mu_0 \in \mathcal{A}(\Omega)$ (cf. Assumption 3), IC $x_0 \in \Omega$, $c_0 \in \mathbb{R}^{N_1}$ (cf. Remark 9), and $w_0 \in \mathbb{R}^{N_1}$ such that $\hat{\mu}(x, w_0) \in \mathcal{A}(\Omega)$ (13).
**Initialization:** Let $c(0) \leftarrow c_0$, $w(0) \leftarrow w_0$.
2: **for** $t \in [0, t_f]$ **do**
3: Apply control $u(t) = \hat{\mu}(x(t), w(t)) + e(t)$ (13) to system (1), tuning critic weights $c(t)$ via (26) and actor weights $w(t)$ via (27) (or (28), cf. Remark 3).
4: **end for**
5: Terminate $e$. Apply final policy $\hat{\mu}_f(x) = \hat{\mu}(x, w(t_f))$ (13).

### C. Robust Adaptive Dynamic Programming [21]

In what follows, suppose that the system (1) evolves under the control $u = \mu_0(x) + e$ (cf. Assumption 4), generating the trajectory $\{x(t)\}_{t\in\mathbb{R}_+}$. For an admissible policy $\mu \in \mathcal{A}(\Omega)$, define the function $\Delta_\psi : \mathbb{R}_+^2 \times \mathcal{A}(\Omega) \to \mathbb{R}^{mN_2}$ by

$$\Delta_\psi(t_0, t_1, \mu) = \int_{t_0}^{t_1} [R(u - \mu(x))] \otimes \Psi(x)\, d\tau \tag{29}$$

where $\otimes$ denotes the Kronecker tensor product. Given a strictly increasing sequence $\{t_k\}_{k=0}^{l}$ and a policy $\mu \in \mathcal{A}(\Omega)$, define $A_\psi : \mathbb{R}_+^{l+1} \times \mathcal{A}(\Omega) \to \mathbb{R}^{l\times mN_2}$ by

$$A_\psi(t_0, \ldots, t_l, \mu) = 2\begin{bmatrix} \Delta_\psi^T(t_0, t_1, \mu) \\ \vdots \\ \Delta_\psi^T(t_{l-1}, t_l, \mu) \end{bmatrix}. \tag{30}$$

At iteration $i$ ($i = 0, 1, \ldots$) of the RADP algorithm, the weights $c_i \in \mathbb{R}^{N_1}$ and $W_i \in \mathbb{R}^{N_2 \times m}$ are solved for as the least-squares solution to the system of equations

$$\mathbf{A}^i_{\mathrm{RADP}} \begin{bmatrix} c_i \\ \mathrm{vec}(W_i) \end{bmatrix} = -\Xi\big(t_0, \ldots, t_l, \hat{\mu}_i\big)$$
$$\mathbf{A}^i_{\mathrm{RADP}} \triangleq \begin{bmatrix} A_\phi(t_0, \ldots, t_l) & A_\psi\big(t_0, \ldots, t_l, \hat{\mu}_i\big) \end{bmatrix} \in \mathbb{R}^{l \times (N_1 + mN_2)} \tag{31}$$

where $\mathrm{vec}(W) \in \mathbb{R}^{N_2 m}$ denotes the vectorization of the matrix $W \in \mathbb{R}^{N_2 \times m}$, and the functions $A_\phi$, $\Xi$, and $A_\psi$ are as defined in (22), (24), and (30), respectively.

**Algorithm 4** RADP Algorithm [21]

1: **Hyperparameters:** $t_f$, $i^*$, $l$, sample times $\{t_k\}_{k=0}^{l}$, $e$, $\mu_0 \in \mathcal{A}(\Omega)$ (cf. Assumption 4), IC $x_0 \in \Omega$, and $W_0 = 0$.
**Initialization:** Let $\hat{\mu}_0(x) \leftarrow \hat{\mu}(x, W_0)$ (12).
2: Apply control $u = \mu_0(x) + e$ to system (1), collecting state-action data $\{(x(t), u(t))\}_{t \in [0, t_f]}$.
3: **for** $i = 0 : i^*$ **do**
4: Calculate for policy $\hat{\mu}_i$ the data $\{\xi(t_k, t_{k+1}, \hat{\mu}_i)\}_{k=0}^{l-1}$ (23) and $\{\Delta_\psi(t_k, t_{k+1}, \hat{\mu}_i)\}_{k=0}^{l-1}$ (29).
5: Perform weight update $c_i$, $W_i$ (31) and policy update $\hat{\mu}_{i+1}(x) \leftarrow \hat{\mu}(x, W_i)$ (12).
6: **end for**
7: Terminate $e$. Apply final policy $\hat{\mu}_f = \hat{\mu}_{i^*+1}$.

*Remark 4 (RADP Robustness Results):* The RADP algorithm, as presented in [21, Algorithm 1], adds a robustifying term to the final policy $\hat{\mu}_f = \hat{\mu}_{i^*+1}$ produced by Algorithm 4 for its stability robustness results (cf. [21, Sec. III-B]). Yet, our initial attempts to implement this robustness term were thwarted by closed-loop stability issues that we observe from Algorithm 4 in practice (cf. Sections IX and X). As noted by Jiang and Jiang [21, Remark 3.2], in the absence of dynamic uncertainties, the RADP algorithm may be run entirely without the developed robustifying term, which is the procedure followed in this article.

### D. Continuous-Time Value Iteration [22]

For CT-VI, a measurable essentially bounded input $u$ (cf. Assumption 6) is applied to the system (1) over the window $[0, t_f]$, generating the trajectory $\{x(t)\}_{t \in [0, t_f]}$. After data have been collected, CT-VI then tunes its weights dynamically over a learning time scale $s \in [0, s_f]$, which is independent of the system time scale $t \in [0, t_f]$. CT-VI updates its critic weights $\{c(s)\}_{s \in [0, s_f]}$ via the tuning law

$$\frac{d}{ds} c(s) = K_\phi^{-1}\big(t_f\big) \int_0^{t_f} \Phi(x) \hat{H}\big(x, \hat{\mu}(x, w(s)), \overline{v}(s)\big) d\tau \tag{32}$$

where

$$K_\phi\big(t_f\big) = \int_0^{t_f} \Phi(x) \Phi^T(x) d\tau \in \mathbb{R}^{N_1 \times N_1}. \tag{33}$$

CT-VI updates its actor weights $\{w(s)\}_{s \in [0, s_f]}$ and Hamiltonian weights $\{v(s)\}_{s \in [0, s_f]}$ via the tuning law

$$\overline{v}(s) = K_\sigma^{-1}\big(t_f\big) \int_0^{t_f} \Sigma(x, u) \left( \frac{d}{d\tau} \hat{V}(x, c(s)) + r(x, u) \right) d\tau \tag{34}$$

TABLE I
RELEVANT TERMS AND DEFINITIONS

| Term | Reference |
|---|---|
| Persistence of Excitation (PE) | [90, Def. 4.3.1, pp. 177] |
| Lyapunov Stability, Asymptotic Stability (AS) | [86, Def. 4.1, pp. 112] |
| Basin of Attraction, Global AS (GAS) | [86, pp. 122] |
| Local AS (LAS), Regional AS (RAS), Semiglobal Stabilization | [86, pp. 473] |
| Practical Stabilization (PS) | [104, Def. 1.2.1, pp. 9] |
| Uniformly Ultimately Bounded (UUB) | [86, Def. 4.6, pp. 169] |
| Input-to-State Stability (ISS) | [86, Def. 4.7, pp. 175] |

where $\overline{v} = [\, w^T \quad v^T \,]^{\mathrm{T}}$ (cf. Section V), and

$$K_\sigma\big(t_f\big) = \int_0^{t_f} \Sigma(x) \Sigma^T(x) d\tau \in \mathbb{R}^{(N_2+N_3) \times (N_2+N_3)}. \tag{35}$$

**Algorithm 5** CT-VI Algorithm [22]

1: **Hyperparameters:** $t_f$, $s_f$, control $u$ (cf. Assumption 6), IC $x_0 \in \Omega$, $c_0 \in \mathbb{R}^{N_1}$ such that $\hat{V}(x, c_0)$ (10) is positive definite and radially unbounded, $w_0 = 0$, $v_0 = 0$.
**Initialization:** Let $c(0) \leftarrow c_0$, $w(0) \leftarrow w_0$, $v(0) \leftarrow v_0$.
2: Apply control $u$ to system (1), collecting state-action data $\{(x(t), u(t))\}_{t \in [0, t_f]}$.
3: **for** $s \in [0, s_f]$ **do**
4: Tune critic weights $c(s)$ via (32) and the actor, Hamiltonian weights $w(s)$, $v(s)$ via (34).
5: **end for**
6: Apply final policy $\hat{\mu}_f(x) = \hat{\mu}(x, w(s_f))$ (20).

## VII. Theoretical Results

This section discusses the key assumptions and properties of the four algorithms studied in this work. Throughout this section, we assume that the baseline hypotheses of Section IV hold, which ensures that the optimal control problem is well-posed. Table I lists the terms needed to understand subsequent analysis and provides specific references to their definitions.

### A. IRL

For IRL, the main convergence and stability results rely upon the following technical lemma, which is a restatement of [19, Lemma 3].

*Lemma 2:* Suppose, for admissible $\mu \in \mathcal{A}(\Omega)$, that the system (1) is simulated under the control $u = \mu(x)$, generating the state trajectory $\{x(t)\}_{t \in \mathbb{R}_+}$. Given that the set $\{\phi_j\}_{j=1}^{N_1}$ is linearly independent, for each $t_0 \geq 0$ and $l \geq N_1$, there exists a strictly increasing sequence $\{t_k\}_{k=1}^{l}$ such that the matrix $A_\phi(t_0, \ldots, t_l) \in \mathbb{R}^{l \times N_1}$ (22) has full column rank $N_1$.

We next move on to the key stability/convergence results.

*Theorem 3 (IRL—Admissibility of Policies $\hat{\mu}_i$):* Suppose that $\mu_0 \in \mathcal{A}(\Omega)$ is admissible. There exists $N_{1,0} \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$, the policies $\{\hat{\mu}_i\}_{i=1}^{\infty}$ generated by Algorithm 2 are each admissible.

*Theorem 4 (IRL—Uniform Approximation):* For each $\epsilon > 0$, there exist $N_{1,0}, i_0 \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$ and $i^* \geq i_0$, we have

$$\left\| \hat{V}_f - V^* \right\|_\infty < \epsilon, \quad \left\| \hat{\mu}_f - \mu^* \right\|_\infty < \epsilon \tag{36}$$

where $\hat{V}_f = \hat{V}_{i^*} = \hat{V}(x, c_{i^*})$ (10) and $\hat{\mu}_f = \hat{\mu}_{i^*+1} = \hat{\mu}(x, c_{i^*})$ (13) are as generated by Algorithm 2. Here, $\|\cdot\|_\infty$ denotes the uniform norm on $\mathcal{C}(\Omega)$.

### B. SPI

As with many ADP algorithms, SPI [20] has a persistence of excitation (PE) requirement.

*Assumption 3 (SPI—PE Assumption):* The signals $\overline{\sigma}_1 = \sigma_1/(\sigma_1^T\sigma_1 + 1)$ and $\sigma_1 = \nabla\Phi(x)[f(x) + g(x)\mu^*(x)]$ are PE.

We also require the use of the following lemma.

*Lemma 3 [20, Lemma 1]:* The solution $\hat{c}^*$ to the least-squares minimization (37) exists and is unique, where

$$\begin{aligned}\hat{c}^* &= \min_{c\in\mathbb{R}^{N_1}} \left\| H\left(x, \mu^*(x), \nabla\hat{V}^T(x, c)\right)\right\|_{L^2(\Omega)} \\ &= \min_{c\in\mathbb{R}^{N_1}} \left\| c^T\nabla\Phi\left[f + g\mu^*\right] + r(x, \mu^*)\right\|_{L^2(\Omega)}. \end{aligned} \tag{37}$$

Before presenting the key stability and approximation results for SPI [20], we make the note that they require the application of original actor tuning law (27), which we had to modify to (28) (cf. Remark 3 for discussion).

*Theorem 5 (SPI—UUB Stability):* Let tuning for the critic network (10) be provided by (26) and tuning for the actor network (13) be provided by (27). Suppose that tuning parameters are selected according to [20, Appendix]. Finally, consider the system (1) simulated under the control $u(t) = \hat{\mu}(x(t), w(t)) + e(t)$ (13), and assume that the PE Assumption 3 is satisfied. Then, there exists $N_{1,0} \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$, the closed-loop system state $x(t)$, the critic error $\tilde{c} = \hat{c}^* - c$ (37), and the actor error $\tilde{w} = \hat{c}^* - w$ are UUB.

*Theorem 6 (SPI—Uniform Approximation):* Let all hypotheses be as in Theorem 5. Then, for each $\epsilon > 0$, there exists $N_{1,0} \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$, there exists $t_{f,0} = t_{f,0}(N_1)$ such that $t_f \geq t_{f,0}$ implies that the uniform approximation result (36) holds for $\hat{V}_f(x) = \hat{V}(x, c(t_f))$ (10) and $\hat{\mu}_f(x) = \hat{\mu}(x, w(t_f))$ (13).

### C. RADP

For RADP [21], we require that the initial policy $\mu_0 \in \mathcal{A}(\Omega)$ is admissible and satisfies the following assumption.

*Assumption 4:* The policy $\mu_0 \in \mathcal{A}(\Omega)$ is admissible and is such that, for the exploration noise $e$, there exists a compact set $\Omega_0 \subset \Omega$ containing the origin in its interior for which, given any initial condition $x_0 \in \Omega_0$, $\Omega$ is an invariant set for the trajectory $x(t)$ generated by the closed-loop system composed of (1) and $u = \mu_0(x) + e$.

*Assumption 5 (RADP—PE-Like Assumption):* There exist $l_0 \in \mathbb{N}$ and $\delta > 0$ such that, for all $l \geq l_0$, we have

$$\delta I_{N_1+mN_2} \leq \frac{1}{l}\sum_{k=0}^{l-1} \zeta_{i,k}\zeta_{i,k}^T \quad \forall\, i \geq 0 \tag{38}$$

where for $k = 0, \ldots, l-1$

$$\zeta_{i,k} = \begin{bmatrix} \Delta\Phi(t_k, t_{k+1}) \\ 2\Delta_\psi(t_k, t_{k+1}, \hat{\mu}_i) \end{bmatrix} \in \mathbb{R}^{N_1+mN_2} \tag{39}$$

and the functions $\Delta\Phi$ and $\Delta_\psi$ are defined in (22) and (30), respectively.

We are now ready to present the key stability and approximation results for RADP [21].

*Theorem 7 (RADP—Admissibility of Policies $\hat{\mu}_i$):* Suppose that Assumptions 4 and 5 hold. There exist $N_{1,0}$, $N_{2,0} \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$ and $N_2 \geq N_{2,0}$, the policies $\{\hat{\mu}_i\}_{i=1}^\infty$ generated by Algorithm 4 are each admissible.

*Theorem 8 (RADP—Uniform Approximation):* Suppose that Assumptions 4 and 5 hold. For each $\epsilon > 0$, there exist $N_{1,0}$, $N_{2,0}$, $i_0 \in \mathbb{N}$ such that, whenever $N_1 \geq N_{1,0}$, $N_2 \geq N_{2,0}$, and $i^* \geq i_0$, the uniform approximation result (36) holds for $\hat{V}_f = \hat{V}_{i^*} = \hat{V}(x, c_{i^*})$ (10) and $\hat{\mu}_f = \hat{\mu}_{i^*+1} = \hat{\mu}(x, W_{i^*})$ (12), as generated by Algorithm 4.

### D. CT-VI

CT-VI [22] has a PE-like assumption that we outline here.

*Assumption 6 (CT-VI—PE-Like Assumption):* The measurable essentially bounded input $u$ is such that there exist $\delta > 0$ and $t_0 > 0$ such that, for all $t_f \geq t_0$, the trajectory $\{x(t)\}_{t\in[0,t_f]}$ under $u$ remains in $\Omega$, and

$$\delta I_{N_1} < \frac{1}{t_f}K_\phi(t_f), \quad \delta I_{N_2+N_3} < \frac{1}{t_f}K_\sigma(t_f) \tag{40}$$

where the matrices $K_\phi(t_f)$ and $K_\sigma(t_f)$ are defined in (33) and (35), respectively.

We are now ready to move on to the key results.

*Theorem 9 [CT-VI—Regional Practical Stabilization (RPS)]:* Suppose that Assumption 6 holds. For each $\epsilon > 0$ such that $B_\epsilon(0) \subset \Omega$, there exist $t_f$, $s_f > 0$, $N_1$, $N_2$, $N_3 \in \mathbb{N}$ such that

$$\nabla V^*(x)^{\mathrm{T}}\left[f(x) + g(x)\hat{\mu}(x, c(s_f))\right] < 0 \quad \forall x \in \Omega\backslash B_\epsilon(0) \tag{41}$$

the weights $c(s)$, $w(s)$, and $v(s)$ being tuned by Algorithm 5.

*Remark 5:* The results of Theorem 9 provided in the original CT-VI work (cf. [22, Theroem 3]) actually guarantee semiglobal PS; i.e., the compact set $\Omega \subset \mathbb{R}^n$ in Theorem 9 may be made arbitrarily large. However, the definition of admissibility in [22] requires policies to be GAS. In our context, admissible policies $\mu \in \mathcal{A}(\Omega)$ only guarantee RAS on a fixed compact set $\Omega \subset \mathbb{R}^n$, so the associated stability results for CT-VI are only regional when applied here. This subtlety is addressed by Bian and Jiang [22, Remark 6].

*Theorem 10 (CT-VI—Uniform Approximation):* Suppose that Assumption 6 holds, and $Q$ is continuous. For each $\epsilon > 0$, there exist $N_{1,0}$, $N_{2,0}$, $N_{3,0} \in \mathbb{N}$, $t_f, s_f > 0$, and compact $\Omega_u \subset \mathbb{R}^m$ containing $u = 0$ in its interior sufficiently large such that, whenever $N_1 \geq N_{1,0}$, $N_2 \geq N_{2,0}$, and $N_3 \geq N_{3,0}$, we have that the uniform approximation result (36) holds for $\hat{V}_f(x) = \hat{V}(x, c(s_f))$ (10) and $\hat{\mu}_f = \hat{\mu}(x, w(s_f))$ (20), as generated by Algorithm 5, and

$$\left\|\hat{H}_f - H^*\right\|_\infty < \epsilon \tag{42}$$

where $\hat{H}_f(x, u) = \hat{H}(x, u, \overline{v}(s_f))$ (15), as generated by Algorithm 5, and $\|\cdot\|_\infty$ in (42) is the uniform norm on $\mathcal{C}(\Omega \times \Omega_u)$.

### E. Summary and Discussion of Methodologies

Table II provides an overview of the essential features of the four methodologies considered.

*Remark 6 (Initial Admissible Policy):* IRL, SPI, and RADP require an initial admissible policy $\mu_0 \in \mathcal{A}(\Omega)$. In contrast to

TABLE II
SUMMARY OF CT–RL METHODOLOGIES

| Alg. | Dyn. Req'd | PE? | Data Reuse? | Conv. Results | Stability Results |
|---|---|---|---|---|---|
| IRL | $g$ | No[a] | No | Uniform | RAS |
| SPI | $f, g$ | Yes | No | Uniform | UUB[b] |
| RADP | None | Yes | Yes | Uniform | RAS |
| CT-VI | None | Yes | Yes | Uniform | RPS |

[a]Cf. Remark 7.
[b]Cf. Remark 6.

IRL and RADP, which (modulo Assumption 4 for RADP) are structurally unconstrained in their selection of $\mu_0 \in \mathcal{A}(\Omega)$, SPI requires that the initial policy $\mu_0$ is implementable in the actor network (13) as $\mu_0(x) = \hat{\mu}(x, w_0)$ for some $w_0 \in \mathbb{R}^{N_1}$. This is comparatively quite restrictive and depends on the input dynamics $g$ and critic basis functions $\{\phi_j\}_{j=1}^{N_1}$ available.

Meanwhile, as is the case in the DT setting, strictly speaking, CT-VI does not require an initial stabilizing policy [22]. Bian and Jiang [22] suggest reinitializing the trajectory $x(t)$ whenever it leaves $\Omega$ in the learning interval $[0, t_f]$ (cf. [22, Remark 3]). However, this is not a luxury afforded in a real-world online learning scenario, so, realistically speaking, a designer will likely require an initial policy $\mu_0 \in \mathcal{A}(\Omega)$ to run CT-VI.

*Remark 7 (Pseudoinversion, Conditioning):* Three of the four algorithms studied here (IRL, RADP, and CT-VI) require the use of the Moore–Penrose pseudoinverse, for which condition number plays a fundamental role in solution accuracy and sensitivity [105]. In the case of IRL and RADP, at iteration $i$, the matrices $\mathbf{A}^i_{\text{IRL}}$ (25) and $\mathbf{A}^i_{\text{RADP}}$ (31) are pseudoinverted to yield the least-squares solutions for their respective weight updates. For CT-VI, in this work, we implement the matrix inverses $K_\phi^{-1}(t_f)$ (32) and $K_\sigma^{-1}(t_f)$ (34) via pseudoinversion for improved computation speed and accuracy.

*Remark 8 (PE Assumptions):* SPI relies on the PE Assumption 3, while RADP and CT-VI rely on the PE-like Assumptions 5 and 6, respectively. As has long been understood, for nonlinear systems, there do not exist systematic frameworks for verifying PE or for selecting a probing noise to ensure PE [20]. As we will illustrate in Sections IX and X, in practice, it is a challenge to excite the system to yield quality state trajectory data, even for simple academic examples.

IRL, strictly speaking, does not have a PE requirement. However, the performance of this algorithm is still deeply tied to the quality of state trajectory data, as full column rank of $\mathbf{A}^i_{\text{IRL}} \mathbb{R}^{l \times N_1}$ is required at each iteration $i$ for the least-squares weight update (25). Lemma 2 furnishes the existence of sample instances to meet the rank condition, but, as will be seen in Sections IX and X, lack of ability to insert a probing noise $e$ makes it difficult to systematically ensure good conditioning.

## VIII. PERFORMANCE EVALUATION SETUP

In this section, we offer four sets of fundamental evaluations of the studied CT-RL algorithms. Throughout, we keep our focus from the perspective of a designer, working from a ground-up assessment that illustrates performance effectiveness, efficiency, limitations, and insights. The first three evaluations examine a second-order academic system (43) to establish performance baselines and insights. The fourth evaluation examines a cart inverted pendulum system (58) to assess the potential of real-world implementability.

These studies were performed in MATLAB R2021a, on an NVIDIA RTX 2060, and Intel i7 (9th Gen) processor. All numerical integrations in this work are performed in MATLAB's adaptive `ode45` solver to ensure solution accuracy. The complete MATLAB software suite used to produce the data in this article is available at [106].

### A. Setup—Second-Order System

The first three evaluations consider the following second-order academic system from [19, Sec. 6.2]:

$$f(x) = \begin{bmatrix} -x_1 + x_2 + 2x_2^3 \\ -\frac{1}{2}(x_1 + x_2) + \frac{1}{2}x_2\left(1 + 2x_2^2\right)\sin^2(x_1) \end{bmatrix}$$
$$g(x) = \begin{bmatrix} 0 \\ \sin(x_1) \end{bmatrix}. \tag{43}$$

Linearization reveals that the origin of this system is an unstable equilibrium point. We run each algorithm over the IC sweep $x(0) = [-1{:}0.25{:}1]^2 \backslash \{(0,0)\}$. In the first three evaluations, a "trial" corresponds to each of the ICs. We define the running cost as $Q(x) = x_1^2 + x_2^2 + 2x_2^4$, and $R = 1$. This example was constructed such that the optimal value $V^*$ and policy $\mu^*$ are available in the closed form and are given by

$$V^*(x) = \frac{1}{2}x_1^2 + x_2^2 + x_2^4 \tag{44}$$
$$\mu^*(x) = -\sin(x_1)\left(x_2 + 2x_2^3\right). \tag{45}$$

*1) Basis Functions:* Examining the optimal value $V^*$ (44) and policy $\mu^*$ (45), for the first evaluation, we select the following minimum-dimension critic basis (10)

$$\Phi(x) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_2^4 \end{bmatrix}, \quad \nabla\Phi(x) = \begin{bmatrix} 2x_1 & 0 \\ 0 & 2x_2 \\ 0 & 4x_2^3 \end{bmatrix}. \tag{46}$$

By inspection of (44) and (46), the optimal critic weights are $c^* = [(1/2), 1, 1]^{\mathrm{T}}$. For the second evaluation, we increment the problem dimension by adding the nonessential basis function $\phi_4(x) = x_1 x_2$, i.e.,

$$\Phi(x) = \begin{bmatrix} x_1^2 & x_2^2 & x_2^4 & x_1 x_2 \end{bmatrix}^{\mathrm{T}} \tag{47}$$

which, by inspection, has optimal weights $c^* = [(1/2), 1, 1, 0]^{\mathrm{T}}$. In the third evaluation, for the sake of comparison, we choose the critic basis as identical to that of the example it was originally studied in [19, Sec. 6.2], with the essential basis function $x_2^4$ removed, i.e.,

$$\Phi(x) = \begin{bmatrix} x_1^2 & x_2^2 & x_1 x_2 & x_1^4 & x_1^3 x_2 & x_1^2 x_2^2 & x_1 x_2^3 \end{bmatrix}^{\mathrm{T}}. \tag{48}$$

With the removal of this term, the optimal value function $V^*$ (44) and policy $\mu^*$ (45) can no longer be approximated exactly.

This is a more realistic scenario than the previous set of exact basis functions. Since it is well-known that, in general, there is no closed-form solution to the HJB equation, a designer would naturally bias their selection toward such lower order terms.

Recall that, since the system (43) is single-input (i.e., $m = 1$), the same basis functions may be used for the RADP actor basis (12) and the CT-VI actor basis (20) (cf. Remark 2). Throughout the first three evaluations, we use the minimum-dimension basis

$$\Psi(x) = \sin(x_1)\begin{bmatrix} x_2 \\ x_2^3 \end{bmatrix}. \tag{49}$$

The optimal actor weights $w^*$ are given by $w^* = [-1, -2]^{\mathrm{T}}$ in the network (12), $w^* = c^*$ in the network (13), and $w^* = [2, 4]^{\mathrm{T}}$ in the network (20). After working out the algebra, it can be checked that

$$\frac{1}{4}(\nabla V^*)^{\mathrm{T}} g R^{-1} g^T \nabla V^* = \sin^2(x_1)\left[x_2^2 + 4x_2^4 + 4x_2^6\right]. \tag{50}$$

Examination of (50) and (19) motivates the minimal choice of Hamiltonian basis

$$\Theta(x) = \begin{bmatrix} \sin^2(x_1)x_2^2 \\ \sin^2(x_1)x_2^4 \\ \sin^2(x_1)x_2^6 \\ x_1^2 \\ x_2^2 \\ x_2^4 \end{bmatrix}, \quad v^* = \begin{bmatrix} 1 \\ 4 \\ 4 \\ -1 \\ -1 \\ -2 \end{bmatrix}. \tag{51}$$

The first three terms in $\Theta(x)$ compose (50), while the last three terms compose $-Q(x)$ in (19) (cf. Remark 1 for discussion).

*2) Initial Stabilizing Policy $\mu_0$:* Vrabie and Lewis [19] use the initial stabilizing policy

$$\mu_0(x) = -\frac{1}{2}\sin(x_1)\left(3x_2 - 0.2x_1^2 x_2 + 12x_2^3\right). \tag{52}$$

However, examining the minimal critic basis $\Phi(x)$ and its Jacobian $\nabla\Phi(x)$ (46), we see that we do not have access to the $x_1^2 x_2$ cross term for implementation of (52) in the actor network (13). Thus, in the spirit of continuity and maintaining a consistent comparison across the methodologies, we choose the similar stabilizing policy

$$\mu_0(x) = -\frac{1}{2}\sin(x_1)\left(3x_2 + 12x_2^3\right) \tag{53}$$

for the first two evaluations. For similar reasons, the critic basis (48) in the third evaluation necessitates that modify the policy (53). We choose for the third evaluation

$$\mu_0(x) = -5\sin(x_1)x_2. \tag{54}$$

*3) Exploration Noise $e$:* We consider the following three default low-, medium-, and high-excitation noises:

$$e_1(t) = 5\cos(t), \quad e_2(t) = 10\cos(t), \quad e_3(t) = 20\cos(t). \tag{55}$$

In the first evaluation, we further perturb $e_3$ as

$$e_3(t) = 20\cos(t) + \cos(0.1t) \tag{56}$$

TABLE III
EXPLORATION NOISES FOR THE FIRST THREE EVALUATIONS

| Eval. | $e_1(t)$ | $e_2(t)$ | $e_3(t)$ |
|---|---|---|---|
| 1 | (55) | (55) | (56) |
| 2 | (57) | (55) | (55) |
| 3[a] | (55) | N/A | N/A |

[a]Noises customized for each method to maximize chances of convergence. See Section IX-C for details.

TABLE IV
HYPERPARAMETERS FOR THE FIRST THREE EVALUATIONS

| Eval. | Alg. | $t_f$ (s) | $s_f$ (s) | $i^*$ | $l$ |
|---|---|---|---|---|---|
| 1, 2 | IRL | 5 | N/A | 5 | 10 |
| | SPI | 500 | N/A | N/A | N/A |
| | RADP | 10 | N/A | 5 | 50 |
| | CT-VI | 10 | 125 | N/A | N/A |
| 3 | IRL | 5 | N/A | 15 | 10 |
| | SPI | 500 | N/A | N/A | N/A |
| | RADP | 10 | N/A | 15 | 50 |
| | CT-VI | 10 | 125 | N/A | N/A |

because CT-VI exhibits convergence issues without the addition of the small low-frequency term. Similarly, in the second evaluation, we further perturb $e_1$ as

$$e_1(t) = 5\cos(t) + 0.25\cos(0.1t) \tag{57}$$

because CT-VI fails to converge for a few of the initial conditions in the sweep with the default exploration noise $e_1$ (55). Furthermore, by our search, no small-amplitude perturbation of the exploration noise $e_3$ (55) is able to make CT-VI converge for all initial conditions in the second evaluation. As a result, in the second evaluation, we choose the default exploration noise $e_3$ (55), and CT-VI is not run for this noise. These selections are summarized in Table III.

*4) Hyperparameter and Weight Initialization:* Hyperparameter selections are listed in Table IV. We use a default learning time $t_f = 10$ s. IRL's lack of probing noise necessitates a shorter learning time (cf. Section IX-A), whereas SPI's dynamic tuning laws require a longer learning time. For IRL, we collect $l = 10$ samples per iteration, while, for RADP, we collect $l = 50$ samples (all at equally spaced time instants). We collect more points for RADP because its associated least-squares minimization is higher dimensional [$N_1 + N_2$; cf. (31)] than that of IRL [$N_1$; cf. (25)].

Additional hyperparameters and initial weights are given as follows. For SPI, we use $\alpha_1 = 10$ in the critic tuning law (26), and $\alpha_2 = 10$ and $F_2 = 5\, I_{N_1}$ in the actor tuning law (28). For IRL, we initialize the critic weights $c_0$ to implement the policy $\mu_0$ (53) in the actor structure (13) for the first two evaluations and the policy $\mu_0$ (54) in the third evaluation. The actor weights $w_0$ for SPI are set to identical values as the IRL critic weights $c_0$ for all evaluations. This initializes IRL and SPI with the same policy $\mu_0$ in their shared actor network structure (13).

*Remark 9 (SPI—Critic Weight Initialization):* Technically speaking, the initial critic weights $c_0$ for SPI may be selected independently of the initial actor weights $w_0$. However, as noted in Remark 3, the modified actor tuning law (28) implemented resembles a tracking control law, which makes the actor weights track the critic weights. Thus, if the initial

critic weights $c_0$ do not correspond to a stabilizing policy in the actor network (13), we observe that closed-loop instability results as the actor weights $w(t)$ converge to the destabilizing critic weights $c(t)$. Thus, in this work, we initialize critic and actor weights to the same values.

For RADP, we initialize the actor weights $w_0 = 0$ as per Algorithm 4. Finally, for CT-VI as per Algorithm 5, we initialize the actor weights $w_0 = 0$, Hamiltonian weights $v_0 = 0$, and critic weights $c_0$ such that the initial critic network is given by $\hat{V}(x, c_0) = x_1^2 + x_2^2$.

### B. Setup—Cart Inverted Pendulum System

The fourth and final evaluation considers the cart inverted pendulum system [107]

$$\ddot{x} = \frac{m_p l \dot{\theta}^2 \sin\theta - m_p g \sin\theta\cos\theta + u}{m_c + m_p \sin^2\theta}$$
$$\ddot{\theta} = \frac{-\ddot{x}}{l}\cos\theta + \frac{g}{l}\sin\theta \tag{58}$$

where $x$ is the cart position (measured in meters), $\theta$ is the pendulum angular displacement (measured in radians relative to the upright position, clockwise positive), $u$ is the horizontal force applied to the cart, $m_c$ is the mass of the cart, $m_p$ is the mass of the pendulum, $l$ is the length of the pendulum, and $g$ is the gravitational field constant. We use the standard values $m_c = 1$ kg, $m_p = 0.1$ kg, $l = 0.5$ m, and $g = 9.81$ m/s$^2$. This simplified model assumes no cart or pendulum friction, and the mass of the pendulum is concentrated at a point at its end. With state variables $[\,x_1 \;\; x_2 \;\; x_3 \;\; x_4\,]^{\mathrm{T}} = [\,x \;\; \dot{x} \;\; \theta \;\; \dot{\theta}\,]^{\mathrm{T}}$, the dynamical equations (58) may be expressed in state-space form (1) as

$$f(x) = \begin{bmatrix} x_2 \\ \dfrac{m_p \sin x_3 \left(l x_4^2 - g\cos x_3\right)}{m_c + m_p \sin^2 x_3} \\ x_4 \\ \dfrac{\sin x_3 \left(\dfrac{g}{l}(m_c + m_p) - m_p x_4^2 \cos x_3\right)}{m_c + m_p \sin^2 x_3} \end{bmatrix}$$
$$g(x) = \frac{1}{m_c + m_p \sin^2 x_3}\begin{bmatrix} 0 \\ 1 \\ 0 \\ -\dfrac{\cos x_3}{l} \end{bmatrix}. \tag{59}$$

We use the standard Q-R cost structure $r(x, u) = x^T Q x + u^T R u$ and the natural choice $Q = I_4$ and $R = 1$.

*1) Basis Functions:* We note, for this example, that the running cost $r(x, u)$ is an even function of $(x, u)$, and the state dynamics $f(x) + g(x)u$ is an odd function of $(x, u)$. It can be checked that this implies the optimal value function $V^*$ is even, and the optimal policy $\mu^*$ is odd. With this insight, we select as our critic basis $\{\phi_j\}_{j=1}^{N_1}$ the even monomials of total degree two (i.e., $N_1 = 10$). We select for the actor basis $\{\psi_j\}_{j=1}^{N_2}$ the odd monomials of total degree less than or equal to three (i.e., $N_2 = 24$). Finally, we select the Hamiltonian basis functions $\{\theta_j\}_{j=1}^{N_3}$ also as the even monomials of total degree two (i.e., $N_3 = N_1 = 10$). We believe these selections to be the natural first-choice for a designer beginning their analysis of this system.

*2) Initial Stabilizing Policy $\mu_0$:* For the initial stabilizing policy $\mu_0$, we examine the linearization $(A, B)$ about the origin and design for it the linear quadratic regulator (LQR) full-state feedback control law $u(x) = Kx$, where $K = R^{-1}B^T P \in \mathbb{R}^{m\times n}$, $P \in \mathbb{R}^{n\times n}$, and $P = P^T > 0$ is the unique positive definite solution of the Riccati equation

$$0 = A^T P + PA - PBR^{-1}B^T P + Q. \tag{60}$$

We use the locally stabilizing nonlinear control law

$$\mu_0(x) = -R^{-1}g^T(x)Px. \tag{61}$$

*3) Exploration Noise $e$:* We default to the exploration noise $e(t) = e_1(t) = 5\cos(t)$ (55). With the policy $\mu_0$ (61) and initial condition $x_0 = [1, 0, 15°, 0]^{\mathrm{T}}$, this exploration noise yields stable cart position oscillations on the order of 3 m and pendulum oscillations on the order of 20°, so, qualitatively, the noise allows the four algorithms to collect rich trajectory data under the initial stabilizing policy without exciting the pendulum instability. We observe the initial policy to achieve stability for exploration noise amplitudes of up to ∼7.5.

*4) Hyperparameter and Weight Initialization:* For IRL, we choose a learning time $t_f = 5$ s. Since we have increased the number of critic basis functions to $N_1 = 10$, we increase the number of data points per iteration from $l = 10$ to $l = 15$. The number of iterations $i^*$ for IRL is changed experimentally, so we leave the discussion to Section X. For SPI, we choose a shorter collection window of $t_f = 100$ s for reasons explained in Section X. We choose all SPI tuning gains/matrices identical to those of Section VIII-A (modulo dimension increases). For RADP, we use $i^* = 20$ iterations; otherwise, all hyperparameters for RADP and CT-VI are chosen identically to those of the first three evaluations. Finally, all weight initializations are performed identically to Section VIII-A, now corresponding to the policy $\mu_0$ (61).

## IX. Performance Evaluation and Analysis—Second-Order System

### A. Evaluation 1—Exact Minimal Bases

*1) Results:* **IRL** behaves well in regard to approximation and weight convergence. To illustrate this point, Table V displays the mean, max, and standard deviation critic weight errors $\|c^* - c_f\|$ observed across the initial condition sweep for the exploration noise $e_3$ (56) in the first column, and the last three columns correspond to the three respective weights in the basis (46). IRL exhibits a final critic weight error $\|c^* - c_f\|$ of less than $10^{-9}$ for all trials. It also has a short average run time of around 0.15 s per trial, as seen in Table VI, which shows the average run time of each algorithm over the IC sweep for the first evaluation.

Next, we discuss conditioning. Table VII shows the IC sweep average condition number of the matrices pseudoinverted for IRL, RADP, and CT-VI for the first three evaluations. In the case of IRL and RADP, the respective matrices $\mathbf{A}^i_{\mathrm{IRL}}$ (25) and $\mathbf{A}^i_{\mathrm{RADP}}$ (31) change numerically with an iteration count $i$, so we have taken the IC sweep average

TABLE V
EVAL. 1: CRITIC WEIGHT ERROR FOR NOISE $e_3$ (56)

| Alg. | Data | $\|c^* - c_f\|$ | $\|c_1^* - c_{f,1}\|$ | $\|c_2^* - c_{f,2}\|$ | $\|c_3^* - c_{f,3}\|$ |
|---|---|---|---|---|---|
| IRL | Mean | 8.9e-11 | 5.15e-15 | 1.74e-14 | 8.9e-11 |
| | Max | 8.04e-10 | 1.68e-14 | 1.23e-13 | 8.04e-10 |
| | Std. | 1.78e-10 | 3.37e-15 | 2.97e-14 | 1.78e-10 |
| SPI | Mean | 1.07e-04 | 3.12e-06 | 7.35e-05 | 7.78e-05 |
| | Max | 1.33e-04 | 4e-06 | 9.15e-05 | 9.64e-05 |
| | Std. | 6.09e-06 | 4.74e-07 | 4.21e-06 | 4.45e-06 |
| RADP | Mean | 6.77 | 0.925 | 6.18 | 2.62 |
| | Max | 7.56 | 1.03 | 6.92 | 2.89 |
| | Std. | 0.348 | 0.0628 | 0.333 | 0.127 |
| CT-VI | Mean | 0.0169 | 0.00382 | 0.0155 | 0.00539 |
| | Max | 0.471 | 0.105 | 0.434 | 0.15 |
| | Std. | 0.0744 | 0.0166 | 0.0686 | 0.0236 |

TABLE VI
EVAL. 1: AVERAGE RUN TIME (S)

| Alg. | $e_1$ | $e_2$ | $e_3$ |
|---|---|---|---|
| IRL | 0.14 | 0.15 | 0.15 |
| SPI | 2.96 | 2.99 | 3.17 |
| RADP | 0.20 | 0.21 | 0.23 |
| CT-VI | 9.15 | 17.21 | 23.00 |

over the final-iteration matrices $\mathbf{A}_{\text{IRL}}^{i^*}$ and $\mathbf{A}_{\text{RADP}}^{i^*}$. IRL struggles with significant conditioning issues, having an average final-iteration condition number on the order of $10^5$.

**SPI** also exhibits good convergence properties (see Table V). We note that conditioning analyses do not apply to SPI, which is an adaptive/gradient-descent-based method and does not require regression in its weight updates. However, these dynamic weight updates require a significantly longer collection window $t_f = 500$ s to converge, and thus, SPI takes a much longer 3 s on average to run (see Table VI).

**RADP** achieves good approximation performance for the two smaller noises $e_1$ and $e_2$ (55) (comparable numerically with that of IRL and SPI), but, as seen in Table V, this degrades for the large exploration noise $e_3$ (56). Its mean critic weight error $\|c^* - c_f\|$ is 6.77 with a standard deviation of only 0.348, suggesting that the error is large across the sweep. RADP fares quite well with conditioning (see Table VII), which remains on the order of 10 for all exploration noises. It also has a short average run time of around 0.2 s per trial (see Table VI).

**CT-VI** converges consistently overall for the smaller exploration noises $e_1$ and $e_2$ (55), with max final critic weight error $\|c^* - c_f\|$ of less than 0.01 and max actor weight error $\|w^* - w_f\|$ of 0.238. The maximum Hamiltonian weight error $\|v^* - v_f\|$ is higher at 2.08 (observed for $e_2$), but CT-VI exhibits a mean error $\|v^* - v_f\|$ of only 0.0672 for this exploration noise, so the peak of 2.08 is an outlier. For the large excitation $e_3$ (56), CT-VI performs well overall, with a mean critic weight error $\|c^* - c_f\|$ of only 0.0169 (see Table V), but there are outliers in which the error gets as large as 0.471 at maximum.

Like IRL, CT-VI struggles with conditioning. In Table VII, conditioning of the matrix $\kappa(K_\phi(t_f))$ averages on the order of $10^2$ for all exploration noises and $\kappa(K_\sigma(t_f))$ averages on the order of $10^6$ for $e_1$ and on the order of $10^4$ for $e_2$ and $e_3$. Since $K_\phi(t_f) \in \mathbb{R}^{N_1 \times N_1}$ with $N_1 = 3$ and $K_\sigma(t_f) \in \mathbb{R}^{(N_2+N_3)\times(N_2+N_3)}$ with $N_2 + N_3 = 8$, it is expected that, in general, the conditioning of $K_\sigma(t_f)$ will fare worse than that of $K_\phi(t_f)$. Evidence of the demanding computational requirements of CT-VI is further witnessed in Table VI. CT-VI requires by far the longest run time at 10–20 s per trial. We note, in addition, that run time for CT-VI increases substantially with increasing exploration noise amplitude, more than doubling on average over the exploration noises tested.

*2) Insights:*

*a) Algorithms perform well for lower excitations in baseline example:* For the lower excitations $e_1$ and $e_2$ (55), all algorithms successfully converge to the optimal weights regardless of the initial condition selected (with the exception of a few outliers). Since this example is low-order, and we have chosen exact bases, the alignment between theoretical guarantees and observed synthesis is to be expected.

*b) Algorithm structure significantly impacts conditioning and numerical performance:* We point the reader to Fig. 1(a), which displays the state trajectory $x_1(t)$ corresponding to exploration noise $e_1$ (55) and IC $x_0 = [1, 1]^{\mathrm{T}}$. RADP and CT-VI use the same trajectory data to perform their learning, yet CT-VI's conditioning is three orders of magnitude worse than RADP's (cf. Table VII). This simple example illustrates the stark impact of algorithm structure on inherent conditioning and numerical properties. For this reason, conditioning should be considered in the design process using similar approaches presented in this study, not just in post-hoc analysis.

*c) Avoid large excitation for RADP:* We offer designers this general observation of RADP: across all evaluations conducted, RADP exhibits convergence issues for large exploration noises. This is of practical concern to real-world designers who are, in general, concerned with achieving *sufficient* excitation to meet PE requirements. We caution that overexcitation is a significant phenomenon, which occurs even for low-order academic examples.

*3) Limitations:*

*a) CT-VI numerical complexity issues associated with tuning structure:* In this simple example, we have already experienced divergence issues with CT-VI for the large exploration noise $e_3$, which necessitates the addition of the low-frequency perturbation $\cos(0.1\ t)$ in (56). We believe that these issues can be explained by examining the tuning procedure of CT-VI in (32) and (34). The Hamiltonian weights $\overline{v}(s)$ yielded by the pseudoinversion of $K_\sigma(t_f)$ in (34) are nested in the integral (32) involved in the pseudoinversion of $K_\phi(t_f)$. The pseudoinversion of $K_\phi(t_f)$ yields the derivative $d/(ds)c(s)$ in the critic weight tuning law (32), which itself must be integrated with respect to the weight tuning time $s$. In sum, the CT-VI algorithm comprises an alternating chain of two pseudoinversions sandwiched between three nested (vector-valued) integrations. When combined with the high condition numbers seen in Table VII, we conclude that the weight convergence issues stem from these numerical considerations. The run time of CT-VI exceeds that of IRL and RADP by a factor of 100 (see Table VI), a further empirical indication of numerical complexity issues.

*b) IRL's lack of probing noise causes data quality degradation as the state is regulated to origin:* The two PI-based algorithms (IRL and RADP) exhibit vastly different conditioning properties (cf. Table VII). Returning to Fig. 1(a) offers clear explanation. Recall that IRL does not allow for the

TABLE VII

EVALS. 1–3: MEAN CONDITION NUMBER

| Alg. | Matrix | Eval. 1 | | | Eval. 2 | | | Eval. 3[a] |
|---|---|---|---|---|---|---|---|---|
| | | $e_1$ | $e_2$ | $e_3$ | $e_1$ | $e_2$ | $e_3$ | $e_1$ |
| IRL[b] | $\mathbf{A}_{IRL}^{i^*}$ | 1.48e+05 | " | " | 5.62e+05 | " | " | 5.75e+11 |
| RADP | $\mathbf{A}_{RADP}^{i^*}$ | 34.94 | 22.21 | 17.19 | 19.68 | 23.10 | 25.61 | 155.45 |
| CT-VI | $K_\phi(t_f)$ | 207.66 | 631.69 | 413.30 | 366.14 | 866.12 | N/A[c] | 6.17e+04 |
| | $K_\sigma(t_f)$ | 6.96e+06 | 5.49e+04 | 4.87e+04 | 6.88e+06 | 5.49e+04 | N/A[c] | 1.06e+07 |

[a]Exploration noises $e_2, e_3$ not executed for Evaluation 3 (cf. Section IX-C for details).
[b]No exploration noise injected for IRL, so we put its data under the $e_1$ column and leave its other entries blank.
[c]CT-VI not executed for exploration noise $e_3$ in Evaluation 2 (cf. Section IX-B for details).

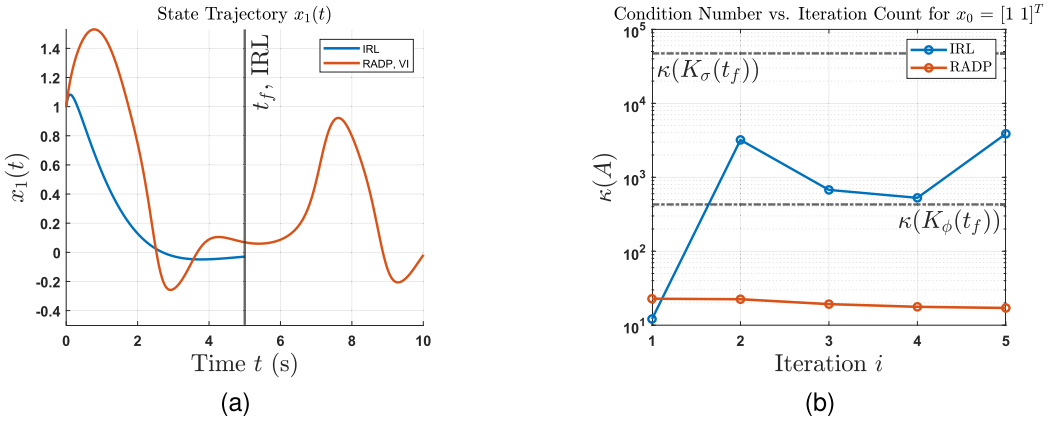



Fig. 1. Eval. 1: data for exploration noise $e_1$ (55) and IC $x_0 = [1, 1]^{\mathrm{T}}$. (a) Learning-phase state trajectory $x_1(t)$. (b) Condition number versus iteration count.

insertion of a probing noise. As the system is simulated under the initial stabilizing policy $\mu_0$ (53) and successive stabilizing policies $\hat{\mu}_1, \ldots, \hat{\mu}_{i^*}$, the state is regulated to the origin. Meanwhile, examining the form of the $i$th iteration weight update matrix $\mathbf{A}_{\mathrm{IRL}}^i \in \mathbb{R}^{l \times N_1}$ (25) ($i = 0, \ldots, i^*$), we see that continuity of the state trajectory $x(t)$ and continuity of the critic basis functions $\Phi(x)$ imply that $\mathbf{A}_{\mathrm{IRL}}^i$ vanishes as variations in the state trajectory samples $\{x(t_k^i)\}_{k=0}^l$ vanish. This explains the steep upward trend in IRL's iterationwise condition number plotted in Fig. 1(b), beginning at around 10 for $i = 1$ and increasing to almost $10^4$ for $i = 5$. It is for this reason that a shorter collection window $t_f = 5$ s is necessary for IRL. If the default $t_f = 10$ s used for the other methods is chosen, the condition number $\kappa(\mathbf{A}_{\mathrm{IRL}}^{i^*})$ regularly exceeds $10^8$ for this example. We, hence, observe the lack of probing noise insertion as a fundamental limitation of the IRL methodology, even though, strictly speaking, it does not have a PE requirement (cf. Table II and Remark 8). As a result of probing noise insertion, RADP has access to richer trajectory data, and its conditioning remains low.

For the same reasons, we make note that the conditioning of IRL fares is significantly worse for ICs chosen nearer the origin. Thus, Fig. 1(b) with the IC $x_0 = [1, 1]^{\mathrm{T}}$ is a best-case across the sweep. In order to combat this conditioning issue, for each iteration $i$, Vrabie and Lewis [19, Sec. 6.2] collect data from multiple trajectories with randomized initial conditions. While this is a legitimate learning procedure, strictly speaking, it is not permissible in an online learning scenario.

*Remark 10* **(Concluding Remarks for Evaluation 1: The Curse of Conditioning in CT-RL):** After considering Table VII, Fig. 1(b), and the subsequent analysis, we wish to characterize whether the observed conditioning issues are *emergent* phenomena or if they are *inherent* to the CT-RL methodologies themselves.

Certainly, this is not an issue of *dimensionality*. The system (43) is second-order and single-input, and the basis dimensions $N_1 = 3$, $N_2 = 2$, and $N_3 = 6$ are chosen to be minimal for this problem. Real-world applications will inevitably be higher dimensional than this one.

Neither is this an issue of *approximation*. Indeed, the example was constructed such that the optimal value $V^*$ and policy $\mu^*$ are available in closed form, and the bases chosen can achieve exact approximation.

Having ruled out the usual culprits of problem dimension and approximation error, we consider that the fundamental conditioning issues illustrated here are intrinsic to the algorithms themselves. In many respects, the problem structure of this evaluation represents the *best-case* performance that could be hoped for from these algorithms. Unfortunately, we shall soon see that the underlying numerics compound subsequent issues of dimensionality and approximation, altogether severely limiting the applicability of these CT-RL methods to real-world design problems.

### B. Evaluation 2—Critic Basis With $N_1 = 4$ Terms

*1) Results:* **IRL** achieves good approximation performance comparable with that of the first evaluation for all probing noises. We present critic weight error data for the exploration noise $e_3$ (55) in Table VIII. IRL achieves a critic weight error $\|c^* - c_f\|$ of less than $10^{-9}$ at max. Meanwhile, examination of Table VII shows that the addition of one critic basis function has increased IRL's average conditioning by a factor of five to $5.6 \times 10^5$. For the sake of comparison, we have again plotted

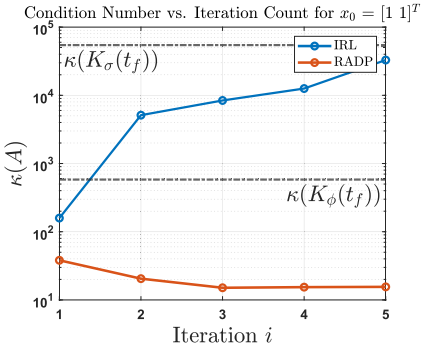



Fig. 2. Eval. 2: condition number versus iteration count for exploration noise $e_1$ (57) and IC $x_0 = [1, 1]^{\mathrm{T}}$.

TABLE VIII
EVAL. 2: CRITIC WEIGHT ERROR FOR NOISE $e_3$ (55)

| Alg. | Data | $\|c^* - c_f\|$ | $\|c_1^* - c_{f,1}\|$ | $\|c_2^* - c_{f,2}\|$ | $\|c_3^* - c_{f,3}\|$ | $\|c_4^* - c_{f,4}\|$ |
|---|---|---|---|---|---|---|
| IRL | Mean | 1.54e-10 | 3.7e-14 | 7.8e-14 | 1.54e-10 | 8.91e-14 |
| | Max | 1.24e-09 | 2.3e-13 | 3.94e-13 | 1.24e-09 | 4.95e-13 |
| | Std. | 2.56e-10 | 5.69e-14 | 8.66e-14 | 2.56e-10 | 1.16e-13 |
| SPI | Mean | 0.0168 | 0.000803 | 0.00849 | 0.0143 | 0.00248 |
| | Max | 0.0176 | 0.000926 | 0.00883 | 0.0149 | 0.00268 |
| | Std. | 0.00022 | 3.59e-05 | 0.000132 | 0.000199 | 7.46e-05 |
| RADP | Mean | 2.87 | 0.468 | 2.62 | 1.09 | 0.0224 |
| | Max | 6.73 | 1.06 | 6.14 | 2.55 | 0.0997 |
| | Std. | 2.76 | 0.45 | 2.51 | 1.04 | 0.0217 |

condition number versus iteration count for the exploration noise $e_1$ (57) and IC $x_0 = [1, 1]^{\mathrm{T}}$ in Fig. 2. Comparison with Fig. 1(b) corroborates the trend in Table VII that conditioning has degraded across the board. In particular, the conditioning of IRL now approaches the $10^5$ mark for this trial, almost a tenfold increase from the previous evaluation. This illustrates that IRL suffers from scalability issues.

Finally, the addition of one basis function has not increased the run time of any of the algorithms significantly, so we have omitted run time data here for the sake of brevity. Note, however, that we wished to carry out more thorough analyses of run time performance, but, for all algorithms, weight convergence issues halted these evaluations before we could introduce sufficient dimensional scaling. Given that these algorithms repeatedly use pseudoinversion, which scales as $\sim\mathcal{O}(l^3)$, in principle, run time scaling will likely present significant challenges in the future.

**SPI** achieves a max critic weight error $\|c^* - c_f\|$ of 0.0465 for exploration noises $e_1$ (57) and $e_2$ (55), and 0.0176 for $e_3$ (see Table VIII). For all exploration noises, its actor weight error $\|w^* - w_f\|$ remains less than 0.001 at max.

*RADP:* We again observe the pattern that RADP performs comparably to IRL and SPI for small excitations but struggles for the large excitation $e_3$ (55), with an average critic weight error $\|c^* - c_f\|$ of 2.87, max of 6.73, and standard deviation of 2.76 (see Table VIII). Comparison of these numbers with those of Table V shows that the max error is comparable for the two evaluations. Meanwhile, the mean error for this evaluation (2.87) is smaller than that of the previous evaluation (6.77). However, the present evaluation standard deviation (2.76) is a factor of ten higher than previous (0.348). This anecdotally suggests that the addition of a basis function to the critic (47) has made the weight convergence of RADP more volatile for large exploration noises. In spite of these issues, RADP does zero the redundant basis function $\phi_4(x) = x_1x_2$ (47) for $e_3$ quite well overall (see Table VIII). Finally, RADP's conditioning has remained low at approximately 20 (see Table VII).

**CT-VI** is not run for the exploration noise $e_3$ (55) due to convergence issues (cf. Section VIII-A for discussion), so its data are absent in Table VIII. It does achieve good convergence properties for the smaller two excitations, for which its critic, actor, and Hamiltonian weight errors remain less than 0.01 at max. Examining Table VII, the conditioning of $K_\sigma(t_f)$ is nearly identical to the previous evaluation, which is expected since the bases $\Psi(x)$ (49) and $\Theta(x)$ (51) composing this matrix have remained unchanged. On the other hand, the condition number $\kappa(K_\phi(t_f))$ has increased from 208 in the previous evaluation to 366 for the exploration noise $e_1$ (+76%) and from 632 to 866 for $e_2$ (+37%).

*2) Limitations: Convergence and Conditioning Degradation With Addition of One Critic Basis Function: Dimensional Scalability Concerns:* IRL still performs well in terms of convergence, but its conditioning has degraded significantly as a result of the addition. SPI, the gradient-descent algorithm for which conditioning issues do not apply, fares the best overall; its convergence properties remain largely unchanged. RADP displays a slight degradation in conditioning and substantial increases in weight volatility. Finally, CT-VI exhibits weight divergence on both the low- and high-amplitude ends of the exploration noise spectrum, due, in large part, to condition number increase of the matrix $K_\phi(t_f)$ (33) structurally affected by the additional critic basis function. Now, all that remains is a central band of amplitudes around $e_2(t) = 10\cos(t)$ for which this algorithm can run properly.

### C. Evaluation 3—Realistic Choice of Critic Basis

Now that the optimal value function $V^*$ (44) cannot be approximated exactly by our choice of critic basis (48), we first establish that good approximation is still achievable. We perform a linear regression (LR) of the optimal value function $V^*$ (44) in the basis functions (48) over the box $[-1, 1]^2$, yielding the $L^2([-1, 1]^2)$-optimal weights

$$c_{\mathrm{lr}} = \begin{bmatrix} 0.2140 & 1.7436 & 0 & 0.2 & 0 & 0.1905 & 0 \end{bmatrix}^{\mathrm{T}}. \quad (62)$$

The associated LR critic $\hat{V}_{\mathrm{lr}}(x) = \hat{V}(x, c_{\mathrm{lr}})$ is plotted alongside the optimal value function $V^*$ (44) in Fig. 3(a). The approximation achieved is quite accurate by visual inspection.

*1) Results:* **IRL and RADP:** We begin our study with the two PI-based algorithms. Comparison of the conditioning data in Table VII to the previous evaluation shows that the conditioning for IRL has increased six orders of magnitude from $5.6 \times 10^5$ to $5.8 \times 10^{11}$. RADP conditioning has fared better, but it still has increased by a factor of eight from 20 to 155. To analyze convergence properties, we plot the critic weights $c_i$ versus iteration count $i$ for IRL and RADP in Fig. 4(c) and (d), respectively. These responses correspond to the exploration noise $e_1$ (55) and IC $x_0 = [1, 1]^{\mathrm{T}}$, which we observe as qualitatively representative.

One observes two distinct regimes of behavior for IRL in Fig. 4(c): an initial phase from $i = 1$ to $i = 10$ iterations where the weights oscillate and drift, and a second phase

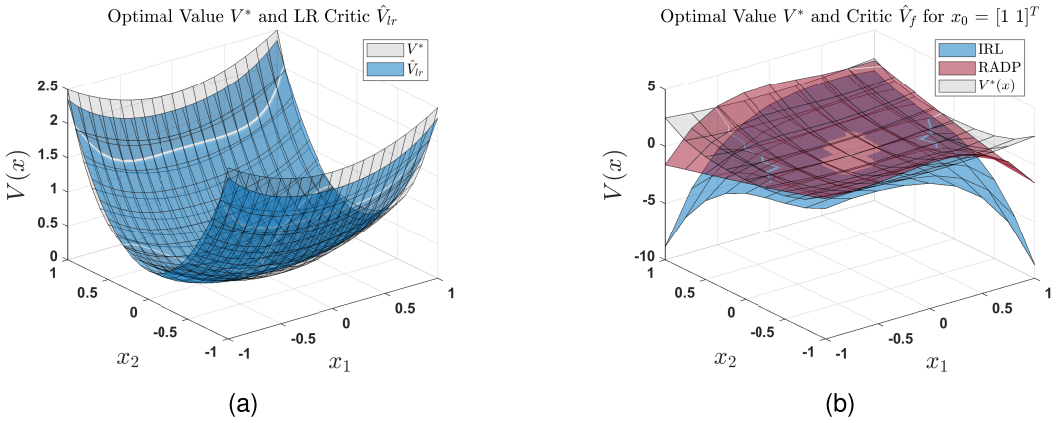



Fig. 3. Eval. 3. (a) Optimal value $V^*$ and LR critic $\hat{V}_{\text{lr}}$. (b) Optimal value $V^*$ and final critic $\hat{V}_f$ for exploration noise $e_1$ (55), IC $x_0 = [1, 1]^{\text{T}}$.

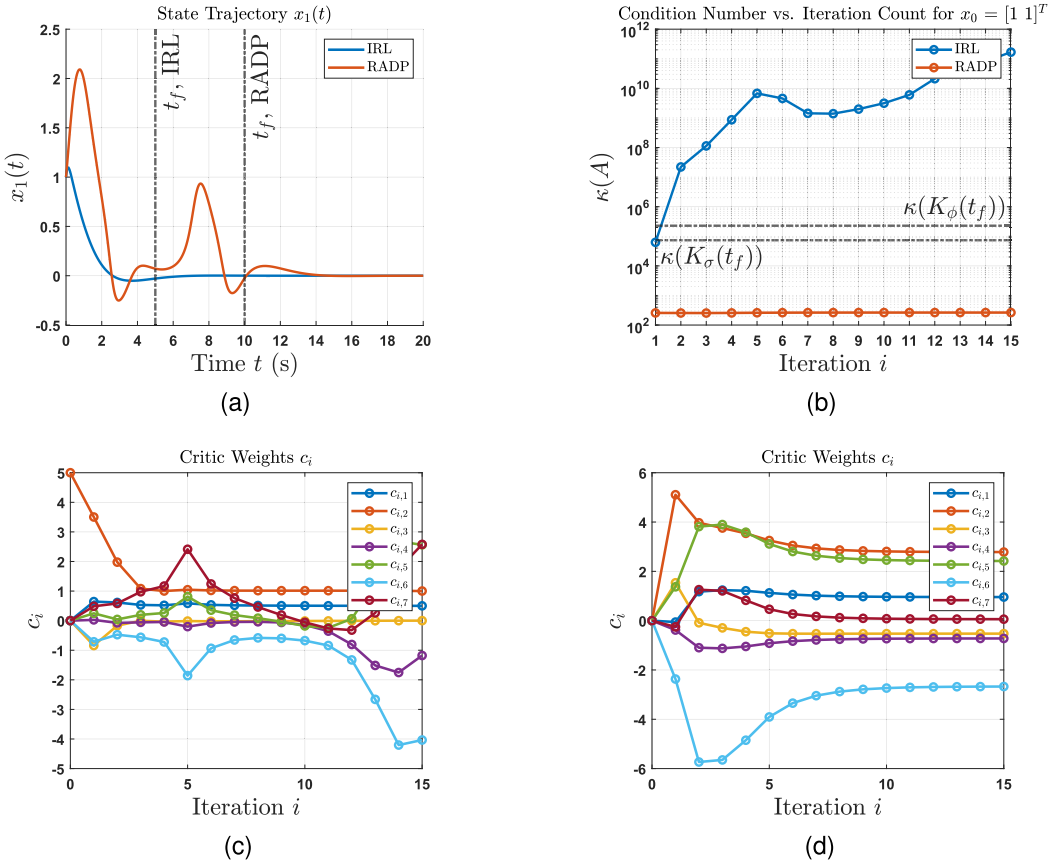



Fig. 4. Eval. 3: data for exploration noise $e_1$ (55) and IC $x_0 = [1, 1]^{\text{T}}$. Top: (a) state trajectory $x_1(t)$ and (b) condition number versus iteration count. Bottom: critic weights $c_i$ versus iteration count for (c) IRL and (d) RADP.

from $i = 10$ to $i = 15$ iterations where the weights begin to diverge. The reason for the latter weight divergence is clear upon viewing the corresponding $x_0 = [1, 1]^{\text{T}}$ state trajectory data in Fig. 4(a). For IRL's learning on $t = [0, 5]$, the latter third of the trajectory is near zero. We have, thus, encountered the same scenario as in Section IX-A, where IRL struggles with conditioning in latter iterations as the state trajectory approaches the origin. Now that the basis order has increased and the basis functions can no longer achieve exact approximation, IRL cannot perform its learning quickly enough to outpace data quality degradation.

The observed weight oscillations/drifting in the first $i = 10$ iterations of Fig. 4(c) are readily explained by the iterationwise condition number of $\mathbf{A}^i_{\text{IRL}}$ plotted in Fig. 4(b). Here, we observe that, even for early iterations (when the quality of trajectory data is relatively high), the conditioning of the IRL problem exceeds $10^4$ across the board. By comparison to the previous evaluation (for which conditioning begins on the order of $10^2$), this is a 100-fold increase.

Table IX displays the mean and standard deviation of each of the final critic weights $c_{f,j}$, $j = 1, \ldots, 7$. As a result of poor conditioning, IRL exhibits large standard deviations for the latter four weights. However, we do note that the weights $c_{f,1}$, $c_{f,2}$, and $c_{f,3}$ exhibit mean values of 0.5, 1, and $\sim$0, respectively, with near-zero standard deviation. These mean values agree with the respective values of the optimal weights $c^*$ for the exact basis (47). The critic weights for RADP, by contrast, have converged in Fig. 4(d). Unfortunately, the

values to which the weights converge are not consistent across the IC sweep (see Table IX).

Having rounded off the weight convergence analysis of IRL and RADP, it now remains to examine the approximation errors of the final critics $\hat{V}_f(x) = \hat{V}(x, c_f)$ (10); they produce in relation to the optimal value function $V^*$ (44). We display these functions in Fig. 3(b) for exploration noise $e_1$ (55) and IC $x_0 = [1, 1]^{\mathrm{T}}$. Unfortunately, both algorithms exhibit wide variation. Indeed, neither critic $\hat{V}_f$ is even positive definite.

**SPI and CT-VI:** Unfortunately, the tuning of both of these algorithms breaks down for this example. We run SPI for the IC $x_0 = [1, 1]^{\mathrm{T}}$. After $t = 35$ s of tuning, the weights are $c(35) \approx w(35) = [0.771, 1.972, 0.345, -0.234, -0.154, -0.0826, -0.179]^{\mathrm{T}}$. If the simulation is continued beyond $t = 35$ s, the state trajectory diverges. A similar phenomenon prevails regardless of the IC $x_0$ in the sweep, the eventual divergence occurring within the first couple dozen seconds of simulation. We test the policy $\hat{\mu}(x, w(35))$ (13) corresponding to the actor weights $w(35)$ without any exploration noise to find that it is indeed stabilizing on $[-1, 1]^2$. Thus, although the SPI tuning has kept the actor weights $w(t)$ stabilizing from $t = 0$ to $t = 35$, the actor network shortly after $\hat{\mu}(x, w(35))$ is unable to reject the exploration noise $e_1$ (55), amidst the unstable dynamics of the system (43). This is discouraging, given that the exploration noise $e_1$ (55) chosen has the smallest amplitude of any tested in this work. Indeed, SPI performs quite well for this exploration noise in previous evaluations.

For CT-VI, regardless of our choice of hyperparameters (i.e., probing noise $e$ and learning time $t_f$), the weight tuning laws (32) and (34) diverge. We present CT-VI's average condition number data for the standard choices $e = e_1$ (55) and $t_f = 10$ s in Table VII. Compared to the previous evaluation, the average condition number of $K_\sigma(t_f)$ for the exploration noise $e_1$ has nearly doubled from $6.9 \times 10^6$ to $10.6 \times 10^6$. Due to the increase in critic basis dimension from $N_1 = 4$ to $N_1 = 7$, the condition number of $K_\phi(t_f)$ has increased two orders of magnitude from 366 to $6.2 \times 10^4$. Unfortunately, we conclude that these sheer condition numbers render CT-VI unusable for this example.

*2) Limitations:*

*a) IRL lack of probing noise results in hyperparameter deadlock:* A designer assessing the performance of IRL in Fig. 4(c) might be tempted to reduce the learning time $t_f$ in hopes of restricting to higher quality trajectory data, thereby improving weight convergence. Alas, these efforts are not fruitful. Recall in Section IX-A that we deduced that the regression matrix $\mathbf{A}^i_{\mathrm{IRL}}$ (25) vanishes as variations in the state trajectory samples $\{x(t^i_k)\}^l_{k=0}$ vanish. This condition occurs as the differences in sample time instants $\{t^i_k\}^l_{k=0}$ converge to zero, i.e., as learning time $t_f \to 0$, final iteration $i^* \to \infty$, or the number of samples $l \to \infty$. On the one hand, the increase in final iteration count from $i^* = 5$ previously to $i^* = 15$ here is unavoidable: generally speaking, higher order problems require more iterations to converge, and we observe this example as no exception. On the other hand, there is little room to reduce the number of samples from $l = 10$ since $l \geq N_1 = 7$ is needed for full column rank in the weight update (25). Thus, both of these hyperparameters have been virtually minimized to ensure the best conditioning possible. It is perhaps of no surprise then that we observe reducing the learning time $t_f$ (or increasing sample count $l$ and/or final iteration $i^*$, all of which we tried), which only exacerbates the poor early iteration conditioning seen in Fig. 4(b), which, in turn, magnifies the early iteration weight oscillations in Fig. 4(c). With all options exhausted, we, unfortunately, conclude that the designer is deadlocked in an effort to balance IRL hyperparameter selection with the underlying numerics.

*b) Conditioning ceiling causes the discrepancy between theoretical approximation results and observed approximation performance:* Both IRL and RADP guarantee uniform approximation of the optimal value function $V^*$ on compact subsets (cf. Theorems 4 and 8, respectively). We note the subtlety that these approximation results are sufficient conditions that are not constructive; i.e., they do not furnish estimates of the number of basis functions $N_1$ required to achieve a desired approximation $\epsilon > 0$. The critic approximation issues observed in Fig. 3(b) reveal that, when increasing the critic basis dimension $N_1$, a practical conditioning ceiling incapacitates approximation performance long before the theoretical threshold can be attained.

*c) SPI excitation requirements exceed closed-loop stability thresholds:* Unfortunately, the SPI instability issue cannot be remedied by decreasing the exploration noise amplitude. For amplitudes ∼3.5 and above, the state eventually diverges. For amplitudes below this threshold, the weight tuning freezes due to insufficient excitation. We further tried modulating the exploration noise by a decaying exponential term; i.e., $e(t) = 5\cos(t)e^{-at}$ for decay rate $a > 0$. Unfortunately, these efforts yield much the same qualitative behavior as varying the exploration noise amplitude. Thus, for SPI, we are unable to find a balance between stability and sufficient excitation. We believe these issues to be, in part, due to the modified actor tuning law (28), which we had to adopt in this work (cf. Remark 3). Due to the actor tuning modifications, the original stability guarantees provided in [20] no longer apply.

*Remark 11:* **Concluding Remarks for Evaluations 1–3, Performance Limitations:** For this third evaluation, the two PI-based algorithms, IRL and RADP, can be successfully run across the IC sweep. However, underlying conditioning issues either render weight convergence inconsistent (RADP) or prevent convergence entirely (IRL). The two dynamic tuning algorithms, SPI and CT-VI, cannot execute to termination due to either closed-loop instability (SPI) or weight divergence (CT-VI). In the case of SPI, our probing noise selection has to meet the conflicting demands of disturbance rejection and sufficient excitation, for which a suitable balance cannot be sought. In the case of CT-VI, conditioning degradation associated with increased problem complexity has rendered its tuning laws numerically intractable. The new phenomena observed that, for SPI aside, all of the key issues witnessed here are direct outgrowths of the novel diagnosis that we performed in our first evaluation, now compounded by the long-understood curses of dimensionality and approximation.

## X. Performance Evaluation and Analysis—Cart Inverted Pendulum System

Having thoroughly analyzed the second-order academic example (43), we now apply the fundamental design insights

TABLE IX
EVAL. 3: CRITIC WEIGHT MEAN AND STANDARD DEVIATION

| Alg. | Data | $c_{f,1}$ | $c_{f,2}$ | $c_{f,3}$ | $c_{f,4}$ | $c_{f,5}$ | $c_{f,6}$ | $c_{f,7}$ |
|---|---|---|---|---|---|---|---|---|
| IRL | Mean | 0.50 | 1.00 | -3.1e-04 | -0.58 | 0.87 | -1.89 | 1.04 |
| | Std. | 1.6e-04 | 9.0e-04 | 4.8e-04 | 0.57 | 1.03 | 1.33 | 1.07 |
| RADP | Mean | 0.70 | 1.63 | -0.24 | -0.32 | 0.54 | -1.11 | 1.03 |
| | Std. | 0.19 | 0.39 | 0.25 | 0.45 | 0.91 | 0.94 | 0.82 |

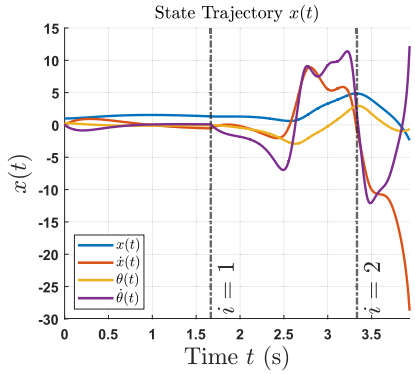



Fig. 5. Eval. 4: IRL state trajectory $x(t)$ for IC $x_0 = [1, 0, 15°, 0]^{\mathrm{T}}$.

gained to the cart inverted pendulum system (58), a benchmark control problem that has direct implications in real-world applications. We, at first, ran tests over the initial cart positions $x_0 = [-1{:}0.1{:}1]$ m, pendulum angles $\theta_0 = [-30{:}1{:}30]$ deg, zero initial translational/rotational velocities $\dot{x}_0 = 0$ m/s, and $\dot{\theta}_0 = 0$ deg/s. However, in our analyses, we encountered algorithm performance issues of similar nature regardless of the IC chosen. Thus, for the purposes of illustration, here, we examine the initial condition $x_0 = [1\text{ m}, 0, 15°, 0]^{\mathrm{T}}$.

*Results:* **IRL** We recall from the analysis presented in Section IX-C that, in general, IRL conditioning degrades with an increasing number of iterations $i^*$. Running IRL for this example at $i^* = 2$ is not problematic (although, of course, $i^* = 2$ is insufficiently many iterations for convergence). Running IRL for $i^* \geq 3$, on the other hand, yields a new issue, which has not been observed previously. We examine the state trajectory data in Fig. 5. Eventually, after $i = 2$ iterations, the critic weights are no longer stabilizing, and state trajectory diverges. Due to poor conditioning (observed on the order of $10^6$ for this example), the critic weights $c_i$ oscillate drastically, which, in turn, makes the policies $\hat{\mu}_i$ update abruptly and excite natural inverted pendulum instability. Corroborated by our insights gained in Section IX-C, decreasing the collection time $t_f$ or increasing the number of samples $l$ or the number of iterations $i^*$ only worsens the conditioning for this example.

**SPI** It exhibits new behavior in this example as well. The weight tuning is observed to freeze (i.e., the weights $c(t)$ and $w(t)$ remain virtually constant over $[0, t_f]$). After diagnosis, we find the culprit to be the term $\sigma_2/m_s^2 = \sigma_2/(\sigma_2^T\sigma_2 + 1)^2 \in \mathbb{R}^{N_1}$ in the critic tuning law (26) vanishing across the state trajectory $x(t)$. The critic tuning law (26) is a Levenberg–Marquardt algorithm modified by Vamvoudakis and Lewis [20], where $(\sigma_2^T\sigma_2 + 1)^2$ is used for normalization instead of the usual $(\sigma_2^T\sigma_2 + 1)$ (cf. [20, below eq. (23)]). Unfortunately, it is the squaring of this normalization term (alongside $\sigma_2 \in \mathbb{R}^{N_1}$ having a large norm across the trajectory), which has caused the vanishing.

Naturally, a designer will increase the amplitude of the probing noise $e$ or the critic tuning gain $\alpha_1 > 0$ in an attempt to unfreeze the critic weight learning. Unfortunately, increasing the amplitude of $e$ from 5 to 7.5 does not fix the issue, and as we have noted in Section VIII-B, increasing the amplitude beyond this point makes the closed-loop system unstable. Meanwhile, the critic weights still freeze when increasing the tuning gain by a factor of 100 from $\alpha_1 = 10$ to $\alpha_1 = 1000$ (and, for any intermediate selections, we tried). This analysis suggests that the modifications made to the Levenberg–Marquardt algorithm by Vamvoudakis and Lewis [20] in the tuning law (26) have frozen the critic weight learning for this higher order example.

**RADP** Regardless of the IC or hyperparameters chosen, we were unable to yield a stabilizing controller $\hat{\mu}_f = \hat{\mu}_{i^*+1}$ from RADP for this system. Similar to the third evaluation (cf. Table IX), RADP's final weight values are observed to be highly sensitive to IC $x_0$. Furthermore, for some ICs (e.g., $x_0 = [1, 0, 30°, 0]^{\mathrm{T}}$), the weights oscillate indefinitely and fail to converge. These convergence and stability issues are likely a result of conditioning. For $x_0 = [1, 0, 15°, 0]^{\mathrm{T}}$, we observed the condition number of $\mathbf{A}^i_{\mathrm{RADP}} \in \mathbb{R}^{l\times(N_1+N_2)}$ (31) to exceed $10^8$ for each iteration $i$ (an increase of six orders of magnitude from the previous evaluation). Given that the increment in critic basis dimension is relatively minor ($N_1 = 7$ in Section IX-C to $N_1 = 10$ here), we attribute this drastic conditioning degradation to the increase in actor basis dimension (from the minimal $N_2 = 2$ actor basis in previous evaluations to the realistic $N_2 = 24$ here).

**CT-VI** Previously identified numerical issues persist as we transition toward a real-world design problem. Regardless of hyperparameter selections, the weight responses diverge. For the natural choices listed in Section VIII-B, running the algorithm for the initial condition $x_0 = [1, 0, 15°, 0]^{\mathrm{T}}$ yields condition numbers of $7.8 \times 10^5$ for $K_\phi(t_f)$ and $5.2 \times 10^{13}$ for $K_\sigma(t_f)$. Unfortunately, conditioning has claimed its last victim, concluding our analysis of this example.

## XI. CONCLUSION AND DISCUSSION

This work provides an extensive review of four seminal CT-RL control methods (IRL [19], SPI [20], RADP [21], and CT-VI [22]), discussing the key theoretical assumptions and results. Our review shows these methods to be well-principled in approach, each offering an impressive suite of theoretical guarantees. All algorithms guarantee uniform convergence to the optimal value and policy, which extends beyond the baseline weight convergence results seen in the RL literature. Furthermore, each ensures closed-loop stability in one of its various notions. RADP even provides stability and robustness results.

These theoretical successes aside, our first-of-its-kind analytical framework illustrates through comprehensive evaluation

studies a fundamental divergence between CT-RL theoretical guarantees and controller synthesis performance. Our in-depth evaluations lead us to posit that it is ultimately this analysis/synthesis discrepancy, which underpins the fundamental CT/DT gap in the RL control literature. As we experienced difficulties in achieving realistic control performance goals when implementing these algorithms on small-scale systems, we further analyzed step by step what hinders their performance and why. Our observations are summarized in the following.

*Challenges Facing the CT-RL Optimal Control Problem:* To the credit of the existing CT-RL algorithms, the CT-RL optimal control problem is considerably more difficult than its DT-RL counterpart. Altogether, combating the multitude of structural challenges in the continuous state, action, and time, and alongside the usual dimensionality, approximation, and PE issues, while rigorously proving convergence, closed-loop stability, and other control-centric performance guarantees, proves to be a three-pronged challenge perhaps unparalleled by any other problem in control systems.

### A. Design Challenges and Performance Limitations Facing Current CT-RL Algorithms

*1) Systematically Achieving PE Proves Difficult:* As noted in Remark 8, there does not exist a systematic way to ensure the PE condition for nonlinear systems. The evaluations conducted here reaffirm the severity of this challenge. Indeed, as manifested empirically by the conditioning data seen in Table VII, collecting quality state trajectory data proves difficult, even for low-order systems and bases. The challenge becomes especially acute for open-loop unstable systems, where the designer must balance excitation within the confines of the disturbance rejection capabilities of the controller. Here, we note that SPI, which tunes its weights via gradient descent and, hence, does not face conditioning concerns, still exhibits significant PE issues, and its weights either freeze due to insufficient excitation or fail to stabilize when the excitation is increased.

*2) Underlying Complexity of Existing Algorithms Causes Performance Limitations:* This work showcases the promising theoretical guarantees offered by existing CT-RL algorithms. However, significant algorithm complexity is required in order to prove these guarantees, resulting in numerical problems (e.g., CT-VI's nested pseudoinversion/integration tuning structure). Indeed, we pose conditioning issues as central—and intrinsic—to these algorithms and their performance shortcomings. In reality, the overly complex nature of these algorithms makes them intractable.

*3) Large Number of Hyperparameters Hinders Systematic Design:* Another side effect of the observed algorithm complexity is the large number of hyperparameters required by each. For example, SPI requires the designer to choose the learning time $t_f$, probing noise $e$, tuning parameters $\alpha_1, \alpha_2 > 0$, $F_1 > 0$, $F_2 = F_2^T > 0$, and initial weights $c_0, w_0$—finding a selection that yields convergent weights is a challenge in and of itself. We attempt to systematically select hyperparameters in Section VIII and justify our rationale (e.g., selecting smaller learning time $t_f$ for IRL to avoid data quality degradation and choosing a larger learning time $t_f$ for SPI to allow its gradient-descent tuning laws to converge), but, as we encounter performance issues, these efforts inevitably give way to haphazard algorithm-specific troubleshooting. Ultimately, not being able to systematically select hyperparameters to achieve good performance for these design algorithms defeats the purpose of their theoretical guarantees.

*4) Dimensional Scalability Issues Limit Real-World Applicability:* Bellman's curse of dimensionality has long explained scalability issues, but these algorithms exhibit severe numerical breakdowns to even small increments in problem dimension (e.g., the addition of one basis function to the critic). Each eventually experiences weight convergence issues and resultantly large approximation errors. Solutions are found to be highly sensitive to initial conditions, signaling difficulty for generalizability.

### B. Directions of Future Research

The limitations of CT-RL algorithms illustrated by this work motivate several potentially fruitful and compelling directions for future research.

*1) Leveraging Established Classical Results:* CT-RL is at the very early stages of development. Practically useful RL design methods validated by systematic performance evaluations are needed. To this end, in the near future, RL algorithm development may benefit from adapting/incorporating classical and model-based architectural features. Such innovations will allow RL algorithm designers to draw from well-established and practically tested classical theory to provide much-needed insights on RL controller synthesis and shed light on performance guarantees/limitations. Conducting transparent "apples-to-apples" performance comparisons with classical techniques is necessary to formalize CT-RL as a control method.

*2) Taking Advantage of Modeling or Models:* By virtue of capturing the interacting dynamics between the agent and the environment, a well-developed model may allow the learning controller to more efficiently explore the state space and, thereby, improve value function approximation. Such models can be obtained from an effective and efficient system identification process or from a first-principles physical model of the environment such as a kinematic model in the case of mechanical/robotic systems.

Modeling may bring several additional benefits: 1) modeling may reduce the learning controller's demand for training data and/or reduce the stringency of PE requirements, thereby mitigating data deficiencies that commonly arise and hinder learning control performance; 2) incorporating a well-defined model structure directly in the algorithm design may alleviate the numerical complexity issues illustrated in this work to some extent; 3) an off-line controller may be designed first to be used before the online learning controller adapts to the specific task needs; and 4) the learning controller may take advantage of system dynamics by rolling-out or planning ahead in solving certain sequential decision and control problems.

Yet, it is important to note that a poorly constructed model may defeat these purposes and may even introduce additional adverse effects to the resulting controller. Unfortunately, few systematic evaluations have been conducted on the effect of modeling errors due to approximation by an NN or other modeling methods. Instead, often, the existing neural-network dynamical approximation works simply assume that the approximation error is within an $\epsilon$ bound, after which control results are obtained by illustrating small,

handcrafted examples. Oftentimes, these works use simple NNs with radial basis functions, which have rarely been associated with the most recent successes of NN applications. Systematic approaches and evaluations are called for to examine the tradeoffs between the advantages of incorporating a model versus the adversities induced by inevitable model inaccuracy. These issues are out of the scope of this work, as few results have been developed to directly account for them in a realistic problem-solving context.

*3) Exploring Nonlinear Network Structures for Improved Approximation/Scaling Performance:* Each of the four methods studied here requires a linearly independent basis, which is by comparison a strong requirement, since almost none of the successful demonstrations of RL control relies on linearly independent bases. Furthermore, CT-RL methods for control almost universally employ single-layer, linear NN approximation structures (such as polynomial basis functions), which again is not representative in comparison to the approximation capabilities of deep networks. Future works that relax such assumptions and take advantage of deep networks could perhaps improve the dimensional scaling and approximation issues exhibited by current CT-RL algorithms. Along this vein, a fruitful future area of study may try to explain why computation-based methods, such as (deep) NNs, are effective, at least in case studies and benchmark problems.

*4) Performing Systematic Comparative Studies of CT-RL and DT-RL Algorithms:* As illustrated in Section II, DT-RL methods have achieved great successes in a variety of controls applications. Having now examined their CT counterparts, a future study that delves into their inherent differences could perhaps shed light on the CT/DT gap and, thereby, uncover new insights for future CT-RL algorithm development. There is a need to investigate at the fundamental level what causes a loss of learning efficiency in CT-RL methods and how to flexibly collect data, reuse data, and remove various theoretical constraints posed as assumptions in developing the major control results.

### C. Future Applications Prospects

These four CT-RL works are instrumental and have inspired an ever-increasing number of follow-up results. Unfortunately, even the most recent follow-up publications usually only present incremental improvements, or they use similar techniques on a slightly varied control problem. As such, they fail to substantiate the proofs with systematic, not to mention realistic, design evaluations. The essential limitations discussed in this study still exist. New practices and innovative approaches to the analysis and synthesis of CT-RL algorithms are much needed.

To this end, we call on future algorithm works to go beyond proving similar theoretical results by substantiating them with systematic numerical studies. Though perhaps beyond the current scope of the developed methods, we should, in the future, consider benchmarking on well-motivated, realistic problems and providing comparisons to other potential solutions. We hope that the novel quantitative analytical framework developed here may be of reference for future studies.

## REFERENCES


[1] R. E. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton Univ. Press, 1957.

[2] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont, MA, USA: Athena Scientific, 2005.

[3] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*, 3rd ed. Hoboken, NJ, USA: Wiley, 2012.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[5] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY, USA: Wiley, 1994.

[6] J. Si, A. G. Barto, W. B. Powell, and D. C. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*. Piscataway, NJ, USA: Wiley, 2004.

[7] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 5, pp. 834–846, Sep./Oct. 1983.

[8] R. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA, USA: MIT Press, 1960.

[9] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA, USA: Athena Scientific, 1996.

[10] F.-Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 39–47, May 2009.

[11] P. J. Werbos, "Neural networks for control and system identification," in *Proc. 28th IEEE Conf. Decis. Control*, Dec. 1989, pp. 260–265.

[12] P. J. Webros, *A Menu of Designs for Reinforcement Learning Over Time*. Cambridge, MA, USA: MIT Press, 1991.

[13] P. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. New York, NY, USA: Van Nostrand, 1992.

[14] A. E. Bryson and W. F. Denham, "A steepest-ascent method for solving optimum programming problems," *J. Appl. Mech.*, vol. 29, no. 2, pp. 247–257, Jun. 1962.

[15] J. Huang and C.-F. Lin, "Numerical approach to computing nonlinear $H_\infty$ control laws," *J. Guid., Control, Dyn.*, vol. 18, pp. 989–994, May 1995.

[16] H. J. Kushner, "Numerical methods for stochastic control problems in continuous time," *SIMA J. Control Optim.*, vol. 28, no. 5, pp. 999–1048, 1990.

[17] R. W. Beard, "Improving the closed-loop performance of nonlinear systems," Ph.D. thesis, Dept. Elect. Eng., Rensselaer Polytech. Inst., Troy, NY, USA, Oct. 1995.

[18] R. W. Beard, G. N. Saridis, and J. T. Wen, "Galerkin approximations of the generalized Hamilton–Jacobi-Bellman equation," *Automatica*, vol. 33, no. 12, pp. 2159–2177, Dec. 1997.

[19] D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Netw.*, vol. 22, no. 3, pp. 237–246, 2009.

[20] K. G. Vamvoudakis and F. L. Lewis, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878–888, May 2010.

[21] Y. Jiang and Z.-P. Jiang, "Robust adaptive dynamic programming and feedback stabilization of nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 882–893, May 2014.

[22] T. Bian and Z.-P. Jiang, "Reinforcement learning and adaptive optimal control for continuous-time nonlinear systems: A value iteration approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 7, pp. 2781–2790, Jul. 2022.

[23] 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, Dec. 2012.

[24] 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, Jun. 2018.

[25] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 2, no. 1, pp. 253–279, 2019.

[26] D. Wang, D. Liu, H. Li, B. Luo, and H. Ma, "An approximate optimal control approach for robust stabilization of a class of discrete-time nonlinear systems with uncertainties," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 5, pp. 713–717, May 2016.

[27] P. He and S. Jagannathan, "Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 37, no. 2, pp. 425–436, Apr. 2007.

[28] P. Zhang, Y. Yuan, and L. Guo, "Fault-tolerant optimal control for discrete-time nonlinear system subjected to input saturation: A dynamic event-triggered approach," *IEEE Trans. Cybern.*, vol. 51, no. 6, pp. 2956–2968, Jun. 2021.

[29] C. Mu, D. Wang, and H. He, "Novel iterative neural dynamic programming for data-based approximate optimal control design," *Automatica*, vol. 81, pp. 240–252, Jul. 2017.

[30] 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.

[31] Q. Wei, D. Liu, Q. Lin, and R. Song, "Discrete-time optimal control via local policy iteration adaptive dynamic programming," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3367–3379, Oct. 2017.

[32] 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, Jul. 2018.

[33] D. Liu, Q. Wei, and P. Yan, "Generalized policy iteration adaptive dynamic programming for discrete-time nonlinear systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 12, pp. 1577–1591, Dec. 2015.

[34] X. Gao, J. Si, Y. Wen, M. Li, and H. Huang, "Reinforcement learning control of robotic knee with human-in-the-loop by flexible policy iteration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5873–5887, Oct. 2022.

[35] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 38, no. 4, pp. 943–949, Jun. 2008.

[36] D. Wang, D. Liu, Q. Wei, D. Zhao, and N. Jin, "Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming," *Automatica*, vol. 48, no. 8, pp. 1825–1832, 2012.

[37] F. Liu, J. Sun, J. Si, W. Guo, and S. Mei, "A boundedness result for the direct heuristic dynamic programming," *Neural Netw.*, vol. 32, pp. 229–235, Aug. 2012.

[38] D. Liu and Q. Wei, "Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 779–789, Apr. 2013.

[39] Q. Wei, F.-Y. Wang, D. Liu, and X. Yang, "Finite-approximation-error-based discrete-time iterative adaptive dynamic programming," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2820–2833, Dec. 2014.

[40] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264–276, Mar. 2001.

[41] R. Hafner and M. Riedmiller, "Reinforcement learning in feedback control: Challenges and benchmarks from technical process control," *Mach. Learn.*, vol. 84, nos. 1–2, pp. 137–169, Jul. 2011.

[42] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. 31st Int. Conf. Mach. Learn.*, Jan. 2014, pp. 387–395.

[43] R. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[44] Y. Hou, L. Liu, Q. Wei, X. Xu, and C. Chen, "A novel DDPG method with prioritized experience replay," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 316–321.

[45] V. Minh et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[46] 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, 2016.

[47] 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.

[48] D. Silver et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[49] F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in *Proc. 22nd Euromicro Int. Conf. Parallel, Distrib., Netw.-Based Process.*, Feb. 2014, pp. 500–507.

[50] K. Mondal, A. A. Rodriguez, S. S. Manne, N. Das, and B. A. Wallace, "Comparison of kinematic and dynamic model based linear model predictive control of non-holonomic robot for trajectory tracking: Critical trade-offs addressed," in *Proc. IASTED Int. Conf. Mechatronics Control*, Dec. 2019, pp. 9–17.

[51] K. Mondal, B. A. Wallace, and A. A. Rodriguez, "Stability versus maneuverability of non-holonomic differential drive mobile robot: Focus on aggressive position control applications," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Aug. 2020, pp. 388–395.

[52] C. Lu, J. Si, and X. Xie, "Direct heuristic dynamic programming for damping oscillations in a large power system," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 38, no. 4, pp. 1008–1013, Aug. 2008.

[53] W. Guo, F. Liu, J. Si, D. He, R. Harley, and S. Mei, "Approximate dynamic programming based supplementary reactive power control for DFIG wind farm to enhance power system stability," *Neurocomputing*, vol. 170, pp. 417–427, Dec. 2015.

[54] W. Guo, F. Liu, J. Si, D. He, R. Harley, and S. Mei, "Online supplementary ADP learning controller design and application to power system frequency control with large-scale wind energy integration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 8, pp. 1748–1761, Aug. 2016.

[55] Q. Wei and D. Liu, "Data-driven neuro-optimal temperature control of water–gas shift reaction using stable iterative adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6399–6408, Nov. 2014.

[56] Y. Jiang, J. Fan, T. Chai, and F. L. Lewis, "Dual-rate operational optimal control for flotation industrial process with unknown operational model," *IEEE Trans. Ind. Electron.*, vol. 66, no. 6, pp. 4587–4599, Jun. 2019.

[57] R. Enns and J. Si, "Apache helicopter stabilization using neural dynamic programming," *J. Guid., Control, Dyn.*, vol. 25, no. 1, pp. 19–25, 2002.

[58] R. Enns and J. Si, "Helicopter trimming and tracking control using direct neural dynamic programming," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 929–939, Aug. 2003.

[59] R. Enns and J. Si, "Helicopter flight-control reconfiguration for main rotor actuator failures," *J. Guid., Control, Dyn.*, vol. 26, no. 4, pp. 572–584, 2003.

[60] Q. Yang, W. Cao, W. Meng, and J. Si, "Reinforcement-learning-based tracking control of waste water treatment process under realistic system conditions and control performance requirements," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 8, pp. 5284–5294, Aug. 2022.

[61] Y. Wen, M. Liu, J. Si, and H. Huang, "Adaptive control of powered transfemoral prostheses based on adaptive dynamic programming," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Aug. 2016, pp. 500–507.

[62] Y. Wen, J. Si, X. Gao, S. Huang, and H. H. Huang, "A new powered lower limb prosthesis control framework based on adaptive dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 2215–2220, Sep. 2017.

[63] Y. Wen, J. Si, A. Brandt, X. Gao, and H. Huang, "Online reinforcement learning control for the personalization of a robotic knee prosthesis," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2346–2356, Jun. 2019.

[64] R. Wu, M. Li, Z. Yao, W. Liu, J. Si, and H. Huang, "Reinforcement learning impedance control of a robotic prosthesis to coordinate with human intact knee motion," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7014–7020, Jul. 2022.

[65] M. Li, Y. Wen, X. Gao, J. Si, and H. Huang, "Toward expedited impedance tuning of a robotic prosthesis for personalized gait assistance by reinforcement learning control," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 407–420, Feb. 2022.

[66] R. Wu, J. Zhong, B. A. Wallace, X. Gao, H. Huang, and J. Si, "Human-robotic prosthesis as collaborating agents for symmetrical walking," in *NeurIPS*, vol. 36, Nov. 2022, pp. 1–15.

[67] Y. Zhu and D. Zhao, "Comprehensive comparison of online ADP algorithms for continuous-time optimal control," *Artif. Intell. Rev.*, vol. 49, no. 4, pp. 531–547, 2017.

[68] D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei, "Adaptive dynamic programming for control: A survey and recent advances," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 142–160, Jan. 2021.

[69] 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.

[70] B. Song, J.-J. Slotine, and Q.-C. Pham, "Stability guarantees for continuous RL control," 2022, *arXiv:2209.07324*.

[71] J. Kim, J. Shin, and I. Yang, "Hamilton–Jacobi deep Q-learning for deterministic continuous-time systems with Lipschitz continuous controls," *J. Mach. Learn. Res.*, vol. 22, no. 1, pp. 9363–9396, Sep. 2021.

[72] J. Lee and R. S. Sutton, "Policy iterations for reinforcement learning problems in continuous time and space—Fundamental theory and methods," *Automatica*, vol. 126, Apr. 2021, Art. no. 109421.

[73] M. Lutter, B. Belousov, S. Mannor, D. Fox, A. Garg, and J. Peters, "Continuous-time fitted value iteration for robust policies," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Oct. 19, 2022, doi: 10.1109/TPAMI.2022.3215769.

[74] C. Yildiz, M. Heinonen, and H. Lähdesmäki, "Continuous-time model-based reinforcement learning," in *Proc. 38th Int. Conf. Mach. Learn.*, Jul. 2021, pp. 12009–12018.

[75] Y. Jiang, J. Fan, T. Chai, J. Li, and F. L. Lewis, "Data-driven flotation industrial process operational optimal control based on reinforcement learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 1974–1989, May 2018.

[76] Y. Li, K. Sun, and S. Tong, "Observer-based adaptive fuzzy fault-tolerant optimal control for SISO nonlinear systems," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 649–661, Feb. 2019.

[77] K. G. Vamvoudakis, D. Vrabie, and F. L. Lewis, "Online adaptive algorithm for optimal control with integral reinforcement learning," *Int. J. Robust Nonlinear Control*, vol. 24, no. 17, pp. 2686–2710, Nov. 2014.

[78] H. Zhang, L. Cui, X. Zhang, and Y. Luo, "Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2226–2236, Dec. 2011.

[79] X. Yang, D. Liu, and D. Wang, "Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints," *Int. J. Control*, vol. 87, no. 3, pp. 553–566, Mar. 2014.

[80] D. Liu, X. Yang, and H. Li, "Adaptive optimal control for a class of continuous-time affine nonlinear systems with unknown internal dynamics," *Neural Comput. Appl.*, vol. 23, nos. 7–8, pp. 1843–1850, Dec. 2013.

[81] H. Liu, W. Zhao, F. L. Lewis, Z.-P. Jiang, and H. Modares, "Attitude synchronization for multiple quadrotors using reinforcement learning," in *Proc. Chin. Control Conf. (CCC)*, Jul. 2019, pp. 2480–2483.

[82] L. Cui et al., "Learning-based balance control of wheel-legged robots," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7667–7674, Oct. 2021.

[83] Y. Jiang and Z.-P. Jiang, "Robust adaptive dynamic programming with an application to power systems," *IEEE Trans. Neural Netw.*, vol. 24, no. 7, pp. 1150–1156, Jul. 2013.

[84] Y. Jiang and Z. P. Jiang, "Adaptive dynamic programming as a theory of sensorimotor control," *Biol. Cybern.*, vol. 108, no. 4, pp. 459–473, 2014.

[85] A. A. Rodriguez, *Analysis and Design of Feedback Control Systems*. Tempe, AZ, USA: CONTROL3D, 2003.

[86] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[87] A. Isidori, *Nonlinear Control Systems: An Introduction*. Berlin, Germany: Springer-Verlag, 1985.

[88] H. W. Bode, *Network Analysis and Feedback Amplifier Design*. New York, NY, USA: D. Van Nostrand, 1945.

[89] H. Nyquist, "Regeneration theory," *Bell Syst. Tech. J.*, vol. 11, no. 1, pp. 126–147, Jan. 1932.

[90] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Upper Saddle River, NJ, USA: Prentice-Hall, 1995.

[91] F. Lin, *Robust Control Design: An Optimal Control Approach*. West Sussex, U.K.: John Wiley & Sons, 2007.

[92] Z. P. Jiang, A. R. Teel, and L. Praly, "Small-gain theorem for ISS systems and applications," *Math. Control, Signals, Syst.*, vol. 7, no. 2, pp. 95–120, 1994.

[93] S. J. Bradtke, B. E. Ydstie, and A. G. Barto, "Adaptive linear quadratic control using policy iteration," in *Proc. Amer. Control Conf. (ACC)*, vol. 3, Jun./Jul. 1994, pp. 3475–3479.

[94] T. Liu and Z. P. Jiang, "A small-gain approach to robust event-triggered control of nonlinear systems," *IEEE Trans. Autom. Control*, vol. 60, no. 8, pp. 2072–2085, Aug. 2015.

[95] X. Zhong and H. He, "An event-triggered ADP control approach for continuous-time system with unknown internal states," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 683–694, Mar. 2017.

[96] Q. Zhao, J. Si, and J. Sun, "Online reinforcement learning control by direct heuristic dynamic programming: From time-driven to event-driven," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 8, pp. 4139–4144, Aug. 2022.

[97] M. Abu-Khalaf, F. L. Lewis, and J. Huang, "Neurodynamic programming and zero-sum games for constrained control systems," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1243–1252, Jul. 2008.

[98] K. G. Vamvoudakis and F. L. Lewis, "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton–Jacobi equations," *Automatica*, vol. 47, no. 8, pp. 1556–1569, Aug. 2011.

[99] A. Odekunle, W. Gao, M. Davari, and Z.-P. Jiang, "Reinforcement learning and non-zero-sum game output regulation for multi-player linear uncertain systems," *Automatica*, vol. 12, Feb. 2020, Art. no. 108672.

[100] C. Chen, L. Xie, K. Xie, F. L. Lewis, and S. Xie, "Adaptive optimal output tracking of continuous-time systems via output-feedback-based reinforcement learning," *Automatica*, vol. 146, Dec. 2022, Art. no. 110581.

[101] H. Modares, F. L. Lewis, and Z.-P. Jiang, "$H_\infty$ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2550–2562, Oct. 2015.

[102] *Software Related to Funded Research of F.L. Lewis*. Accessed: Apr. 8, 2022. [Online]. Available: https://lewisgroup.uta.edu/code/Software%20from%20Research.htm

[103] K. Vamvoudakis, D. Vrabie, and F. Lewis, "Online policy iteration based algorithms to solve the continuous-time infinite horizon optimal control problem," in *Proc. IEEE Symp. Adapt. Dyn. Program. Reinforcement Learn.*, Mar. 2009, pp. 36–41.

[104] V. Lakshmikantham, S. Leela, and A. A. Martynyuk, *Practical Stability of Nonlinear Systems*. Singapore: World Scientific, 1990.

[105] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed. Philadelphia, PA, USA: SIAM, 2002.

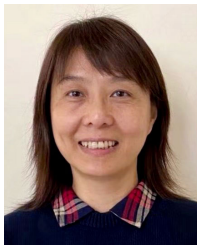
[106] *TNNLS 2022—CT-RL Optimal Control*. Accessed: Dec. 1, 2022. [Online]. Available: https://github.com/bawalla2/TNNLS-2022–CT-RL-Optimal-Control.git

[107] K. Ogata, *Modern Control Engineering*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1997.

**Brent A. Wallace** received the B.S. degree from Arizona State University, Tempe, AZ, USA, in 2019, where he is currently pursuing the Ph.D. degree with the School of Electrical, Computer and Energy Engineering.

He was a Research Intern with The Aerospace Corporation Microelectronics Group, El Segundo, CA, USA. His research interests include adaptive dynamic programming, nonlinear optimal control, and control applications in aerospace systems.

Mr. Wallace received the NSF Graduate Research Fellowship in 2020.



**Jennie Si** (Fellow, IEEE) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 1985 and 1988, respectively, and the Ph.D. degree from the University of Notre Dame, Notre Dame, IN, USA, in 1992.

She consulted for Intel, Arizona Public Service, and Medtronic, all in Phoenix, AZ, USA. She has been a Faculty Member with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA, since 1991. Her research focuses on reinforcement learning control utilizing tools from optimal control theory, reinforcement learning, and neural networks. Her recent work also involves optimal adaptive control of wearable robots.

Dr. Si was a recipient of the NSF/White House Presidential Faculty Fellow Award in 1995 and the Motorola Engineering Excellence Award in 1995. She is a Distinguished Lecturer of the IEEE Computational Intelligence Society. She has served on several professional organizations' executive boards and international conference committees. She was an Advisor to the NSF Social Behavioral and Economical Directory. She has served on several proposal review panels. She was an Associate Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, the IEEE TRANSACTIONS ON SEMICONDUCTOR MANUFACTURING, and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and an Action Editor of *Neural Networks*. She is a Senior Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.