ROBUST ADAPTIVE DYNAMIC PROGRAMMING FOR CONTINUOUS-TIME LINEAR AND NONLINEAR SYSTEMS

DISSERTATION

Submitted in Partial Fulfillment of

The Requirements for

the Degree of

DOCTOR OF PHYLOSOPHY

(Electrical Engineering)

at the

NEW YORK UNIVERSITY
POLYTECHNIC SCHOOL OF ENGINEERING

by

Yu Jiang

May 2014

Approved:

_____
Department Head Signature

_____
Date

Copy No. #_____

Students ID# N12872239

Approved by the Guidance Committee

Major: Electrical Engineering

_____

Zhong-Ping Jiang
Professor of Electrical and Computer Engineering

_____

Date

_____

Francisco de León
Associate Professor of Electrical and Computer Engineering

_____

Date

_____

Peter Voltz
Associate Professor of Electrical and Computer Engineering

_____

Date

Minor: Mathematics

_____

Gaoyong Zhang
Professor of Mathematics

_____

Date

Vita

Yu Jiang was born in Xi'an, China, in 1984. He obtained the B.Sc. degree in Applied Mathematics from Sun Yat-sen University, Guangzhou, China, in 2006, and the M.Sc. degree in automation science and engineering from South China University of Technology, Guangzhou, China, in 2009. He won the National First Grade Award in the 2005 Chinese Undergraduate Mathematical Contest in Modeling.

Currently, he is a fifth-year Ph.D. candidate working in the Control and Networks (CAN) Lab at Polytechnic School of Engineering, New York University, under the guidance of Professor Zhong-Ping Jiang. His research interests include robust adaptive dynamic programming and its applications in engineering and biological systems. In summer 2013, he interned at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA. He received the Shimemura Young Author Prize (with Zhong-Ping Jiang) at the 9th Asian Control Conference, Istanbul, Turkey, 2013.

List of publications

Book

1. **Yu Jiang** and Zhong-Ping Jiang, “Robust Adaptive Dynamic Programming”, in preparation.

Book Chapter

1. **Yu Jiang** and Zhong-Ping Jiang, "Robust adaptive dynamic programming", in Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, F. L. Lewis and D. Liu, Eds, John Wiley and Sons, 2012.

Journal Papers (Under review)

1. Tao Bian, **Yu Jiang** and Zhong-Ping, “Adaptive dynamic programming for stochastic systems with state and control dependent noise”, IEEE Transactions on Automatic Control, submitted, April 2014.
2. **Yu Jiang** and Zhong-Ping Jiang, “Global adaptive dynamic programming for continuous-time nonlinear systems,” IEEE Transactions on Automatic Control, submitted, Dec 2013.
3. **Yu Jiang**, Yebin Wang, Scott Bortoff, and Zhong-Ping Jiang, “Optimal Co-Design of Nonlinear Control Systems Based on A Modified Policy Iteration Method,” IEEE Transactions on Neural Networks and Learning Systems, major revision, Dec 2013.
4. **Yu Jiang** and Zhong-Ping Jiang, “A robust adaptive dynamic programming principle for sensorimotor control with signal-dependent noise,” Journal of Systems Science and Complexity, revised, Mar 2014.
5. Tao Bian, **Yu Jiang**, and Zhong-Ping Jiang, “Decentralized and adaptive optimal control of large-scale systems with application to power systems”, IEEE Transactions on Industrial Electronics, revised, Apr 2014.
6. **Yu Jiang** and Zhong-Ping Jiang, “Adaptive dynamic programming as a theory of sensorimotor control,” Biological Cybernetics, revised Mar 2014.

Journal Papers (Published or accepted)

1. **Yu Jiang**, Yebin Wang, Scott Bortoff, and Zhong-Ping Jiang, “An Iterative Approach to the Optimal Co-Design of Linear Control System,” Automatica, provisionally accepted.
2. Tao Bian, **Yu Jiang**, and Zhong-Ping Jiang, “Adaptive dynamic programming and optimal control of nonlinear nonaffine systems,” Automatica, provisionally accepted.

3. **Yu Jiang** and Zhong-Ping Jiang, “Robust adaptive dynamic programming and feedback stabilization of Nonlinear Systems,” IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 5, pp. 882-893, May 2014.
4. Zhong-Ping Jiang and **Yu Jiang**, “Robust adaptive dynamic programming for linear and nonlinear systems: An overview”, European Journal of Control, vol. 19, no. 5, pp. 417-425, 2013.
5. **Yu Jiang** and Zhong-Ping Jiang, "Robust adaptive dynamic programming with an application to power systems", IEEE Transactions on Neural Networks and Learning Systems, vol. 24, no.7, pp. 1150- 1156, 2013.
6. **Yu Jiang** and Zhong-Ping Jiang, "Robust adaptive dynamic programming for large-scale systems with an application to multimachine power systems," IEEE Transactions on Circuits and Systems, Part II, vol. 59, no. 10, pp. 693-697, 2012.
7. Ning Qian, **Yu Jiang**, Zhong-Ping Jiang, and Pietro Mazzoni, "Movement duration, Fitts's law, and an infinite-horizon optimal feedback control model for biological motor systems", Neural Computation, vol. 25, no. 3, pp. 697-724, 2012.
8. **Yu Jiang** and Zhong-Ping Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown system dynamics", Automatica, vol. 48, no. 10, pp. 2699-2704, Oct. 2012.
9. **Yu Jiang** and Zhong-Ping Jiang, "Approximate dynamic programming for optimal stationary control with control-dependent noise," IEEE Transactions on Neural Networks, vol. 22, no.12, 2392-2398, 2011.

Conference Papers

1. **Yu Jiang** and Zhong-Ping Jiang, “Global adaptive dynamic programming and global optimal control for a class of nonlinear systems”, accepted in the 2014 IFAC World Congress.
2. **Yu Jiang**, Zhong-Ping Jiang, “Robust adaptive dynamic programming for sensorimotor control with signal-dependent noise,” in Proceedings of the 2013 IEEE Signal Processing in Medicine and Biology Symposium, Brooklyn, NY, 2013.
3. Zhong-Ping Jiang and **Yu Jiang**, “Robust Adaptive Dynamic Programming: Recent results and applications”, in Proceedings of the 32nd Chinese Control Conference, Xi'An, China, pp. 968-973, 2013.
4. **Yu Jiang** and Zhong-Ping Jiang, “Robust adaptive dynamic programming for optimal nonlinear control,” in proceedings of the 9th Asian Control Conference. **(Shimemura Young Author Award)**.

5. Zhong-Ping Jiang and **Yu Jiang**, “A new approach to robust and optimal nonlinear control design,” the Third IASTED Asian Conference on Modeling, Identification and Control, Phuket, Thailand, 2013.
6. **Yu Jiang** and Zhong-Ping Jiang, "Adaptive dynamic programming as a theory of motor control", accepted in the 2012 IEEE Signal Processing in Medicine and Biology Symposium, New York, NY, 2012.
7. **Yu Jiang** and Zhong-Ping Jiang, "Robust adaptive dynamic programming for nonlinear control design," accepted in the51st IEEE Conference on Decision and Control, Dec. 2012, Maui, Hawaii, USA.
8. **Yu Jiang** and Zhong-Ping Jiang, "Computational adaptive optimal control with an application to blood glucose regulation in type 1 diabetics," in Proceedings of the 31th Chinese Control Conference, Hefei, China, pp. 2938-2943, July, 2012.
9. **Yu Jiang** and Zhong-Ping Jiang, "Robust adaptive dynamic programming: An overview of recent results", in Proceedings of the 20th International Symposium on Mathematical Theory of Networks and Systems, Melbourne, Australia, 2012.
10. **Yu Jiang** and Zhong-Ping Jiang, "Robust approximate dynamic programming and global stabilization with nonlinear dynamic uncertainties," in Proceedings of the Joint IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, pp. 115-120, 2011.
11. **Yu Jiang**, Srinivasa Chemudupati, Jan Morup Jorgensen, Zhong-Ping Jiang, and Charles S. Peskin, "Optimal control mechanism involving the human kidney," in Proceedings of the Joint IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, pp. 3688-3693, 2011.
12. **Yu Jiang** and Zhong-Ping Jiang, "Approximate dynamic programming for stochastic systems with additive and multiplicative noise," in Proceedings of the IEEE Multi-Conference on Systems and Control, pp. 185-190, Denver, CO, 2011.
13. **Yu Jiang**, Zhong-Ping Jiang, and Ning Qian, "Optimal control mechanisms in human arm reaching movements," in Proceedings of the 30th Chinese Control Conference, pp. 1377-1382, Yantai, China, 2011.
14. **Yu Jiang** and Zhong-Ping Jiang, "Approximate dynamic programming for output feedback control," in Proceedings of Chinese Control Conference, pp. 5815-5820, Beijing, China, 2010.
15. **Yu Jiang** and Jie Huang, "Output regulation for a class of weakly minimum phase systems and its application to a nonlinear benchmark system," in Proceedings of American Control Conference, pp. 5321-5326, St. Louis, USA, 2009.

Acknowledgement

I would first and foremost like to thank Prof. Zhong-Ping Jiang. Without his generous support and guidance, this dissertation would not have been possible and it would remain incomplete if I did not thank him with utmost sincerity. In the past five years, he not only introduced me to the exciting topic of robust adaptive dynamic programming, but also set high goals for my research and at the same time keeps helping and encouraging me to work towards them. He gives me flexibility to work on any problem that interests me, makes me question theories which I took for granted, and challenges me to seek breakthroughs. Indeed, it was a great honor and pleasure working under his guidance.

I would like to thank Prof. Charles Peskin at Courant Institute of Mathematical Sciences (CIMS) and Prof. Ning Qian from Columbia University for introducing me to the wonderful subjects of biological related control problems.

I would like to thank Prof. Francisco de León for providing a lot of constructive suggestions and professional advice when I was trying to apply my theory to solve power systems related control problems.

I would also like to thank Dr. Yebin Wang at Mitsubishi Electric Research Laboratories (MERL) for offering me a chance to intern at the prestigious industrial research lab and to learn how to apply control theories to practical engineering systems.

I would like to extend my heartfelt thanks to Prof. Peter Voltz, Prof. Gaoyong Zhang, and Prof. Francisco de León for taking their valuable time reading and reviewing my dissertation.

I would like to thank Srinivasa Chemudupati, Ritchy Laurent, Zhi Chen, Po-Chen Chen, Xiyun Wang, Xinhe Chen, Qingcheng Zhu, Yang Xian, Xuesong Lu, Siddhartha Srikantham, Tao Bian, Weinan Gao, Bei Sun, Jeffery Pawlick and all my current and former fellow lab mates for creating a supportive, productive, and fun environment.

Last but not least, I thank the National Science Foundation for supporting my research work.

To Misi – "a prudent wife is from the Lord"

Abstract

Robust Adaptive Dynamic Programming for Continuous-Time Linear and Nonlinear Systems

By

Yu Jiang

Advisor: Zhong-Ping Jiang

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy (Electrical Engineering)

May 2014

The field of adaptive dynamic programming and its applications to control engineering problems has undergone rapid progress over the past few years. Recently, a new theory called Robust Adaptive Dynamic Programming (for short, RADP) has been developed for the design of robust optimal controllers for linear and nonlinear systems subject to both parametric and dynamic uncertainties. This dissertation integrates our recent contributions to the development of the theory of RADP and illustrates its potential applications in both engineering and biological systems.

In order to develop the RADP framework, our attention is first focused on the development of an ADP-based online learning method for continuous-time (CT) linear systems with completely unknown system. This problem is challenging due to the different structures between CT and discrete-time (DT) algebraic Riccati equations (AREs), and therefore methods developed for DT ADP cannot be directly applied in the CT setting. This obstacle is overcome in our work by taking advantages of exploration noise. The methodology is

immediately extended to deal with CT affine nonlinear systems, via neural-networks-based approximation of the Hamilton-Jacobi-Bellman (HJB) equation, of which the solution is extremely difficult to be obtained analytically. To achieve global stabilization, for the first time we propose an idea of global ADP (or GADP), in which we relax the problem of solving the Hamilton-Jacobi-Bellman (HJB) equation to an optimization problem, of which a suboptimal solution is obtained via a sum-of-squares-program-based policy iteration method. The resultant control policy is globally stabilizing, instead of semi-globally or locally stabilizing.

Then, we develop RADP aimed at computing globally stabilizing and suboptimal control policies in the presence of dynamic uncertainties. A key strategy is to integrate ADP theory with techniques in modern nonlinear control with a unique objective of filling a gap in the past literature of ADP without taking into account dynamic uncertainties. The development of this framework contains two major steps. First, we study an RADP method for partially linear systems (i.e., linear systems with nonlinear dynamic uncertainties) and weakly nonlinear large-scale systems. Global stabilization of the systems can be achieved by selecting performance indices with appropriate weights for the nominal system. Second, we extend the RADP framework for affine nonlinear systems with nonlinear dynamic uncertainties. To achieve robust stabilization, we resort to tools from nonlinear control theory, such as gain assignment and the ISS nonlinear small-gain theorem.

From the perspective of RADP, we derive a novel computational mechanism for sensorimotor control. Sharing some essential features of reinforcement learning, which was originally observed from mammals, the RADP model for sensorimotor control suggests that, instead of identifying the system dynamics of both the motor system and the environment, the central nervous system (CNS) computes iteratively a robust optimal control policy using the real-time sensory data. By comparing our numerical results with

experimentally observed data, we show that the proposed model can reproduce movement trajectories which are consistent with experimental observations. In addition, the RADP theory provides a unified framework that connects optimality and robustness properties in the sensorimotor system. Therefore, we argue that the CNS may use RADP-like learning strategies to coordinate movements and to achieve successful adaptation in the presence of static and/or dynamic uncertainties.

Contents

List of Figures

List of Tables

List of Symbols

$\mathbb{R}$	The set of real numbers.
$\mathbb{R}_+$	The set of all non-negative real numbers.
$\mathbb{Z}_+$	The set of all non-negative integers.
$\mathcal{C}^1$	The set of all continuously differentiable functions.
$\mathcal{P}$	The set of all functions in $\mathcal{C}^1$ that are also positive definite and proper.
$\|\cdot\|$	The Euclidean norm for vectors, or the induced matrix norm for matrices.
$\|\cdot\|$	For any piecewise continuous function $u : \mathbb{R}_+ \to \mathbb{R}^m$, $\|u\| = \sup\{|u(t)|, t \geq 0\}$.
$\otimes$	Kronecker product.
$\text{vec}(\cdot)$	$\text{vec}(A)$ is the mn-vector formed by stacking the columns of $A \in \mathbb{R}^{n\times m}$ on top of one another, or more precisely, starting with the first column and ending with the last column of A.
$\nu(\cdot)$	$\nu(P) = [p_{11}, 2p_{12}, \cdots, 2p_{1n}, p_{22}, 2p_{23}, \cdots, 2p_{n-1,n}, p_{nn}]^T$, $\forall P = P^T \in \mathbb{R}^{n\times n}$.
$\mu(\cdot)$	$\mu(x) = [x_1^2, x_1x_2, \cdots, x_1x_n, x_2^2, x_2x_3, \cdots, x_{n-1}x_n, x_n^2]^T$, $\forall x \in \mathbb{R}^n$.
$\|\cdot\|_R^2$	For any vector $u \in \mathbb{R}^m$ and any positive definite matrix $R \in \mathbb{R}^{m\times m}$, we define $\|u\|_R^2$ as u^TRu.
$[\cdot]_{d_1,d_2}$	For any non-negative integers d_1, d_2 satisfying $d_2 \geq d_1$, $[x]_{d_1,d_2}$ is the vector of all $\binom{n+d_2}{d_2} - \binom{n+d_1}{d_1}$ distinct monic monomials in $x \in \mathbb{R}^n$ with degree no less than d_1 and no greater than d_2, and arranged in lexicographic order [30].
$\mathbb{R}[x]_{d_1,d_2}$	The set of all polynomials in $x \in \mathbb{R}^n$ with degree no less than d_1 and no greater than d_2.
$\mathbb{R}[x]_{d_1,d_2}^m$	The set of m-dimensional vectors, of which each entry belongs to $\mathbb{R}[x]_{d_1,d_2}$.
∇	∇V refers to the gradient of a differentiable function $V : \mathbb{R}^n \to \mathbb{R}$.

List of Abbreviations

ADP	Adaptive/approximate dynamic programming
ARE	Algebraic Riccati equation
DF	Divergent force field
DP	Dynamic programming
GAS	Global asymptotical stability
HJB	Hamilton-Jacobi-Bellman (equation)
IOS	Input-to-state stability
ISS	Input-to-state stability
LQR	Linear quadratic regulator
PE	Persistent excitation
NF	Null-field
PI	Policy iteration
RADP	Robust adaptive dynamic programming
RL	Reinforcement learning
SDP	Semidefinite programming
SOS	Sum-of-squares
SUO	Strong unboundedness observability
VF	Velocity-dependent force field
VI	Value iteration

Chapter 1

Introduction

1.1 From RL to RADP

1.1.1 RL, DP, and ADP

Reinforcement learning (RL) [155] is originally observed from the learning behavior in mammals. Generally speaking, RL concerns how an agent should modify its actions to better interact with the unknown environment such that a long term goal can be achieved (see Figure 1.1). The definition of RL can be quite general. Indeed, the well-known *trial-and-error* method can be considered as one simple scheme of reinforcement learning, because trial-and-error, together with *delayed reward* [181], are two important features of RL [155]. In the seminal book by Sutton and Barto [155], the RL problem is referred to as *how to map situations to actions so as to minimize a numerical reward signal*. As an important branch in machine learning theory, RL has been brought to the computer science and control science literature as a way to study artificial intelligence in the 1960s [115, 117, 176]. Since then, numerous contributions to RL, from a control perspective, have been made (see, for example, [5, 154, 181, 102, 103, 174, 88]).

On the other hand, Dynamic programming (DP) [8] offers a theoretical way to

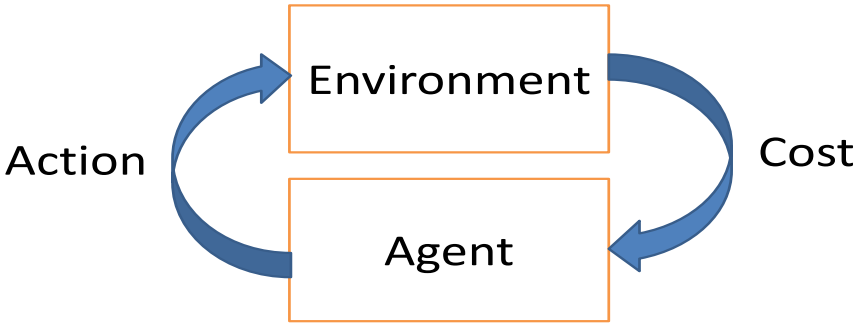

Figure 1.1: Illustration of RL. The agent gives an action to the unknown environment, and evaluates the related cost, based on which the agent can further improve the action to reduce the cost.

solve multistage decision making problems. However, it suffers from the inherent computational complexity, also known as the *curse of dimensionality* [127]. Therefore, the need for approximative methods has been recognized as early as in the late 1950s [7]. In [58], an iterative technique called policy iteration (PI) was devised by Howard for Markov decision processes. Also, Howard called the iterative method developed by Bellman [8, 7] as value iteration (VI). Computing the optimal solution through successive approximations, PI is closely related to learning methods. In 1968, Werbos pointed out that PI can be employed to perform RL [185]. Starting from then, many real-time RL methods for finding online optimal control policies have emerged and they are broadly called approximate/adaptive dynamic programming (ADP) [102, 100, 177, 186, 189, 190, 191, 193, 127, 144, 188, 202], or neurodynamic programming [10]. The main feature of ADP [186, 187] is that it employs idea from reinforcement learning [155] to achieve online approximation of the cost function, without using the knowledge of the system dynamics.

1.1.2 The development of ADP

The development of ADP theory consists of three phases. In the first phase, ADP was extensively investigated within the communities of computer science and operations research. Two basic algorithms, policy iteration [58] and value iteration [8], are usually employed. In [154], Sutton introduced the temporal difference method. In 1989, Watkins proposed the well-known Q-learning method in his PhD thesis [181]. Q-learning shares similar features with the action-dependent HDP scheme proposed by Werbos in [189]. Other related research work under a discrete time and discrete state-space Markov decision process framework can be found in [11, 10, 18, 23, 127, 130, 156, 155] and references therein. In the second phase, stability is brought into the context of ADP while real-time control problems are studied for dynamic systems. To the best of the author's knowledge, Lewis is the first who contributes to the integration of stability theory and ADP theory [102]. An essential advantage of ADP theory is that an optimal control policy can be obtained via a recursive numerical algorithm using online information without solving the HJB equation (for nonlinear systems) and the algebraic Riccati equation (ARE) (for linear systems), even when the system dynamics are not precisely known. Optimal feedback control designs for linear and nonlinear dynamic systems have been proposed by several researchers over the past few years; see, e.g., [12, 34, 118, 122, 167, 173, 196, 203]. While most of the previous work on ADP theory was devoted to discrete-time (DT) systems (see [100] and references therein), there has been relatively less research for the continuous-time (CT) counterpart. This is mainly because ADP is considerably more difficult for CT systems than for DT systems. Indeed, many results developed for DT systems [107] cannot be extended straightforwardly to CT systems. Nonetheless, early attempts were made to apply Q-learning for CT systems via discretization technique [4, 35]. However, convergence and stability analysis of these schemes are challenging. In [122], Murray *et. al* proposed an implementation method which requires the measurements

of the derivatives of the state variables. As said previously, Lewis and his co-worker proposed the first solution to stability analysis and convergence proofs for ADP-based control systems by means of LQR theory [173]. A synchronous policy iteration scheme was also presented in [166]. For CT linear systems, the partial knowledge of the system dynamics (i.e., the input matrix) must be precisely known. This restriction has been completely removed in [68]. A nonlinear variant of this method can be found in [75].

The third phase in the development of ADP theory is related to extensions of previous ADP results to nonlinear uncertain systems. Neural networks and game theory are utilized to address the presence of uncertainty and nonlinearity in control systems. See, e.g. [51, 167, 168, 203, 100, 198, 204, 183]. An implicit assumption in these papers is that the system order is known and that the uncertainty is static, not dynamic. The presence of dynamic uncertainty has not been systematically addressed in the literature of ADP. By dynamic uncertainty, we refer to the mismatch between the nominal model and the real plant when the order of the nominal model is lower than the order of the real system. A closely related topic of research is how to account for the effect of unseen variables [188]. It is quite common that the full-state information is often missing in many engineering applications and only the output measurement or partial-state measurements are available. Adaptation of the existing ADP theory to this practical scenario is important yet non-trivial. Neural networks are sought for addressing the state estimation problem [37, 87]. However, the stability analysis of the estimator/controller augmented system is by no means easy, because the total system is highly interconnected. The configuration of a standard ADP-based control system is shown in Figure 1.2.

Our recent work [67, 73, 70, 67, 69] on the development of robust ADP (for short, RADP) theory is exactly targeted at addressing these challenges.

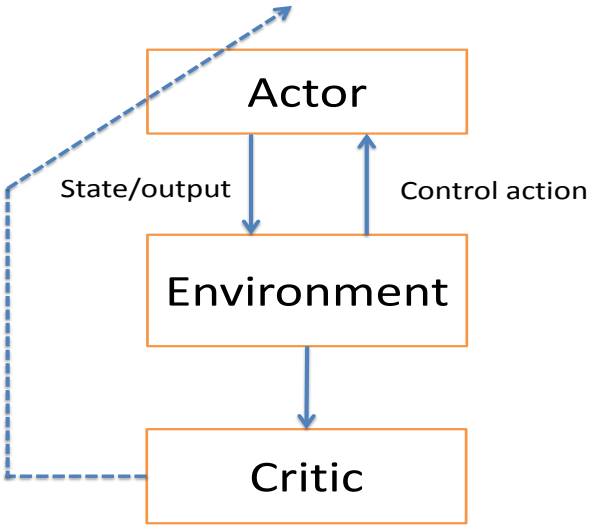

Figure 1.2: Configuration of an ADP-based control system. The Critic evaluates online the control policy, and the Actor implements the improved control policy.

1.1.3 What is RADP?

RADP is developed to address the presence of dynamic uncertainty in linear and nonlinear dynamical systems. See Figure 1.3 for an illustration. There are several reasons for which we pursue a new framework for RADP. First and foremost, it is well-known that building an exact mathematical model for physical systems often is a hard task. Also, even if the exact mathematical model can be obtained for some particular engineering and biological applications, simplified models are often more preferable for system analysis and control synthesis than the original complex system model. While we refer the mismatch between the simplified model and the original system to as dynamic uncertainty here, the engineering literature often uses the term of *unmodeled dynamics* instead. Secondly, the observation errors may often be captured by dynamic uncertainty. From the literature of modern nonlinear control [95, 80, 82], it is known that the presence of dynamic uncertainty makes the feedback control problem extremely challenging in the context of nonlinear systems. In order to broaden the application scope of ADP theory in the presence of dynamic uncertainty,

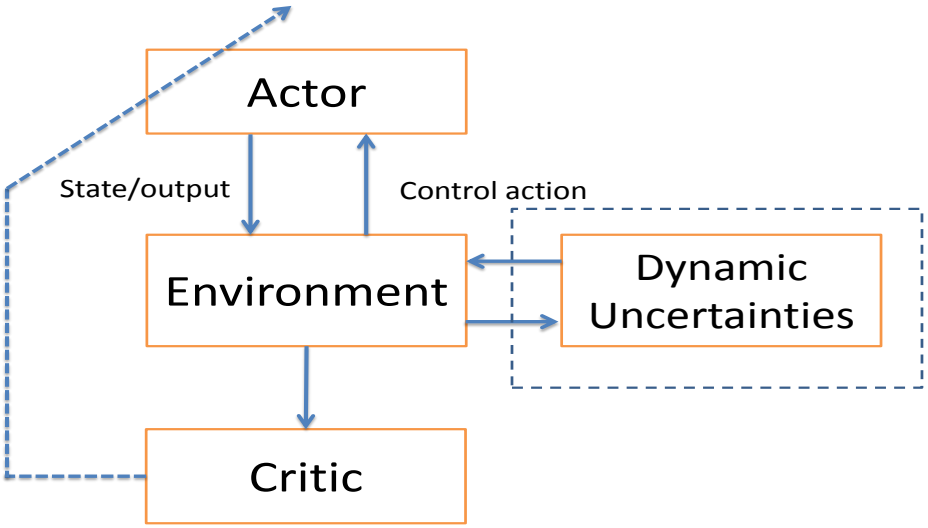

Figure 1.3: RADP with dynamic uncertainty, different from classical ADP architecture, a new component, known as dynamic uncertainty, is taken into consideration.

our strategy is to integrate tools from nonlinear control theory, such as Lyapunov designs, input-to-state stability theory [150], and nonlinear small-gain techniques [83]. This way RADP becomes applicable to wide classes of uncertain dynamic systems with incomplete state information and unknown system order/dynamics.

Additionally, RADP can be applied to large-scale dynamic systems as shown in our recent paper [70]. By integrating a simple version of the cyclic-small-gain theorem [109], asymptotic stability can be achieved by assigning appropriate weighting matrices for each subsystem. Further, certain suboptimality property can be obtained. Because of several emerging applications of practical importance such as smart electric grid, intelligent transportation systems and groups of mobile autonomous agents, this topic deserves further investigations from a RADP point of view. The existence of unknown parameters and/or dynamic uncertainties, and the limited information of state variables, give rise to challenges for the decentralized or distributed controller design of large-scale systems.

1.2 Contributions of this dissertation

Here we outline the key contributions of the dissertation as follows.

- We, for the first time, develop ADP methods for CT systems with completely unknown dynamics.
  - By taking advantages of the exploration noise, we remove the assumption in the past literature of ADP where partial knowledge of the system dynamics must be known [68].
  - We extend the method to affine nonlinear systems and rigorously show its stability and convergence properties [75].
  - We introduce sum-of-squares-based relaxation method into ADP theory to achieve global stabilization and suboptimal control of uncertain nonlinear systems via online learning [71].
- We propose the new theory of RADP, which fills a gap of ADP in the past literature where dynamic uncertainties or unmodeled dynamics are *not* addressed.
  - Inspired by the small-gain theorem [83], for partially linear systems, we give conditions on the design of the performance index to achieve robust stability [69].
  - We extend this technique to a class of large-scale systems [70].
  - By integration with tools from nonlinear control theory, *e.g.*, gain-assignment [129], and the Lyapunov-based small-gain condition [81], we redesign the approximated optimal controller to achieve robust stabilization of nonlinear system with dynamic uncertainties [75].
- We have applied the RADP theory to study both engineering and reverse-engineering problems.

– The RADP theory is used to study the robust optimal control of multimachine power systems [70, 73].

– Observing good consistency between experimental data [22, 41, 142] and our simulation results, we suggest that biological systems may use RADP-like schemes to interact with uncertain environment [74, 76].

Chapter 2

ADP for linear systems with completely unknown dynamics

The adaptive controller design for unknown linear systems has been intensively studied in the past literature [60], [113], [158]. A conventional way to design an adaptive optimal control law can be pursued by identifying the system parameters first and then solving the related algebraic Riccati equation. However, adaptive systems designed this way are known to respond slowly to parameter variations from the plant.

On the other hand, approximate/adaptive dynamic programming (ADP) [186] theories have been broadly applied for solving optimal control problems for uncertain systems in recent years (see, for example, [102, 177, 38, 2, 12, 34, 196, 66, 101, 173, 203, 192]). Among all the different ADP approaches, for discrete-time (DT) systems, the action-dependent heuristic dynamic programming (ADHDP) [189], or Q-learning [181], is an online iterative scheme that does not depend on the model to be controlled, and it has found applications in many engineering disciplines [1, 196, 101, 184].

Nevertheless, due to the different structures of the algebraic Riccati equations between DT and continuous-time (CT) systems, results developed for the DT setting cannot be directly applied for solving CT problems. Although some early attempts

have been made in [4, 122, 173], a common feature of all the existing ADP-based results is that partial knowledge of the system dynamics is assumed to be exactly known in the setting of CT systems.

The primary objective of this chapter is to remove this assumption on partial knowledge of the system dynamics, and thus to develop a truly knowledge-free ADP algorithm. More specifically, we propose a novel computational adaptive optimal control methodology that employs the approximate/adaptive dynamic programming technique to iteratively solve the algebraic Riccati equation using the online information of state and input, without requiring the a priori knowledge of the system matrices. In addition, all iterations can be conducted by using repeatedly the same state and input information on some fixed time intervals. It should be noticed that our approach serves as a fundamental computational tool to study ADP related problems for CT systems in the remainder of this dissertation.

This Chapter is organized as follows: In Section 2.1, we briefly introduce a policy iteration technique for solving standard CT linear systems. In Section 2.2, we develop our computational adaptive optimal control method and show its convergence. A practical online algorithm is provided. In Section 2.3, we apply the proposed approach to the optimal controller design problem of a turbocharged diesel engine with exhaust gas recirculation. Concluding remarks as well as potential future extensions are contained in Section 2.4.

2.1 Problem formulation and preliminaries

Consider a CT linear system described by

$$\dot{x} = Ax + Bu \tag{2.1}$$

where $x \in \mathbb{R}^n$ is the system state fully available for feedback control design; $u \in \mathbb{R}^m$ is the control input; $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are unknown constant matrices. In addition, the system is assumed to be stabilizable.

Recall the LQR design mentioned in Section 9.1.1. The design objective is to find a linear optimal control law in the form of

$$u = -Kx \tag{2.2}$$

which minimizes the following performance index

$$J = \int_0^\infty (x^T Q x + u^T R u) dt \tag{2.3}$$

where $Q = Q^T \geq 0$, $R = R^T > 0$, with $(A, Q^{1/2})$ observable.

By [99], solution to this problem can be found by solving the following well-known algebraic Riccati equation (ARE)

$$A^T P + PA + Q - PBR^{-1}B^T P = 0, \tag{2.4}$$

which has a unique symmetric positive definite solution P^*. Then, the optimal feedback gain matrix K^* in (2.2) is thus determined by

$$K^* = R^{-1}B^T P^*. \tag{2.5}$$

Since (2.4) is nonlinear in P, it is usually difficult to directly solve P^* from (2.4), especially for large-size matrices. Nevertheless, many efficient algorithms have been developed to numerically approximate the solution of (2.4). One of such algorithms was developed in [89], and is introduced in the following:

Theorem 2.1.1 ([89]). *Let $K_0 \in \mathbb{R}^{m\times n}$ be any stabilizing feedback gain matrix, and*

let P_k be the real symmetric positive definite solution of the Lyapunov equation

$$(A - BK_k)^T P_k + P_k (A - BK_k) + Q + K_k^T R K_k = 0 \tag{2.6}$$

where K_k, with $k = 1, 2, \cdots$, are defined recursively by:

$$K_k = R^{-1} B^T P_{k-1}. \tag{2.7}$$

Then, the following properties hold:

1. *$A - BK_k$ is Hurwitz,*
2. *$P^* \leq P_{k+1} \leq P_k$,*
3. *$\lim_{k\to\infty} K_k = K^*$, $\lim_{k\to\infty} P_k = P^*$.*

In [89], by iteratively solving the Lyapunov equation (2.6), which is linear in P_k, and updating K_k by (2.7), solution to the nonlinear equation (2.4) is numerically approximated.

For the purpose of solving (2.6) without the knowledge of A, in [173], (2.6) was implemented online by

$$x^T(t) P_k x(t) - x^T(t + \delta t) P_k x(t + \delta t) = \int_t^{t+\delta t} \left(x^T Q x + u_k^T R u_k \right) d\tau \tag{2.8}$$

where $u_k = -K_k x$ is the control input of the system on the time interval $[t, t + \delta t]$.

Since both x and u_k can be measured online, a real symmetric solution P_k can be uniquely determined under certain persistent excitation (PE) condition [173]. However, as we can see from (2.7), the exact knowledge of system matrix B is still required for the iterations. Also, to guarantee the PE condition, the state may need to be reset at each iteration step, but this may cause technical problems for stability analysis of the closed loop system [173]. An alternative way is to add exploration noise [19],

[167], [1], [196] such that $u_k = -K_k x + e$, with e the exploration noise, is used as the true control input in (2.8). As a result, P_k solved from (2.8) and the one solved from (2.6) are not exactly the same. In addition, after each time the control policy is updated, information of the state and input must be re-collected for the next iteration. This may slow down the learning process, especially for high-dimensional systems.

2.2 ADP-based online learning with completely unknown dynamics

In this section, we will present our new online learning strategy that does not rely on A nor B.

To this end, we rewrite the original system (2.1) as

$$\dot{x} \;=\; A_k x + B(K_k x + u) \tag{2.9}$$

where $A_k = A - BK_k$.

Then, along the solutions of (2.9), by (2.6) and (2.7) it follows that

$$\begin{aligned} & x(t+\delta t)^T P_k x(t+\delta t) - x(t)^T P_k x(t) \\ = \;& \int_t^{t+\delta t} \left[x^T (A_k^T P_k + P_k A_k) x + 2(u + K_k x)^T B^T P_k x \right] d\tau \\ = \;& -\int_t^{t+\delta t} x^T Q_k x \; d\tau + 2 \int_t^{t+\delta t} (u + K_k x)^T R K_{k+1} x \; d\tau \end{aligned} \tag{2.10}$$

where $Q_k = Q + K_k^T R K_k$.

Remark 2.2.1. *Notice that in* (2.10)*, the term* $x^T(A_k^T P_k + P_k A_k)x$ *depending on the unknown matrices* A *and* B *is replaced by* $-x^T Q_k x$*, which can be obtained by measuring the state online. Also, the term* $B^T P_k$ *involving* B *is replaced by* RK_{k+1}*, in which* K_{k+1} *is treated as another unknown matrix to be solved together with* P_k*.*

Therefore, (2.10) *plays an important role in separating the system dynamics from the iterative process. As a result, the requirement of the system matrices in (2.6) and (2.7) can be replaced by the state and input information measured online.*

Remark 2.2.2. *It is also noteworthy that in* (2.10) *we always have exact equality if P_k, K_{k+1} satisfy (2.6), (2.7), and x is the solution of system (2.9) with arbitrary control input u. This fact enables us to employ $u = -K_0 x + e$, with e the exploration noise, as the input signal for learning, without affecting the convergence of the learning process.*

Next, we show that given a stabilizing K_k, a pair of matrices (P_k, K_{k+1}), with $P_k = P_k^T > 0$, satisfying (2.6) and (2.7) can be uniquely determined without knowing A or B , under certain condition. To this end, we define the following two operators:

$$\nu(P) : \mathbb{R}^{n\times n} \to \mathbb{R}^{\frac{1}{2}n(n+1)}, \quad \text{and} \quad \mu(x) : \mathbb{R}^n \to \mathbb{R}^{\frac{1}{2}n(n+1)}$$

such that

$$\begin{aligned} \nu(P) &= [p_{11}, 2p_{12}, \cdots, 2p_{1n}, p_{22}, 2p_{23}, \cdots, 2p_{n-1,n}, p_{nn}]^T, \\ \mu(x) &= [x_1^2, x_1x_2, \cdots, x_1x_n, x_2^2, x_2x_3, \cdots, x_{n-1}x_n, x_n^2]^T. \end{aligned}$$

In addition, by Kronecker product representation, we have

$$x^T Q_k x = \left(x^T \otimes x^T\right) \operatorname{vec}(Q_k),$$

and

$$\begin{aligned} & (u + K_k x)^T R K_{k+1} x \\ = \; & \left[(x^T \otimes x^T)(I_n \otimes K_k^T R) + (x^T \otimes u^T)(I_n \otimes R)\right] \operatorname{vec}(K_{k+1}). \end{aligned}$$

Further, for positive integer l, we define matrices $\delta_{xx} \in \mathbb{R}^{l\times\frac{1}{2}n(n+1)}$, $I_{xx} \in \mathbb{R}^{l\times n^2}$, $I_{xu} \in \mathbb{R}^{l\times mn}$, such that

$$\begin{aligned}
\delta_{xx} &= \left[\begin{array}{cccc} \mu(x(t_1)) - \mu(x(t_0)), & \mu(x(t_2)) - \mu(x(t_1)), & \cdots, & \mu(x(t_l)) - \mu(x(t_{l-1})) \end{array} \right]^T, \\
I_{xx} &= \left[\begin{array}{cccc} \int_{t_0}^{t_1} x\otimes x\ d\tau, & \int_{t_1}^{t_2} x\otimes x\ d\tau, & \cdots, & \int_{t_{l-1}}^{t_l} x\otimes x\ d\tau \end{array} \right]^T, \\
I_{xu} &= \left[\begin{array}{cccc} \int_{t_0}^{t_1} x\otimes u\ d\tau, & \int_{t_1}^{t_2} x\otimes u\ d\tau, & \cdots, & \int_{t_{l-1}}^{t_l} x\otimes u\ d\tau \end{array} \right]^T,
\end{aligned}$$

where $0 \le t_0 < t_1 < \cdots < t_l$.

Then, for any given stabilizing gain matrix K_k, (2.10) implies the following matrix form of linear equations

$$\Theta_k \left[\begin{array}{c} \nu(P_k) \\ \mathrm{vec}\,(K_{k+1}) \end{array} \right] = \Xi_k \tag{2.11}$$

where $\Theta_k \in \mathbb{R}^{l\times\left[\frac{1}{2}n(n+1)+mn\right]}$ and $\Xi_k \in \mathbb{R}^l$ are defined as:

$$\begin{aligned}
\Theta_k &= \left[\delta_{xx}, -2I_{xx}(I_n\otimes K_k^T R) - 2I_{xu}(I_n\otimes R)\right], \\
\Xi_k &= -I_{xx}\ \mathrm{vec}(Q_k).
\end{aligned}$$

Notice that if Θ_k has full column rank, (2.11) can be directly solved as follows:

$$\left[\begin{array}{c} \nu(P_k) \\ \mathrm{vec}\,(K_{k+1}) \end{array} \right] = (\Theta_k^T\Theta_k)^{-1}\Theta_k^T\Xi_k. \tag{2.12}$$

Now, we are ready to give the following computational adaptive optimal control algorithm for practical online implementation. A flowchart of Algorithm 2.2.1 is shown in Figure 2.1.

Remark 2.2.3. *Computing the matrices I_{xx} and I_{xu} carries the main burden in*

Algorithm 2.2.1 ADP algorithm

1: Employ $u = -K_0 x + e$ as the input on the time interval $[t_0,\ t_l]$, where K_0 is stabilizing and e is the exploration noise. Compute δ_{xx}, I_{xx} and I_{xu} until the rank condition in (2.13) below is satisfied. Let $k = 0$.
2: Solve P_k and K_{k+1} from (2.12).
3: Let $k \leftarrow k + 1$, and repeat Step 2 until $\|P_k - P_{k-1}\| \leq \epsilon$ for $k \geq 1$, where the constant $\epsilon > 0$ is a predefined small threshold.
4: Use $u = -K_k x$ as the approximated optimal control policy.

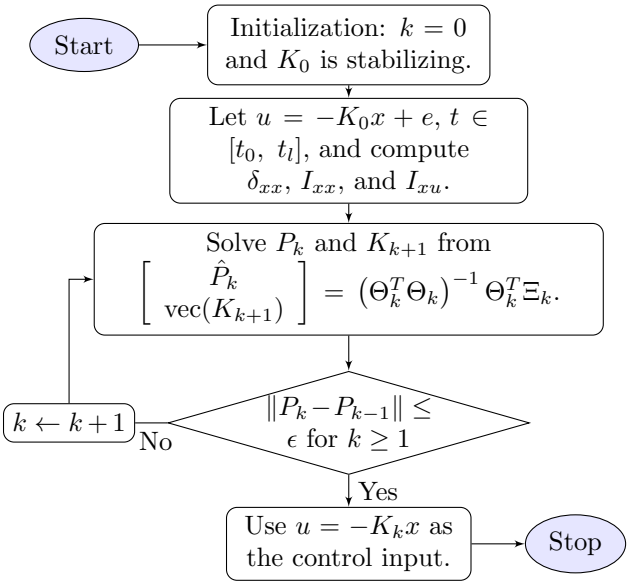

Figure 2.1: Flowchart of Algorithm 2.2.1.

performing Algorithm 2.2.1. The two matrices can be implemented using $\frac{1}{2}n(n+1) + mn$ *integrators in the learning system to collect information of the state and the input.*

Remark 2.2.4. *In practice, numerical error may occur when computing* I_{xx} *and* I_{xu}*. As a result, the solution of (2.11) may not exist. In that case, the solution of (2.12) can be viewed as the least squares solution of (2.11).*

Next, we show that the convergence of Algorithm 2.2.1 can be guaranteed under certain condition.

Lemma 2.2.1. *If there exists an integer $l_0 > 0$, such that, for all $l \geq l_0$,*

$$\text{rank}\left(\left[\begin{array}{cc} I_{xx}, & I_{xu} \end{array}\right]\right) = \frac{n(n+1)}{2} + mn, \tag{2.13}$$

then Θ_k has full column rank for all $k \in \mathbb{Z}_+$.

Proof: It amounts to show that the following linear equation

$$\Theta_k X = 0 \tag{2.14}$$

has only the trivial solution $X = 0$.

To this end, we prove by contradiction. Assume $X = \left[\begin{array}{cc} Y_v^T & Z_v^T \end{array}\right]^T \in \mathbb{R}^{\frac{1}{2}n(n+1)+mn}$ is a nonzero solution of (2.14), where $Y_v \in \mathbb{R}^{\frac{1}{2}n(n+1)}$ and $Z_v \in \mathbb{R}^{mn}$. Then, a symmetric matrix $Y \in \mathbb{R}^{n\times n}$ and a matrix $Z \in \mathbb{R}^{m\times n}$ can be uniquely determined, such that $\nu(Y) = Y_v$ and $\text{vec}(Z) = Z_v$.

By (2.10), we have

$$\Theta_k X = I_{xx}\text{vec}(M) + 2I_{xu}\text{vec}(N) \tag{2.15}$$

where

$$M = A_k^T Y + Y A_k + K_k^T(B^T Y - RZ) + (YB - Z^T R)K_k, \tag{2.16}$$

$$N = B^T Y - RZ. \tag{2.17}$$

Notice that since M is symmetric, we have

$$I_{xx}\text{vec}(M) = I_{\bar{x}}\nu(M) \tag{2.18}$$

where $I_{\bar{x}} \in \mathbb{R}^{l \times \frac{1}{2}n(n+1)}$ is defined as:

$$I_x = \left[\int_{t_0}^{t_1} \mu(x)d\tau, \quad \int_{t_1}^{t_2} \mu(x)d\tau, \quad \cdots, \quad \int_{t_{l-1}}^{t_l} \mu(x)d\tau \right]^T. \tag{2.19}$$

Then, (2.14) and (2.15) imply the following matrix form of linear equations

$$\left[I_x, \quad 2I_{xu} \right] \left[\begin{array}{c} \nu(M) \\ \text{vec}(N) \end{array} \right] = 0. \tag{2.20}$$

Under the rank condition in (2.13), we know $\left[I_x, \quad 2I_{xu} \right]$ has full column rank. Therefore, the only solution to (2.20) is $\nu(M) = 0$ and $\text{vec}(N) = 0$. As a result, we have $M = 0$ and $N = 0$.

Now, by (2.17) we know $Z = R^{-1}B^TY$, and (2.16) is reduced to the following Lyapunov equation

$$A_k^T Y + Y A_k = 0. \tag{2.21}$$

Since A_k is Hurwitz for all $k \in \mathbb{Z}_+$, the only solution to (2.21) is $Y = 0$. Finally, by (2.17) we have $Z = 0$.

In summary, we have $X = 0$. But it contradicts with the assumption that $X \neq 0$. Therefore, Θ_k must have full column rank for all $k \in \mathbb{Z}_+$. The proof is complete. □

Theorem 2.2.1. *Starting from a stabilizing $K_0 \in \mathbb{R}^{m \times n}$, when the condition of Lemma 2.2.1 is satisfied, the sequences $\{P_i\}_{i=0}^{\infty}$ and $\{K_j\}_{j=1}^{\infty}$ obtained from solving (2.12) converge to the optimal values P^* and K^*, respectively.*

Proof: Given a stabilizing feedback gain matrix K_k, if $P_k = P_k^T$ is the solution of (2.6), K_{k+1} is uniquely determined by $K_{k+1} = R^{-1}B^TP_k$. By (2.10), we know that P_k and K_{k+1} satisfy (2.12). On the other hand, let $P = P^T \in \mathbb{R}^{n \times n}$ and $K \in \mathbb{R}^{m \times n}$,

such that

$$\Theta_k \begin{bmatrix} \nu(P) \\ \mathrm{vec}(K) \end{bmatrix} = \Xi_k.$$

Then, we immediately have $\nu(P) = \nu(P_k)$ and $\mathrm{vec}(K) = \mathrm{vec}(K_{k+1})$. By Lemma 2.2.1, $P = P^T$ and K are unique. In addition, by the definitions of $\nu(P)$ and $\mathrm{vec}(K)$, $P_k = P$ and $K_{k+1} = K$ are uniquely determined.

Therefore, the policy iteration (2.12) is equivalent to (2.6) and (2.7). By Theorem 2.1.1, the convergence is thus proved. □

Remark 2.2.5. *It can be seen that Algorithm 2.2.1 contains two separated phases: First, an initial stabilizing control policy with exploration noise is applied and the online information is recorded in matrices δ_{xx}, I_{xx}, and I_{xu} until the rank condition in (2.13) is satisfied. Second, without requiring additional system information, the matrices δ_{xx}, I_{xx}, and I_{xu} are repeatedly used to implement the iterative process. A sequence of controllers, that converges to the optimal control policy, can be obtained.*

Remark 2.2.6. *The choice of exploration noise is not a trivial task for general reinforcement learning problems and other related machine learning problems, especially for high dimensional systems. In solving practical problems, several types of exploration noise have been adopted, such as random noise [1], [196], exponentially decreasing probing noise [167]. For the simulations in the next section, we will use the sum of sinusoidal signals with different frequencies, as in [69].*

Remark 2.2.7. *In some sense, our approach is related to the ADHDP [189], or Q-learning [181] method for DT systems. Indeed, it can be viewed that we solve the following matrix H_k at each iteration step*

$$H_k = \begin{bmatrix} H_{11,k} & H_{12,k} \\ H_{21,k} & H_{22,k} \end{bmatrix} = \begin{bmatrix} P_k & P_k B \\ B^T P_k & R \end{bmatrix}. \tag{2.22}$$

Once this matrix is obtained, the control policy can be updated by $K_{k+1} = H_{22,k}^{-1} H_{21,k}$. *The DT version of the* H_k *matrix can be found in [19] and [102].*

2.3 Application to a turbocharged diesel engine

In this section, we study the controller design for a turbocharged diesel engine with exhaust gas recirculation [84]. The open loop model is a six-th order CT linear system. The system matrices A and B are directly taken from [84] and shown as follows:

$$A = \begin{bmatrix} -0.4125 & -0.0248 & 0.0741 & 0.0089 & 0 & 0 \\ 101.5873 & -7.2651 & 2.7608 & 2.8068 & 0 & 0 \\ 0.0704 & 0.0085 & -0.0741 & -0.0089 & 0 & 0.0200 \\ 0.0878 & 0.2672 & 0 & -0.3674 & 0.0044 & 0.3962 \\ -1.8414 & 0.0990 & 0 & 0 & -0.0343 & -0.0330 \\ 0 & 0 & 0 & -359.0000 & 187.5364 & -87.0316 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.0042 & -1.0360 & 0.0042 & 0.1261 & 0 & 0 \\ 0.0064 & 1.5849 & 0 & 0 & -0.0168 & 0 \end{bmatrix}^T.$$

In order to illustrate the efficiency of the proposed computational adaptive optimal control strategy, the precise knowledge of A and B is not used in the design of optimal controllers. Since the physical system is already stable, the initial stabilizing feedback gain can be set as $K_0 = 0$.

The weighting matrices are selected to be

$$Q = \text{diag}\begin{pmatrix} 1, & 1, & 0.1, & 0.1, & 0.1, & 0.1 \end{pmatrix}, \; R = I_2.$$

In the simulation, the initial values for the state variables are randomly selected around the origin. From $t = 0s$ to $t = 2s$, the following exploration noise is used as

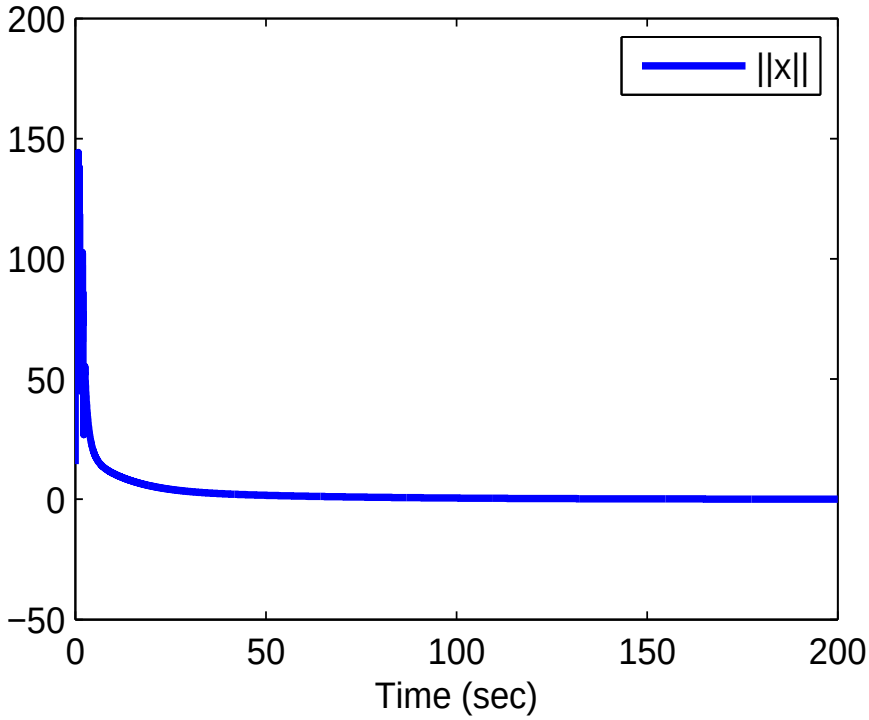

Figure 2.2: Trajectory of the Euclidean norm of the state variables during the simulation.

the system input

$$e = 100 \sum_{i=1}^{100} \sin(\omega_i t) \tag{2.23}$$

where ω_i, with $i = 1, \cdots, 100$, are randomly selected from $[-500, 500]$.

State and input information is collected over each interval of $0.01s$. The policy iteration started at $t = 2s$, and convergence is attained after 16 iterations, when the stopping criterion $\|P_k - P_{k-1}\| \leq 0.03$ is satisfied. The formulated controller is used as the actual control input to the system starting from $t = 2s$ to the end of the simulation. The trajectory of the Euclidean norm of all the state variables is shown in Figure 2.2. The system output variables $y_1 = 3.6x_6$ and $y_2 = x_4$, denoting the mass air flow (MAF) and the intake manifold absolute pressure (MAP) [84], are plotted in Figure 2.3.

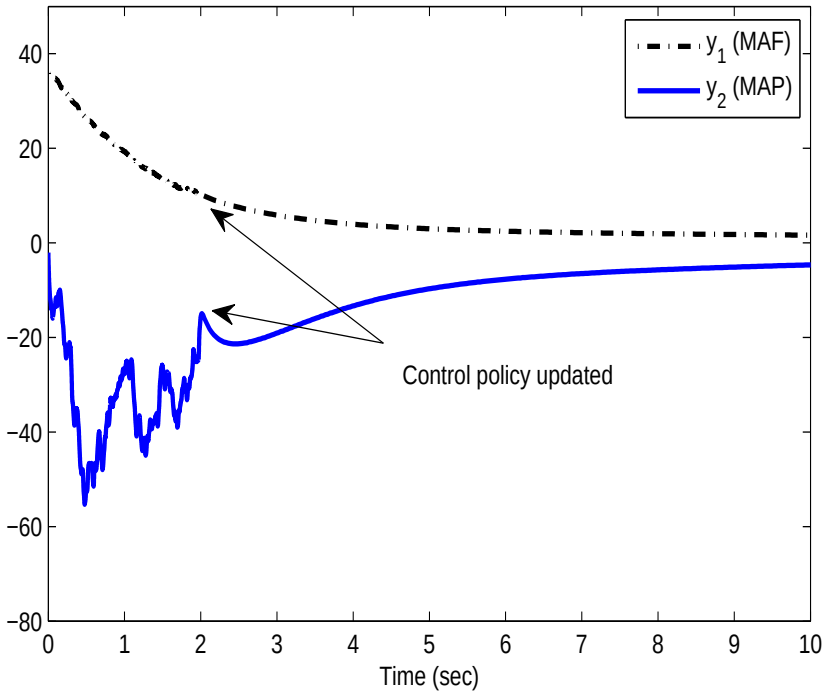

Figure 2.3: Trajectories of the output variables from $t = 0s$ to $t = 10$s.

The proposed algorithm gives the cost and the feedback gain matrices as shown below:

$$P_{15} = \begin{bmatrix} 127.5331 & 0.5415 & 16.8284 & 1.8305 & 1.3966 & 0.0117 \\ 0.5415 & 0.0675 & 0.0378 & 0.0293 & 0.0440 & 0.0001 \\ 16.8284 & 0.0378 & 18.8105 & -0.3317 & 4.1648 & 0.0012 \\ 1.8305 & 0.0293 & -0.3317 & 0.5041 & -0.1193 & -0.0001 \\ 1.3966 & 0.0440 & 4.1648 & -0.1193 & 3.3985 & 0.0004 \\ 0.0117 & 0.0001 & 0.0012 & -0.0001 & 0.0004 & 0.0006 \end{bmatrix},$$

$$K_{15} = \begin{bmatrix} -0.7952 & -0.0684 & -0.0725 & 0.0242 & -0.0488 & -0.0002 \\ 1.6511 & 0.1098 & 0.0975 & 0.0601 & 0.0212 & 0.0002 \end{bmatrix}.$$

By solving directly the algebraic Riccati equation (2.4), we obtain the optimal

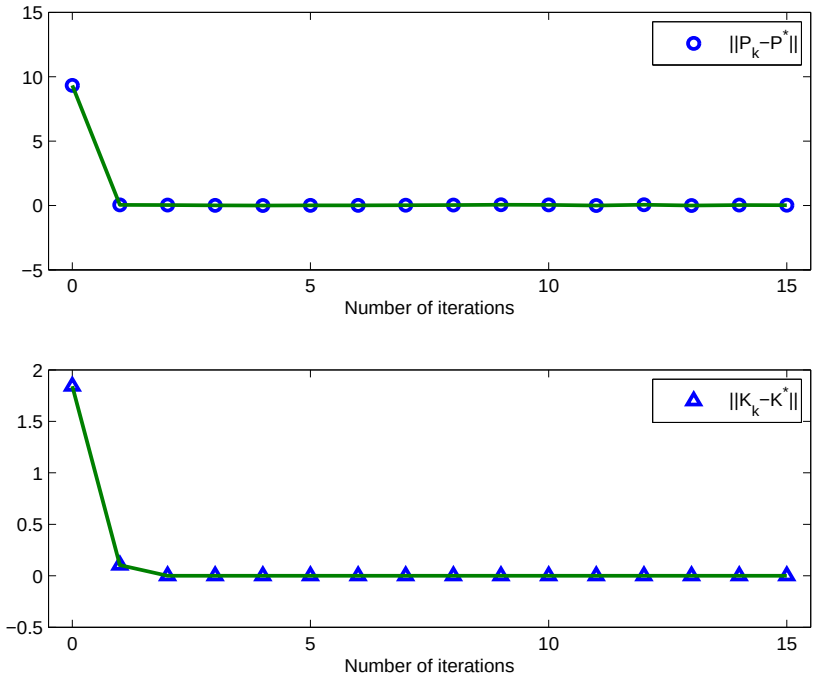

Figure 2.4: Convergence of P_k and K_k to their optimal values P^* and K^* during the learning process.

solutions:

$$P^* = \begin{bmatrix} 127.5325 & 0.5416 & 16.8300 & 1.8307 & 1.4004 & 0.0117 \\ 0.5416 & 0.0675 & 0.0376 & 0.0292 & 0.0436 & 0.0001 \\ 16.8300 & 0.0376 & 18.8063 & -0.3323 & 4.1558 & 0.0012 \\ 1.8307 & 0.0292 & -0.3323 & 0.5039 & -0.1209 & -0.0001 \\ 1.4004 & 0.0436 & 4.1558 & -0.1209 & 3.3764 & 0.0004 \\ 0.0117 & 0.0001 & 0.0012 & -0.0001 & 0.0004 & 0.0006 \end{bmatrix},$$

$$K^* = \begin{bmatrix} -0.7952 & -0.0684 & -0.0726 & 0.0242 & -0.0488 & -0.0002 \\ 1.6511 & 0.1098 & 0.0975 & 0.0601 & 0.0213 & 0.0002 \end{bmatrix}.$$

The convergence of P_k and K_k to their optimal values is illustrated in Figure 2.4. Notice that if B is accurately known, the problem can also be solved using the method in [173]. However, that method requires a total learning time of $32s$ for 16 iterations, if the state and input information within $2s$ is collected for each iteration. In addition, the method in [173] may need to reset the state at each iteration step, in order to satisfy the PE condition.

2.4 Conclusions

A novel computational policy iteration approach for finding online adaptive optimal controllers for CT linear systems with completely unknown system dynamics has been presented in this chapter. This method solves the algebraic Riccati equation iteratively using system state and input information collected online, without knowing the system matrices. A practical online algorithm was proposed and has been applied to the controller design for a turbocharged diesel engine with unknown parameters. The methodology developed in this chapter serves as an important computational tool to study the adaptive optimal control of CT systems. It is essential to the theories

developed in the remaining chapters of this dissertation.

Chapter 3

RADP for uncertain partially linear systems

In the previous chapter, we have developed an ADP methodology for CT linear systems. Similar to the past literature of ADP, it is assumed that the system order is known and the state variables are fully available. However, the system order may be unknown due to the presence of dynamic uncertainties (or unmodeled dynamics) [82], which are motivated by engineering applications in situations where the exact mathematical model of a physical system is not easy to be obtained. Of course, dynamic uncertainties also make sense for the mathematical modeling in other branches of science such as biology and economics. This problem, often formulated in the context of robust control theory, cannot be viewed as a special case of output feedback control. In addition, the ADP methods developed in the past literature may fail to guarantee not only optimality, but also the stability of the closed-loop system when dynamic uncertainty occurs. In the seminal paper [192], Werbos also pointed out the related issue that the performance of learning may deteriorate when using incomplete data in ADP.

In order to capture and model this feature from biological learning, in this chap-

ter we propose a new concept of robust adaptive dynamic programming, a natural extension of ADP to uncertain dynamic systems. It is worth noting that we focus on the presence of dynamic uncertainties of which the state variables and the system order are not precisely known.

As the first distinctive feature of the proposed RADP framework, the controller design issue is addressed from a point of view of robust control with disturbance attenuation. Specifically, by means of the popular backstepping approach [95], we will show that a robust control policy, or adaptive critic, can be synthesized to yield an arbitrarily small L_2-gain with respect to the disturbance input. In addition, by studying the relationship between optimality and robustness, it is shown that in the absence of disturbance input, the robust control policy also preserves optimality with respect to some iteratively constructed cost function. It should be mentioned that in [1] the theory of zero-sum games was employed in ADP design but the gain from the disturbance input to the output cannot be made arbitrarily small.

Our study on the effects of dynamic uncertainties, or unmodeled dynamics, is motivated by engineering applications in situations where the exact mathematical model of a physical system is not easy to be obtained. The presence of dynamic uncertainty gives rise to interconnected systems for which the controller design and robustness analysis become technically challenging. With this observation in mind, we will adopt notions of input-to-output stability and strong unboundedness observability introduced in the nonlinear control community; see, for instance, [61, 83], and [150]. We achieve the robust stability and suboptimality properties for the overall interconnected system, by means of Lyapunov and small-gain techniques [83].

This chapter is organized as follows. Section 3.1 formulates the problem. Section 3.2 investigates the relationship between optimality and robustness for a general class of partially linear, uncertain composite systems [133]. Section 3.3 presents a RADP scheme for partial-state feedback design. Section 3.4 demonstrates the proposed

RADP design methodology by means of an one-machine infinite-bus power system. Concluding remarks are contained in Section 3.5.

3.1 Problem formulation

Consider the following partially linear composite system

$$\dot{w} = f(w, y), \tag{3.1}$$

$$\dot{x} = Ax + B\left[z + \Delta_1(w, y)\right], \tag{3.2}$$

$$\dot{z} = Ex + Fz + G\left[u + \Delta_2(w, y)\right], \tag{3.3}$$

$$y = Cx \tag{3.4}$$

where $[x^T, z^T]^T \in \mathbb{R}^n \times \mathbb{R}^m$ is the system state vector; $w \in \mathbb{R}^{n_w}$ is the state of the dynamic uncertainty; $y \in \mathbb{R}^q$ is the output; $u \in \mathbb{R}^m$ is the control input; $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{q\times n}$, $E \in \mathbb{R}^{m\times n}$, $F \in \mathbb{R}^{m\times m}$, and $G \in \mathbb{R}^{m\times m}$ are unknown constant matrices with the pair (A, B) stabilizable and G nonsingular; $\Delta_1(w, y) = D\Delta(w, y)$ and $\Delta_2(w, y) = H\Delta(w, y)$ are the outputs of the dynamic uncertainty with $D, H \in \mathbb{R}^{m\times p}$ two unknown constant matrices; the unknown functions $f : \mathbb{R}^{n_w} \times \mathbb{R}^q \to \mathbb{R}^{n_w}$ and $\Delta : \mathbb{R}^{n_w} \times \mathbb{R}^q \to \mathbb{R}^p$ are locally Lipschitz satisfying $f(0, 0) = 0$, $\Delta(0, 0) = 0$. In addition, assume the upper bounds of the norms of B, D, H, and G^{-1} are known.

Our objective is to find online a robust optimal control policy that globally asymptotically stabilizes the system (3.1)-(3.4) at the origin.

3.2 Optimality and robustness

In this section, we show the existence of a robust optimal control policy that globally asymptotically stabilizes the overall system (3.1)-(3.4). To this end, let us make a few assumptions on (3.1), which are often required in the literature of nonlinear control

design [61, 82, 95].

Assumption 3.2.1. *The w-subsystem* (3.1) *has strong unboundedness observability (SUO) property with zero offset [83] and is input-to-output stable (IOS) with respect to y as the input and* Δ *as the output [83, 150].*

Assumption 3.2.2. *There exist a continuously differentiable, positive definite, radially unbounded function* $U : \mathbb{R}^{n_w} \to \mathbb{R}_+$*, and a constant* $c \geq 0$ *such that*

$$\dot{U} = \frac{\partial U(w)}{\partial w} f(w, y) \leq -2|\Delta|^2 + c|y|^2 \tag{3.5}$$

for all $w \in \mathbb{R}^{n_w}$ *and* $y \in \mathbb{R}^q$*.*

We now show that an arbitrarily small L_2 gain can be obtained for the subsystem (3.2)-(3.4). Given any arbitrarily small constant $\gamma > 0$, we can choose Q and R in (3.15) such that $Q \geq \gamma^{-1} C^T C$ and $R^{-1} \geq DD^T$. Define $K^* = R^{-1} B^T P^*$, $\xi = z + K^* x$, $A_c = A - BK^*$, and let $S^* > 0$ be the symmetric solution of the ARE

$$\bar{F}^T S^* + S^* \bar{F} + W - S^* G R_1^{-1} G^T S^* = 0 \tag{3.6}$$

where $W > 0$, $R_1^{-1} \geq \bar{D}\bar{D}^T$, $\bar{F} = F + K^* B$, and $\bar{D} = H + G^{-1} K^* BD$. Further, define $\bar{E} = E + K^* A_c - FK^*$, $M^* = R_1^{-1} G^T S^*$, $N^* = S^* \bar{E}$.

The following theorem gives the small-gain condition for the robust asymptotic stability of the overall system (3.1)-(3.4).

Theorem 3.2.1. *Under Assumptions 3.2.1 and 3.2.2, the control policy*

$$u^* \;=\; -\left[(M^{*T} R_1)^{-1} (N^* + RK^*) + M^* K^* \right] x - M^* z \tag{3.7}$$

globally asymptotically stabilizes the closed-loop system comprised of (3.1)-(3.4), if

the small-gain condition holds:

$$\gamma c < 1. \tag{3.8}$$

Proof. Define

$$V(x, z, w) = x^T P^* x + \xi^T S^* \xi + U(w). \tag{3.9}$$

Then, along the solutions of the closed-loop system comprised of (3.1)-(3.4) and (3.7), by completing squares, it follows that

$$\begin{aligned}
\dot{V} &= \frac{d}{dt}(x^T P^* x) + \frac{d}{dt}(\xi^T S^* \xi) + \dot{U} \\
&\leq -\gamma^{-1}|y|^2 + |\Delta|^2 + 2x^T P^* B\xi + |\Delta|^2 \\
&\quad -2\xi^T B^T P^* x + \left(c|y|^2 - 2|\Delta|^2\right) \\
&\leq -\gamma^{-1}(1 - c\gamma)|y|^2
\end{aligned}$$

Therefore, we know $\lim_{t\to\infty} y(t) = 0$. By Assumption 3.2.1, all solutions of the closed-loop system are globally bounded. Moreover, a direct application of LaSalle's Invariance Principle [86] yields the GAS property of the trivial solution of the closed-loop system.

The proof is thus complete. □

Next, we show that the control policy (3.7) is suboptimal, i.e., it is optimal with respect to some cost function in the absence of the dynamic uncertainty. Notice that, with $\Delta \equiv 0$, the system (3.2)-(3.3) can be rewritten in a more compact matrix form:

$$\dot{X} = A_1 X + G_1 v \tag{3.10}$$

where $X = \begin{bmatrix} x^T \\ \xi^T \end{bmatrix}$, $v = u + G^{-1}\left[\bar{E} + (S^*)^{-1}B^T P^*\right] x$, $A_1 = \begin{bmatrix} A_c & B \\ -(S^*)^{-1}B^T P^* & \bar{F} \end{bmatrix}$,

and $G_1 = \begin{bmatrix} 0 \\ G \end{bmatrix}$.

Proposition 3.2.1. *Under the conditions of Theorem 3.2.1, the performance index*

$$J_1 = \int_0^\infty \left[X^T Q_1 X + v^T R_1 v\right] d\tau \tag{3.11}$$

for system (3.10) is minimized under the control policy

$$v^* = u^* + G^{-1}\left[\bar{E} + (S^*)^{-1} B^T P^*\right] x \tag{3.12}$$

Proof. It is easy to check that $\bar{P}^* = \text{block diag}\,(P^*, S^*)$ is the solution to the following ARE

$$A_1^T \bar{P}^* + \bar{P}^* A_1 + Q_1 - \bar{P}^* G_1 R_1^{-1} G_1^T \bar{P}^* = 0 \tag{3.13}$$

where $Q_1 = \text{block diag}\,\left(Q + K^{*T} R K^*, W\right)$.

Therefore, by linear optimal control theory [99], we obtain the optimal control policy

$$v^* = -R_1^{-1} G_1^T \bar{P}^* X = -M^* \xi. \tag{3.14}$$

The proof is thus complete. □

Remark 3.2.1. *It is of interest to note that Theorem 3.2.1 can be generalized to higher-dimensional systems with a lower-triangular structure, by a repeated application of backstepping and small-gain techniques in nonlinear control.*

Remark 3.2.2. *The cost function introduced here is different from the ones used in game theory [1, 171], where the policy iterations are developed based on the game algebraic Riccati equation (GARE). The existence of a solution of the GARE cannot*

be guaranteed when the input-output gain is arbitrarily small. Therefore, a significant advantage of our method vs. the game-theoretic approach of [1, 171] is that we are able to render the gain arbitrarily small.

3.3 RADP design

In this section, we develop a novel robust-ADP scheme to approximate the robust optimal control policy (3.7). This scheme contains two learning phases. Phase-one computes the matrices K^* and P^*. Then, based on the results derived from phase-one, the second learning phase further computes the matrices S^*, M^*, and N^*. It is worth noticing that the knowledge of A, B, E, and F is not required in our learning algorithm. In addition, we will analyze the robust asymptotic stability of the overall system under the approximated control policy obtained from our algorithm.

3.3.1 Phase-one learning

First, recall that, given K_0 such that $A - BK_0$ is Hurwitz, we can solve numerically an ARE in the following form

$$P^*A + A^TP^* + Q - P^*BR^{-1}B^TP^* = 0 \tag{3.15}$$

with $Q = Q^T \geq 0$, $R = R^T > 0$, and $(A, Q^{1/2})$ observable, by iteratively finding P_k and K_k from

$$\begin{aligned} 0 &= (A - BK_k)^TP_k + P_k(A - BK_k) + Q + K_k^TRK_k, & (3.16)\\ K_{k+1} &= R^{-1}B^TP_k. & (3.17) \end{aligned}$$

Now, assume all the conditions of Theorem 3.2.1 are satisfied. Along the trajec-

tories of (3.2), it follows that

$$\begin{aligned} x^T P_k x\big|_t^{t+T} &= 2\int_t^{t+T} (z+\Delta_1+K_k x)^T R K_{k+1} x d\tau \\ &\quad - \int_t^{t+T} x^T (Q + K_k^T R K_k) x d\tau. \end{aligned} \tag{3.18}$$

Using Kronecker product representation, (7.59) can be rewritten as

$$\begin{aligned} & x^T \otimes x^T\big|_t^{t+T} \mathrm{vec}(P_k) \\ = \; & 2\left[\int_t^{t+T} x^T \otimes (z+\Delta_1+K_k x)^T d\tau\right](I_n \otimes R)\mathrm{vec}(K_{k+1}) \\ & - \left[\int_t^{t+T} x^T \otimes x^T d\tau\right]\mathrm{vec}(Q + K_k^T R K_k). \end{aligned} \tag{3.19}$$

For any $\phi \in \mathbb{R}^{n_\phi}$, $\varphi \in \mathbb{R}^{n_\psi}$, and sufficiently large $l > 0$, we define the operators $\delta_{\phi\psi} : \mathbb{R}^{n_\phi} \times \mathbb{R}^{n_\psi} \to \mathbb{R}^{l \times n_\phi n_\psi}$ and $I_{\phi\psi} : \mathbb{R}^{n_\phi} \times \mathbb{R}^{n_\psi} \to \mathbb{R}^{l \times n_\phi n_\psi}$ such that

$$\begin{aligned} \delta_{\phi\psi} &= \left[\begin{array}{cccc} \phi\otimes\psi|_{t_0}^{t_1} & \phi\otimes\psi|_{t_1}^{t_2} & \cdots & \phi\otimes\psi|_{t_{l-1}}^{t_l} \end{array}\right]^T, \\ I_{\phi\psi} &= \left[\begin{array}{cccc} \int_{t_0}^{t_1} \phi\otimes\psi d\tau & \int_{t_1}^{t_2} \phi\otimes\psi d\tau & \cdots & \int_{t_{l-1}}^{t_l} \phi\otimes\psi d\tau \end{array}\right]^T \end{aligned}$$

where $0 \le t_0 < t_1 < \cdots < t_l$ are constants.

Then, (3.19) implies the following matrix form of linear equations

$$\Theta_k \left[\begin{array}{c} \mathrm{vec}(P_k) \\ \mathrm{vec}(K_{k+1}) \end{array}\right] = \Xi_k \tag{3.20}$$

where $\Theta_k \in \mathbb{R}^{l \times n(n+m)}$ and $\Xi_k \in \mathbb{R}^l$ are defined as

$$\begin{aligned} \Theta_k &= \left[\delta_{xx} \; -2I_{xx}(I_n \otimes K_k^T R) - 2(I_{xz} + I_{x\Delta_1})(I_n \otimes R)\right], \\ \Xi_k &= -I_{xx}\ \mathrm{vec}(Q + K_k^T R K_k). \end{aligned}$$

Given K_k such that $A - BK_k$ is Hurwitz, if there is a unique pair of matrices (P_k, K_{k+1}), with $P_k = P_k^T$, satisfying (3.20), we are able to replace (3.16) and (3.17) with (3.20). In this way, the iterative process does not need the knowledge of A and B.

Next, we approximate the matrices S^*, M^*, and N^*, which also appear in (3.7).

3.3.2 Phase-two learning

For the matrix $K_k \in \mathbb{R}^{m\times n}$ obtained from phase-one learning, let us define $\hat{\xi} = z + K_k x$. Then,

$$\dot{\hat{\xi}} = E_k x + F_k \hat{\xi} + G(u + \Delta_2) + K_k B \Delta_1 \tag{3.21}$$

where $E_k = E + K_k(A - BK_k) - FK_k$, $F_k = F + K_k B$.

Similarly as in phase-one learning, we seek the online implementation of the following iterative equations:

$$\begin{aligned} 0 &= S_{k,j}\, F_{k,j} + F_{k,j}^T\, S_{k,j} + W + M_{k,j}^T\, R_1 M_{k,j} && (3.22)\\ M_{k,j+1} &= R_1^{-1} G^T S_{k,j} && (3.23) \end{aligned}$$

where $F_{k,j} = F_k - GM_{k,j}$, and we assume there exists $M_{k,0}$ such that $F_{k,0}$ is Hurwitz.

Now, along the solutions of (3.21), we have

$$\begin{aligned} \hat{\xi}^T S_{k,j} \hat{\xi}\Big|_t^{t+T} &= -\int_t^{t+T} \hat{\xi}^T \left(W + M_{k,j}^T R_1 M_{k,j}\right) \hat{\xi} d\tau \\ &\quad +2\int_t^{t+T} (\hat{u} + M_{k,j}\hat{\xi})^T R_1 M_{k,j+1} \hat{\xi} d\tau \\ &\quad +2\int_t^{t+T} \hat{\xi}^T N_{k,j} x d\tau + 2\int_t^{t+T} \Delta_1^T L_{k,j} \hat{\xi} d\tau \end{aligned}$$

where $\hat{u} = u + \Delta_2$, $N_{k,j} = S_{k,j} E_k$ and $L_{k,j} = B^T K_k^T S_{k,j}$.

Then, we obtain the following linear equations that can be used to approximate the solution to the ARE (3.6).

$$\Phi_{k,j} \text{ vec}\left(\begin{bmatrix} S_{k,j} & M_{k,j+1} & N_{k,j} & L_{k,j} \end{bmatrix}\right) = \Psi_{k,j} \tag{3.24}$$

where $\Phi_{k,j} \in \mathbb{R}^{l\times m(n+m)}$ and $\Psi_{k,j} \in \mathbb{R}^{l}$ are defined as:

$$\begin{aligned}
\Phi_{k,j} &= \left[\delta_{\hat{\xi}\hat{\xi}} - 2I_{\hat{\xi}\hat{\xi}}(I_m \otimes M_{k,j}^T R_1) - 2I_{\hat{\xi}\hat{u}}(I_m \otimes R_1) - 2I_{x\hat{\xi}} - 2I_{\hat{\xi}\Delta_1}\right], \\
\Psi_{k,j} &= -I_{\hat{\xi}\hat{\xi}}\text{vec}(W_k).
\end{aligned}$$

Notice that $\delta_{\hat{\xi}\hat{\xi}}, I_{\hat{\xi}\hat{u}}, I_{\hat{\xi}\hat{\xi}}, I_{\hat{\xi}\Delta_1} \in \mathbb{R}^{l\times m^2}$, $I_{x\hat{\xi}} \in \mathbb{R}^{l\times nm}$ can be obtained by

$$\begin{aligned}
\delta_{\hat{\xi}\hat{\xi}} &= \delta_{zz} + 2\delta_{xz}\left(K_k^T \otimes I_m\right) + \delta_{xx}\left(K_k^T \otimes K_k^T\right), \\
I_{\hat{\xi}\hat{u}} &= I_{z\hat{u}} + I_{x\hat{u}}\left(K_k^T \otimes I_m\right), \\
I_{\hat{\xi}\hat{\xi}} &= I_{zz} + 2I_{xz}\left(K_k^T \otimes I_m\right) + I_{xx}\left(K_k^T \otimes K_k^T\right), \\
I_{x\hat{\xi}} &= I_{xz} + I_{xx}(I_n \otimes K_k^T), \\
I_{\hat{\xi}\Delta_1} &= I_{x\Delta_1}\left(K_k^T \otimes I_m\right) + I_{z\Delta_1}.
\end{aligned}$$

Clearly, (3.24) does not rely on the knowledge of E, F, or G.

3.3.3 Implementation issues

Similar as in previous policy-iteration-based algorithms, an initial stabilizing control policy is required in the learning phases. Here, we assume there exist an initial control policy $u_0 = -K_x x - K_z z$ and a positive definite matrix $\bar{P} = \bar{P}^T$ satisfying $\bar{P} > \bar{P}^*$, such that along the trajectories of the closed-loop system comprised of (3.1)-(3.4) and

u_0, we have

$$\frac{d}{dt}\left[X^T\bar{P}X + U(w)\right] \leq -\epsilon|X|^2 \tag{3.25}$$

where $\epsilon > 0$ is a constant.

Notice that this initial stabilizing control policy u_0 can be obtained using the idea of gain assignment [83]. In addition, to satisfy the rank condition in Lemma 3.3.1 below, additional exploration noise may need to be added into the control signal.

The robust-ADP scheme can thus be summarized in the following algorithm:

Algorithm 3.3.1 Robust-ADP algorithm

1: Apply an initial control policy $u = u_0$ to the system.
2: Let $k = 0$. Set Q, R, W, and R_1 according to Lemma 3.2.1. Select a sufficiently small constant $\epsilon' > 0$.
3: **repeat**
4: Solve P_k, K_{k+1} from (3.20). Let $k \leftarrow k + 1$.
5: **until** $|P_k - P_{k-1}| < \epsilon'$
6: Select W, R_1 according to Lemma 3.2.1. Let $j = 0$.
7: **repeat**
8: Solve $S_{k,j}$, $M_{k,j+1}$, $N_{k,j}$ and $L_{k,j}$ from (3.24). Let $j \leftarrow j + 1$.
9: **until** $|S_{k,j} - S_{k,j-1}| < \epsilon'$
10: Use

$$\tilde{u} = -\left[(M_{k,j}^T R_1)^{-1}(N_{k,j} + RK_k) + M_{k,j}K_k\right]x - M_{k,j}z \tag{3.26}$$

as the approximate control input.

3.3.4 Convergence analysis

Lemma 3.3.1. *Suppose A_k and $F_{k,j}$ are Hurwitz and there exists an integer $l_0 > 0$, such that the following holds for all $l \geq l_0$:*

$$\begin{aligned} &\operatorname{rank}\left(\begin{bmatrix} I_{xx} & I_{xz} & I_{zz} & I_{x\hat{u}} & I_{z\hat{u}} & I_{x\Delta_1} & I_{z\Delta_1} \end{bmatrix}\right) \\ = \;& \frac{n(n+1)}{2} + \frac{m(m+1)}{2} + 3mn + 2m^2. \end{aligned} \tag{3.27}$$

Then,

1. *there exist unique* $P_k = P_k^T$ *and* K_{k+1} *satisfying (3.20), and*

2. *there exist unique* $S_{k,j} = S_{k,j}^T$, $M_{k,j+1}$, $N_{k,j}$, $L_{k,j}$ *satisfying (3.24).*

Proof. The proof of 1) has been given in the previous setion, and is restated here for the readers' convenience. Actually, we only need to show that, given any constant matrices $P = P^T \in \mathbb{R}^{n\times n}$ and $K \in \mathbb{R}^{m\times n}$, if

$$\Theta_k \begin{bmatrix} \mathrm{vec}(P) \\ \mathrm{vec}(K) \end{bmatrix} = 0, \tag{3.28}$$

we will have $P = 0$ and $K = 0$.

By definition, we have

$$\Theta_k \begin{bmatrix} \mathrm{vec}(P) \\ \mathrm{vec}(K) \end{bmatrix} = I_{xx}\mathrm{vec}(Y) + 2(I_{xz} + I_{x\Delta_1})\mathrm{vec}(Z) \tag{3.29}$$

where

$$\begin{aligned} Y &= A_k^T P + P A_k + K_k^T(B^T P - RK) + (PB - K^T R)K_k, && (3.30)\\ Z &= B^T P - RK. && (3.31) \end{aligned}$$

Notice that since Y is symmetric, (3.28) and (3.29) imply

$$0 = I_x\ \nu(Y) + (I_{xz} + I_{x\Delta_1})\mathrm{vec}(2Z). \tag{3.32}$$

where

$$I_x = \begin{bmatrix} \int_{t_0}^{t_1}\mu(x)d\tau & \int_{t_1}^{t_2}\mu(x)d\tau & \cdots & \int_{t_{l-1}}^{t_l}\mu(x)d\tau \end{bmatrix}^T.$$

Under the rank condition in Lemma 3.3.1, we have

$$\begin{aligned}\operatorname{rank}\left(\begin{bmatrix} I_{xx} & I_{xz}+I_{x\Delta_1}\end{bmatrix}\right) &\geq \operatorname{rank}\left([I_{xx}, I_{xz}, I_{zz}, I_{x\hat{u}}, I_{z\hat{u}}, I_{x\Delta_1}, I_{z\Delta_1}]\right)\\ &\quad -2mn-\frac{1}{2}m(m+1)-2m^2\\ &= \frac{1}{2}n(n+1)+mn,\end{aligned}$$

which implies $\begin{bmatrix} I_x & I_{xz}+I_{x\Delta_1}\end{bmatrix}$ has full column rank. Hence, $Y = Y^T = 0$ and $Z = 0$.

Finally, since A_k is Hurwitz for each $k \in \mathbb{Z}_+$, the only matrices $P = P^T$ and K simultaneously satisfying (3.30) and (3.31) are $P = 0$ and $K = 0$.

Now we prove 2). Similarly, suppose there exist some constant matrices $S, M, L \in \mathbb{R}^{m\times m}$ with $S = S^T$, and $N \in \mathbb{R}^{m\times n}$ satisfying

$$\Phi_{k,j}\ \operatorname{vec}\left(\begin{bmatrix} S & M & N & L\end{bmatrix}\right) = 0.$$

Then, we have

$$\begin{aligned}0 &= I_{\hat{\xi}\hat{\xi}}\operatorname{vec}\left[SF_{k,j}+F_{k,j}^TS+M_{k,j}^T(G^TS-R_1M)\right.\\ &\quad \left.(SG-M^TR_1)M_{k,j}\right]+I_{\xi\hat{u}}\ 2\operatorname{vec}(G^TS-R_1M)\\ &\quad +I_{x\xi}\ 2\operatorname{vec}(SE_k-N)+I_{\hat{\xi}\Delta_1}\ 2\operatorname{vec}(B^TK_k^TS-L)\end{aligned}$$

By definition, it holds:

$$\left[I_{xx}, I_{\hat{\xi}\hat{\xi}}, I_{x\hat{\xi}}, I_{\hat{\xi}\hat{u}}, I_{x\hat{u}}, I_{x\Delta_1}, I_{\hat{\xi}\Delta_1}\right] = [I_{xx}, I_{xz}, I_{zz}, I_{x\hat{u}}, I_{z\hat{u}}, I_{x\Delta_1}, I_{z\Delta_1}]\,T_n$$

where T_n is a nonsingular matrix. Therefore,

$$\begin{aligned}
& \frac{1}{2}m(m+1)+2m^2+mn \\
\geq\ & \operatorname{rank}\left(\begin{bmatrix} I_{\hat{\xi}\hat{\xi}} & I_{\hat{\xi}\hat{u}} & I_{x\hat{\xi}} & I_{\hat{\xi}\Delta_1} \end{bmatrix}\right) \\
\geq\ & \operatorname{rank}\left(\left[I_{xx}, I_{\hat{\xi}\hat{\xi}}, I_{x\hat{\xi}}, I_{\hat{\xi}\hat{u}}, I_{x\hat{u}}, I_{x\Delta_1}, I_{\hat{\xi}\Delta_1}\right]\right) - \frac{1}{2}n(n+1) - 2mn \\
=\ & \operatorname{rank}\left([I_{xx}, I_{xz}, I_{zz}, I_{x\hat{u}}, I_{z\hat{u}}, I_{x\Delta_1}, I_{z\Delta_1}]\right) - \frac{1}{2}n(n+1) - 2mn \\
=\ & \frac{1}{2}m(m+1)+2m^2+mn.
\end{aligned}$$

Following the same reasoning from (3.29) to (3.32), we obtain

$$0 = SF_{k,j} + F_{k,j}^T S + M_{k,j}^T(G^T S - R_1 M) + (SG - M^T R_1)M_{k,j} \tag{3.33}$$

$$0 = G^T S - R_1 M, \tag{3.34}$$

$$0 = SE - N, \tag{3.35}$$

$$0 = BK_k S - L \tag{3.36}$$

where $[S, E, M, L] = 0$ is the only possible solution.

□

3.4 Application to synchronous generators

The power system considered in this chapter is an interconnection of two synchronous generators described by [97]:

$$\Delta\dot{\delta}_i = \Delta\omega_i, \tag{3.37}$$

$$\Delta\dot{\omega}_i = -\frac{D}{2H_i}\Delta\omega + \frac{\omega_0}{2H_i}\left(\Delta P_{mi} + \Delta P_{ei}\right), \tag{3.38}$$

$$\Delta\dot{P}_{mi} = \frac{1}{T_i}\left(-\Delta P_{mi} - k_i\Delta\omega_i + u_i\right), \quad i = 1, 2 \tag{3.39}$$

where, for the i-th generator, $\Delta\delta_i$, $\Delta\omega_i$, ΔP_{mi}, and ΔP_{ei} are the deviations of rotor angle, relative rotor speed, mechanical input power, and active power, respectively. The control signal u_i represents deviation of the valve opening. H_i, D_i, ω_0, k_i, and T_i are constant system parameters.

The active power ΔP_{ei} is defined as

$$\Delta P_{e1} = -\frac{E_1 E_2}{X}\left[\sin(\delta_1 - \delta_2) - \sin(\delta_{10} - \delta_{20})\right] \tag{3.40}$$

and $\Delta P_{e2} = -\Delta P_{e1}$, where δ_{10} and δ_{20} are the steady state angles of the first and second generators. The second synchronous generator is treated as the dynamic uncertainty, and it has a fixed controller $u_2 = -a_1\Delta\delta_2 - a_2\Delta\omega_2 - a_3\Delta P_{m2}$, with a_1, a_2, and a_3 its feedback gains.

Our goal is to design a robust optimal control policy u_1 for the interconnected power system. For simulation purpose, the parameters are specified as follows: $D_1 = 1$, $H_1 = 3$, $\omega_0 = 314.159$ rad/s, $T_1 = 5s$, $\delta_{10} = 2$ rad, $D_2 = 1$, $T_2 = 5$, $X = 15$, $k_2 = 0$, $H_2 = 3$, $a_1 = 0.2236$, $a_2 = -0.2487$, $a_3 = -7.8992$. Weighting matrices are $Q = \text{block diag}(5, 0.0001)$, $R = 1$, $W = 0.01$, and $R_1 = 100$. The exploration noise we employed for this simulation is the sum of sinusoidal functions with different frequencies.

In the simulation, two generators were operated on their steady states from $t = 0s$ to $t = 1s$. An impulse disturbance on the load was simulated at $t = 1s$, and the overall system started to oscillated. The RADP algorithm was applied to the first generator from $t = 2s$ to $t = 3s$. Convergence is attained after six iterations of phase-one learning followed by ten iterations of phase-two learning, when the stopping criterions $|P_k - P_{k-1}| \leq 10^{-6}$ and $|S_{k,j} - S_{k,j-1}| \leq 10^{-6}$ are both satisfied. The linear control policy formulated after the RADP algorithm is as follows:

$$\tilde{u}_1 = -256.9324\Delta\delta_1 - 44.4652\Delta\omega_1 - 153.1976\Delta P_{m1}.$$

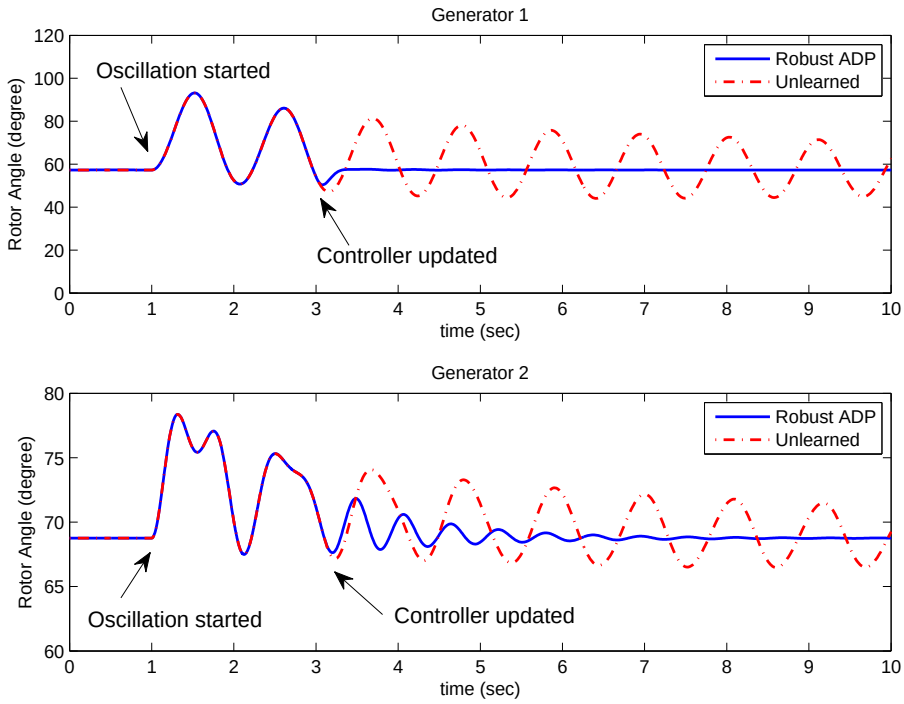

Figure 3.1: Trajectories of the rotor angle.

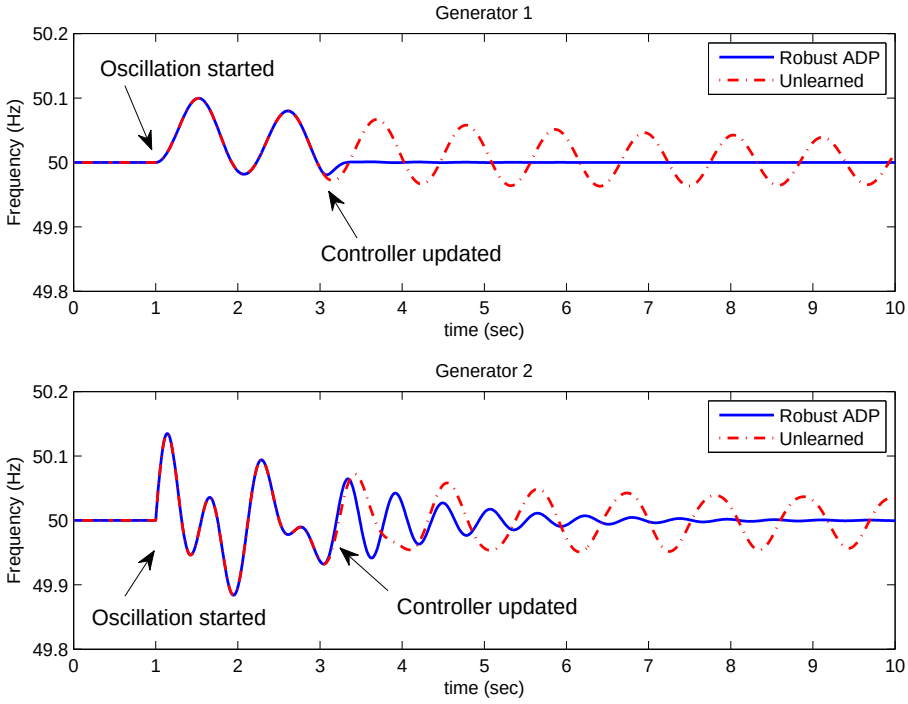

Figure 3.2: Trajectories of the angular velocity.

The ideal robust optimal control policy is given for comparison as follows

$$u_1^* = -259.9324\Delta\delta_1 - 44.1761\Delta\omega_1 - 153.1983\Delta P_{m1}.$$

Trajectories of the output variables and convergence of the feedback gain matrices are shown in Figures 3.1-3.2.

The new control policy for Generator 1 is applied from $t = 3s$ to the end of the simulation. It can be seen that oscillation has been significantly reduced after RADP-based online learning.

3.5 Conclusions

In this chapter, we have proposed a framework of RADP to compute globally asymptotically stabilizing control policies with suboptimality and robustness properties in the presence of dynamic uncertainties. A learning algorithm is provided for the online computation of partial-state feedback control laws. The novel control scheme is developed by integration of ADP, and some tools developed within the nonlinear control community. Different from previous ADP schemes in the past literature, the RADP framework can handle systems with dynamic uncertainties of which the state variables and the system order are not precisely known. As an illustrative example, the proposed algorithm has been applied to the robust optimal control for a two-machine power system. In Chapter 4, the results developed here will be extended to study multi-machine power systems.

Chapter 4

RADP for large-scale systems

The development of intelligent online learning controller gains remarkable popularity in the operation of large-scale complex systems, such as power systems. In recent years, considerable attention has been paid to the stabilization of large-scale complex systems [78], [116], [135], [145], [175], as well as the related consensus and synchronization problems [24], [105], [146], [200]. Examples of large-scale systems arise from ecosystems, transportation networks, and power systems, to name only a few, [49], [104], [110], [132], [180], [206]. Often, in real-world applications, precise mathematical models are hard to build and the model mismatches, caused by parametric and dynamic uncertainties, are thus unavoidable. This, together with the exchange of only local system information, makes the design problem extremely challenging in the context of complex networks.

In this chapter, we intend to extend the RADP theories in Chapters 2 and 3 for decentralized optimal control of multimachine power systems, and a more generalized a class of large-scale uncertain systems. The controller design for each subsystem only needs to utilize local state variables without knowing the system dynamics. By integrating a simple version of the cyclic-small-gain theorem [109], asymptotic stability can be achieved by assigning appropriate weighting matrices for each subsystem.

As a by-product, certain suboptimality properties can be obtained.

This chapter is organized as follows. Section 4.1 studies the global and robust optimal stabilization of a class of large-scale uncertain systems. Section 4.2 develops the robust ADP scheme for large-scale systems. Section 4.3 presents a novel solution to decentralized stabilization based on the proposed methodology. It is our belief that the proposed design methodology will find wide applications in large-scale systems. Finally, Section 4.4 gives some brief concluding remarks.

4.1 Stability and optimality for large-scale systems

In this section, we first describe the class of large-scale uncertain systems to be studied. Then, we present our novel decentralized optimal controller design scheme. It will also be shown that the closed-loop interconnected system enjoys some suboptimality properties.

4.1.1 Description of large-scale systems

Consider the complex large-scale system of which the i-subsystem ($1 \leq i \leq N$) is described by

$$\dot{x}_i = A_i x_i + B_i \left[u_i + \Psi_i(y)\right], \quad y_i = C_i x_i, \quad 1 \leq i \leq N \tag{4.1}$$

where $x_i \in \mathbb{R}^{n_i}$, $y_i \in \mathbb{R}^{q_i}$, and $u_i \in \mathbb{R}^{m_i}$ are the state, the output and the control input for the i-th subsystem; $y = \left[y_1^T, y_2^T, \cdots, y_N^T\right]^T$; $A_i \in \mathbb{R}^{n_i \times n_i}$, $B_i \in \mathbb{R}^{n_i \times m_i}$ are unknown system matrices. $\Psi_i(\cdot) : \mathbb{R}^q \to \mathbb{R}^{m_i}$ are unknown interconnections satisfying $|\Psi_i(y)| \leq d_i|y|$ for all $y \in \mathbb{R}^q$, with $d_i > 0$, $\sum_{i=1}^N n_i = n$, $\sum_{i=1}^N q_i = q$, and $\sum_{i=1}^N m_i = m$. It is also assumed that (A_i, B_i) is a stabilizable pair, that is, there exists a constant matrix K_i such that $A_i - B_i K_i$ is a stable matrix.

Notice that the decoupled system can be written in a compact form:

$$\dot{x} = A_D x + B_D u \tag{4.2}$$

where $x = \left[x_1^T, x_2^T, \cdots, x_N^T\right]^T \in \mathbb{R}^n$, $u = \left[u_1^T, u_2^T, \cdots, u_N^T\right]^T \in \mathbb{R}^m$, A_D =block diag(A_1, A_2, $\cdots$, A_N) $\in \mathbb{R}^{n\times n}$, B_D =block diag(B_1, B_2,$\cdots$,B_N) $\in \mathbb{R}^{n\times m}$.

For system (4.2), we define the following quadratic cost

$$J_D = \int_0^\infty \left(x^T Q_D x + u^T R_D u\right)\ d\tau \tag{4.3}$$

where Q_D =block diag(Q_1,Q_2,$\cdots$,Q_N) $\in \mathbb{R}^{n\times n}$, R_D =block diag(R_1,R_2,$\cdots$, R_N) $\in \mathbb{R}^{m\times m}$, $Q_i \in \mathbb{R}^{n_i\times n_i}$, and $R_i \in \mathbb{R}^{m_i\times m_i}$, with $Q_i = Q_i^T \geq 0$, $R_i = R_i^T > 0$, and $(A_i, Q_i^{1/2})$ observable, for all $1 \leq i \leq N$.

By linear optimal control theory [99], a minimum cost $J_D^{\odot}$ in (4.3) can be obtained by employing the following decentralized control policy

$$u_D^{\odot} = -K_D x \tag{4.4}$$

where K_D = block diag($K_1, K_2, \cdots, K_N$) is given by

$$K_D = R_D^{-1} B_D^T P_D \tag{4.5}$$

and P_D = block diag($P_1, P_2, \cdots, P_N$) is the unique symmetric positive definite solution of the algebraic Riccati equation

$$A_D^T P_D + P_D A_D - P_D B_D R_D^{-1} B_D^T P_D + Q_D = 0. \tag{4.6}$$

4.1.2 Decentralized stabilization

Now, we analyze the stability of the closed-loop system comprised of 4.1 and the decentralized controller (4.4). We show that by selecting appropriate weighting matrices Q_D and R_D, global asymptotic stability can be achieved for the large-scale closed-loop system.

To begin with, we give two lemmas.

Lemma 4.1.1. *For any $\gamma_i > 0$ and $\epsilon_i > 0$, let $u^{\odot}$ be the decentralized control policy obtained from (4.4)-(4.6) with $Q_i \geq (\gamma_i{}^{-1} + 1)\, C_i^T C_i + \gamma_i{}^{-1}\epsilon_i I_{n_i}$ and $R_i^{-1} \geq d_i^2 I_{m_i}$. Then, along the solutions of the closed-loop system consisting of* (4.1) *and (4.4), we have*

$$\frac{d}{dt}\left(x_i^T \gamma_i P_i x_i\right) \leq -|y_i|^2 - \epsilon_i |x_i|^2 + \gamma_i \sum_{j=1, j\neq i}^{N} |y_j|^2. \tag{4.7}$$

Proof. Along the solutions of the closed-loop system, we have

$$
\begin{aligned}
& \frac{d}{dt}\left(x_i^T P_i x_i\right) \\
= \;& x_i^T P_i \left[A_i x_i + B_i u_i + B_i \Psi_i(y)\right] + \left[A_i x_i + B_i u_i + B_i \Psi_i(y)\right]^T P_i x_i \\
= \;& x_i^T P_i \left[A_i x_i - B_i K_i x_i + B_i \Psi_i(y)\right] + \left[A_i x_i - B_i K_i x_i + B_i \Psi_i(y)\right]^T P_i x_i \\
= \;& x_i^T P_i (A_i - B_i K_i) x_i + x_i^T (A_i - B_i K_i)^T P_i x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i \\
= \;& x_i^T \left[P_i (A_i - B_i K_i) + (A_i - B_i K_i)^T P_i\right] x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i \\
= \;& x_i^T \left[-Q_i - P_i B_i R_i^{-1} B_i^T P_i\right] x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i \\
= \;& -x_i^T Q_i x_i - x_i^T P_i B_i R_i^{-1} B_i^T P_i x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i \\
\leq \;& -x_i^T Q_i x_i - d_i^2 x_i^T P_i B_i B_i^T P_i x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i \\
= \;& -x_i^T Q_i x_i + d_i^{-2} \Psi_i^T(y) \Psi_i(y) \\
& - d_i^2 x_i^T P_i B_i B_i^T P_i x_i + x_i^T P_i B_i \Psi_i(y) + \Psi_i^T(y) B_i^T P_i x_i - d_i^{-2} \Psi_i^T(y) \Psi_i(y) \\
= \;& -x_i^T Q_i x_i + d_i^{-2} \Psi_i^T(y) \Psi_i(y) \\
& - \left[d_i B_i^T P_i x_i - d_i^{-1} \Psi_i(y)\right]^T \left[d_i B_i^T P_i x_i - d_i^{-1} \Psi_i(y)\right] \\
\leq \;& -x_i^T Q_i x_i + d_i^{-2} \Psi_i^T(y) \Psi_i(y) \\
\leq \;& -x_i^T \left[\left({\gamma_i}^{-1} + 1\right) C_i^T C_i + {\gamma_i}^{-1} \epsilon_i I_{n_i}\right] x_i + d_i^{-2} \Psi_i^T(y) \Psi_i(y) \\
\leq \;& -\left({\gamma_i}^{-1} + 1\right) |y_i|^2 - {\gamma_i}^{-1} \epsilon_i |x_i|^2 + |y|^2 \\
\leq \;& -{\gamma_i}^{-1} |y_i|^2 - {\gamma_i}^{-1} \epsilon_i |x_i|^2 + \sum_{j=1, j \neq i}^{N} |y_j|^2.
\end{aligned}
$$

Therefore,

$$
\frac{d}{dt}\left(\gamma_i x_i^T P_i x_i\right) \leq -|y_i|^2 - \epsilon_i |x_i|^2 + \gamma_i \sum_{j=1, j \neq i}^{N} |y_j|^2.
$$

The proof is complete. □

Lemma 4.1.2. *Under the conditions of Lemma 4.1.1, suppose the following cyclic-*

small-gain condition holds

$$\sum_{j=1}^{N-1} j \sum_{1\le i_1<i_2<\cdots<i_{j+1}\le j+1} \gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}} < 1. \tag{4.8}$$

Then, there exist constants $c_i > 0$ for all $1 \le i \le N$, such that along the solutions of the closed-loop system (4.1) *and (4.4), we have*

$$\frac{d}{dt}\left(\sum_{i=1}^{N} x_i^T c_i\gamma_i P_i x_i\right) \le -|y|^2 - \sum_{j=1}^{N} c_i\gamma_i\epsilon_i|x_i|^2. \tag{4.9}$$

Proof. To begin with, let us consider the following linear equations

$$\begin{bmatrix} -1 & \gamma_2 & \gamma_3 & \cdots & \gamma_N \\ \gamma_1 & -1 & \gamma_3 & \cdots & \gamma_N \\ \gamma_1 & \gamma_2 & -1 & \ddots & \gamma_N \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \gamma_1 & \gamma_2 & \gamma_3 & \cdots & -1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \\ \vdots \\ -1 \end{bmatrix} \tag{4.10}$$

First, we show that, if the cyclic-small-gain condition (4.8) holds, the equation (4.10) can be solved as

$$c_i = \frac{\prod_{j=1,j\neq i}^{N}(\gamma_j+1)}{1-\sum_{j=1}^{N-1} j\sum_{1\le i_1<i_2<\cdots<i_{j+1}\le j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}}} > 0. \tag{4.11}$$

Indeed, it can be proved by mathematical induction: 1) if $N = 2$, (4.10) is reduced to

$$\begin{bmatrix} -1 & \gamma_2 \\ \gamma_1 & -1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}. \tag{4.12}$$

and the solution is $c_1 = \dfrac{1+\gamma_2}{1-\gamma_1\gamma_2}$, $c_2 = \dfrac{1+\gamma_1}{1-\gamma_1\gamma_2}$. Notice that the solution is unique,

because the cyclic-small-gain condition (4.8) guarantees that the determinant of the coefficient matrix is non-zero.

2) Suppose (4.11) is the solution of (4.10) with $N = N' - 1$, we show it is also valid for $N = N'$. Then, from the first row of (4.10), we have

$$\begin{aligned} & -\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)+\sum_{i=2}^{N'-1}\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)\gamma_i \\ = & \sum_{j=1}^{N'-2} j \sum_{1\leq i_1<i_2<\cdots<i_{j+1}\leq j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}}-1 \end{aligned} \tag{4.13}$$

Now,

$$\begin{aligned} & -\prod_{j=1,j\neq i}^{N'}(\gamma_j+1)+\sum_{i=2}^{N'}\prod_{j=1,j\neq i}^{N'}(\gamma_j+1)\gamma_i \\ = & -\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)(\gamma_{N'}+1)+\sum_{i=2}^{N'-1}\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)(\gamma_{N'}+1)\gamma_i+\prod_{j=1}^{N'-1}(\gamma_j+1)\gamma_{N'} \\ = & \left[-\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)+\sum_{i=2}^{N'-1}\prod_{j=1,j\neq i}^{N'-1}(\gamma_j+1)\gamma_i\right](\gamma_{N'}+1)+\prod_{j=1}^{N'-1}(\gamma_j+1)\gamma_{N'} \\ = & \left[\sum_{j=1}^{N'-2} j \sum_{1\leq i_1<i_2<\cdots<i_{j+1}\leq j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}}-1\right](\gamma_{N'}+1)+\prod_{j=1}^{N'-1}(\gamma_j+1)\gamma_{N'} \\ = & \sum_{j=1}^{N'-1} j \sum_{2\leq i_1<i_2<\cdots<i_{j+1}\leq j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}}+\sum_{j=1}^{N'-2} j \sum_{1\leq i_1<i_2<\cdots<i_{j+1}\leq j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}} \\ & +\prod_{j=1}^{N'-1}(\gamma_j+1)\gamma_{N'}-\gamma_{N'}-1 \\ = & \sum_{j=1}^{N'-1} j \sum_{1\leq i_1<i_2<\cdots<i_{j+1}\leq j+1}\gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_{j+1}}-1. \end{aligned}$$

This implies, with $N = N'$, the first row of (4.10) is valid with the solution (4.11). Same derivations can be applied to the rest rows.

Together with Lemma 2.1 we obtain

$$
\begin{aligned}
& \frac{d}{dt}\left(\sum_{i=1}^{N} x_i^T c_i \gamma_i P_i x_i\right) \\
\leq\ & -\sum_{i=1}^{N} c_i \gamma_i \epsilon_i |x_i|^2 + \sum_{i=1}^{N} c_i \left(-|y_i|^2 + \gamma_i \sum_{j=1, j\neq i}^{N} |y_j|^2\right) \\
\leq\ & -\sum_{i=1}^{N} c_i \gamma_i \epsilon_i |x_i|^2 + \begin{bmatrix} |y_1|^2 \\ |y_2|^2 \\ |y_3|^2 \\ \cdots \\ |y_N|^2 \end{bmatrix}^T \begin{bmatrix} -1 & \gamma_2 & \gamma_3 & \cdots & \gamma_N \\ \gamma_1 & -1 & \gamma_3 & \cdots & \gamma_N \\ \gamma_1 & \gamma_2 & -1 & \ddots & \gamma_N \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \gamma_1 & \gamma_2 & \gamma_3 & \cdots & -1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_N \end{bmatrix} \\
=\ & -\sum_{j=1}^{N} c_i \gamma_i \epsilon_i |x_i|^2 - |y|^2.
\end{aligned}
$$

The proof is complete. □

In summary, we obtain the following theorem:

Theorem 4.1.1. *The overall closed-loop system* (4.1), (4.4) *is globally asymptotically stable if the cyclic-small-gain condition (4.8) holds.*

Proof. Define the Lyapunov candidate

$$V_N = \sum_{i=1}^{N} x_i^T c_i \gamma_i P_i x_i. \tag{4.14}$$

By Lemma 4.1.2, along the solutions of (4.1), it follows

$$\dot{V}_N \leq -\sum_{j=1}^{N} c_i \gamma_i \epsilon_i |x_i|^2 - |y|^2. \tag{4.15}$$

Hence, the closed-loop system is globally asymptotically stable. □

Remark 4.1.1. *It is of interest to note that a more generalized cyclic-small-gain condition based on the notion of input-to-output stability [83], [150] can be found in [109].*

4.1.3 Suboptimality analysis

Suppose $\Psi_i(\cdot)$ is differentiable at the origin for all $1 \leq i \leq N$, system (4.1) can be linearized around the origin as

$$\dot{x} = A_D x + B_D u + A_C x, \quad y = C_D x. \tag{4.16}$$

Notice that under the decentralized control policy (4.4), the cost (4.3) yields a minimum cost value $J_D^{\oplus}$ for the coupled system (4.16), which may differ from $J_D^{\odot}$. In order to study the relationship between $J_D^{\oplus}$ and $J_D^{\odot}$, define

$$M_D = \text{blockdiag}(\sigma_1 I_{n_1}, \sigma_2 I_{n_2}, \cdots, \sigma_N I_{n_N}) \in \mathbb{R}^{n \times n} \tag{4.17}$$

where $\sigma_i > 0$ with $1 \leq i \leq N$.

To quantify the suboptimality of the closed-loop system composed of (4.16) and (4.4), we recall the following concept and theorem from [175]:

Definition 4.1.1 ([175])**.** *The decentralized control law (4.4) is said to be suboptimal for system* (4.1)*, if there exists a positive number σ such that*

$$J_D^{\oplus} \leq \sigma^{-1} J_D^{\odot}. \tag{4.18}$$

Theorem 4.1.2 ([175])**.** *Suppose there exists a matrix M_D as defined in (4.17) such that the matrix*

$$F(M_D) = A_C^T M_D^{-1} P_D + M_D^{-1} P_D A_C + (I - M_D^{-1})(Q_D + K_D^T R_D K_D) \tag{4.19}$$

satisfy $F(M_D) \leq 0$. *Then, the control* $u_D^{\odot}$ *is suboptimal for (4.16) with degree*

$$\sigma = \min_{1 \leq i \leq N} \{\sigma_i\}. \tag{4.20}$$

The following theorem summarizes the suboptimality of the controller (4.4) under the cyclic-small-gain condition 4.8.

Theorem 4.1.3. *The decentralized controller* $u_D^{\odot}$ *is suboptimal for system* (4.1) *with degree*

$$\sigma = \min_{1 \leq i \leq N} \left\{ \frac{1}{c_i \gamma_i} \min_{1 \leq i \leq N} \left(\frac{c_i \gamma_i}{\gamma_i \epsilon_i^{-1} \lambda_M + 1}, 1 \right) \right\} \tag{4.21}$$

if the condition (4.8) holds.

Proof. Let $\sigma_i^{-1} = \alpha c_i \gamma_i$ with $\alpha > \dfrac{1}{\min\limits_{1 \leq i \leq N} (c_i \gamma_i)}$ and $\alpha \geq 1$.

Then, by (4.19) we obtain

$$\begin{aligned}
\frac{d}{dt}\left(x^T M_D^{-1} P_D x\right) &= x^T \left(M_D^{-1} P_D A_D + A_D M_D^{-1} P_D + M_D^{-1} P_D A_C + A_C^T M_D^{-1} P_D\right) x \\
&= x^T \left(-M_D^{-1} Q_D - M_D^{-1} K_D^T R_D K_D + M_D^{-1} P_D A_C + A_C^T M_D^{-1} P_D\right) x \\
&= x^T \left[F(M_D) - Q_D - K_D^T R_D K_D\right] x
\end{aligned}$$

Therefore, by Lemma 4.1.2, it follows that

$$\begin{aligned} x^T F(M_D)x &= \frac{d}{dt}\left(x^T M_D^{-1} P_D x\right) + x^T\left(Q_D + K_D^T R_D K_D\right)x \\ &\leq \sum_{i=1}^{N} x_i^T\left[-\sigma_i^{-1}(Q_i - C_i^T C_i) - \alpha|y_i|^2 + Q_i + K_i^T R_i K_i\right] x_i \\ &= -\sum_{i=1}^{N} x_i^T\left[\sigma_i^{-1}(Q_i - C_i^T C_i) + \alpha C_i^T C_i - Q_i - K_i^T R_i K_i\right] x_i \\ &= -\sum_{i=1}^{N} x_i^T\left[\left(\sigma_i^{-1} - 1\right)(Q_i - C_i^T C_i) - K_i^T R_i K_i\right] x_i \\ &\leq -\sum_{i=1}^{N}\left[\left(\alpha c_i \gamma_i - 1\right)\frac{\epsilon_i}{\gamma_i} - \lambda_M\right]|x_i|^2 \end{aligned}$$

where λ_M denotes the maximal eigenvalue of $K_i^T R_i K_i$.

Notice that $F(M_D) \leq 0$, if we set

$$\alpha = \max_{1\leq i\leq N}\left(\frac{\gamma_i \epsilon_i^{-1}\lambda_M + 1}{c_i\gamma_i}, 1\right) \tag{4.22}$$

Therefore, we obtain

$$\sigma = \min_{1\leq i\leq N}\left\{\frac{1}{c_i\gamma_i}\min_{1\leq i\leq N}\left(\frac{c_i\gamma_i}{\gamma_i\epsilon_i^{-1}\lambda_M + 1}, 1\right)\right\} \tag{4.23}$$

The proof is complete by Theorem 4.1.2. □

4.2 The RADP design for large-scale systems

Consider the following algebraic Riccati equation

$$A_i^T P_i + P_i A_i + Q_i - P_i B_i R_i^{-1} B_i^T P_i = 0, \quad 1 \leq i \leq N. \tag{4.24}$$

It has been shown in [89] that, given $K_i^{(0)}$ such that $A_i - B_i K_i^{(0)}$ is Hurwitz,

sequences $\{P_i^{(k)}\}$ and $\{K_i^{(k)}\}$ uniquely determined by

$$\begin{aligned} 0 &= (A_i^{(k)})^T P_i^{(k)} + P_i A_i^{(k)} + Q_i^{(k)}, \qquad (4.25) \\ K_i^{(k+1)} &= R_i^{-1} B_i^T P_i^{(k)} \qquad (4.26) \end{aligned}$$

with $A_i^{(k)} = A_i - B_i K_i^{(k)}$, and $Q_i^{(k)} = Q_i + (K_i^{(k)})^T R_i K_i^{(k)}$, have the properties that $\lim_{k\to\infty} P_i^{(k)} = P_i$, $\lim_{k\to\infty} K_i^{(k)} = K_i = R_i^{-1} B_i^T P_i$, and $A_i^{(k)}$ is Hurwitz for all $k \in \mathbb{Z}_+$.

For the i-th subsystem, along the solutions of (4.1), it follows that

$$\begin{aligned} x_i^T P_i^{(k)} x_i \Big|_t^{t+\delta t} &= 2\int_t^{t+\delta t} (\hat{u}_i + K_i^{(k)} x_i)^T R_i K_i^{(k+1)} x_i d\tau \\ &\quad - \int_t^{t+\delta t} x_i^T Q_i^{(k)} x_i \, d\tau \qquad (4.27) \end{aligned}$$

where $\hat{u}_i = u_i + \Psi_i(y)$.

For sufficiently large integer $l_i \geq 0$, define $\delta_{xx}^i \in \mathbb{R}^{l_i \times \frac{1}{2} n_i (n_i+1)}$, $I_{xx}^i \in \mathbb{R}^{l_i \times n_i^2}$, and $I_{xu}^i \in \mathbb{R}^{l_i \times m_i n_i}$ as follows

$$\begin{aligned} \delta_{xx}^i &= \left[\begin{array}{cccc} \mu(x_i)|_{t_{0,i}}^{t_{1,i}} & \mu(x_i)|_{t_{1,i}}^{t_{2,i}} & \cdots & \mu(x_i)|_{t_{l_i-1,i}}^{t_{l_i,i}} \end{array} \right]^T, \\ I_{xx}^i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} x_i \otimes x_i d\tau & \int_{t_{1,i}}^{t_{2,i}} x_i \otimes x_i d\tau & \cdots & \int_{t_{l_i-1,i}}^{t_{l_i,i}} x_i \otimes x_i d\tau \end{array} \right]^T, \\ I_{xu}^i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} x_i \otimes \hat{u}_i d\tau & \int_{t_{1,i}}^{t_{2,i}} x_i \otimes \hat{u}_i d\tau & \cdots & \int_{t_{l_i-1,i}}^{t_{l_i,i}} x_i \otimes \hat{u}_i d\tau \end{array} \right]^T \end{aligned}$$

where $\mu(x_i) \in \mathbb{R}^{\frac{1}{2} n_i (n_i+1)}$ is defined as

$$\mu(x_i) = \left[x_{i,1}^2, x_{i,1}x_{i,2}, \cdots, x_{i,1}x_{i,n_i}, x_{i,2}^2, x_{i,2}x_{i,3}, \cdots, x_{i,n_i-1}x_{i,n_i}, x_{i,n_i}^2 \right]^T$$

and $0 \leq t_{0,i} < t_{1,i} < \cdots < t_{l_i,i}$, for $i = 1, 2, \cdots, N$. Also, for any symmetric matrix

$P \in \mathbb{R}^{n_i \times n_i}$, we define $\nu(P) \in \mathbb{R}^{\frac{1}{2}n_i(n_i+1)}$ such that

$$\nu(P) = [p_{11}, 2p_{12}, \cdots, 2p_{1n_i}, p_{22}, 2p_{23}, \cdots, 2p_{n_i-1,n_i}, p_{n_i,n_i}]^T.$$

Then, (7.59) implies the following matrix form of linear equations

$$\Theta_i^{(k)} \begin{bmatrix} \nu(P_i^{(k)}) \\ \mathrm{vec}(K_i^{(k+1)}) \end{bmatrix} = \Xi_i^{(k)} \tag{4.28}$$

where the matrices $\Theta_i^{(k)} \in \mathbb{R}^{l_i \times \frac{1}{2}n_i(n_i+1)+n_i m_i}$ and $\Xi_i^{(k)} \in \mathbb{R}^{l_i}$ are defined as

$$\begin{aligned} \Theta_i^{(k)} &= \begin{bmatrix} \delta_{xx}^i & -2I_{xx}^i(I_{n_i} \otimes (K_i^{(k)})^T R_i) - 2I_{xu}^i(I_{n_i} \otimes R_i) \end{bmatrix}, \\ \Xi_i^{(k)} &= -I_{xx}^i \ \mathrm{vec}(Q_i^{(k)}). \end{aligned}$$

Clearly, if (4.28) has a unique solution, we are able to replace (4.25) and (4.26) by (4.28). In this way, the knowledge of both A_i and B_i is no longer needed.

Assumption 4.2.1. $\mathrm{rank}\,([I_{xx}^i,\ I_{xu}^i]) = \dfrac{n_i(n_i+1)}{2} + n_i m_i$.

Theorem 4.2.1. *Under Assumption 5.2.1, the matrices $P_i^{(k)} = (P_i^{(k)})^T$ and $K_i^{(k+1)}$ determined by (4.28) satisfy $\lim\limits_{k\to\infty} P_i^{(k)} = P_i$ and $\lim\limits_{k\to\infty} K_i^{(k)} = K_i$.*

Proof. Step 1): First of all, we show that, for each $i = 1, 2, \cdots, N$, and $k = 0, 1, \cdots$, equation (4.28) has a unique solution $(P_i^{(k)}, K_i^{(k+1)})$ with $P_i^{(k)} = (P_i^{(k)})^T$.

Notice that it amounts to show that the following linear equation

$$\Theta_i^{(k)} X_i = 0 \tag{4.29}$$

has only the trivial solution $X_i = 0$, for each $i = 1, 2, \cdots, N$, and $k = 0, 1, \cdots$

To this end, we prove by contradiction: Assume $X_i = \begin{pmatrix} Y_v^i \\ Z_v^i \end{pmatrix} \in \mathbb{R}^{\frac{1}{2}n_i(n_i+1)+m_i n_i}$

is a nonzero solution of (4.29), where $Y_v^i \in \mathbb{R}^{\frac{1}{2}n_i(n_i+1)}$ and $Z_v^i \in \mathbb{R}^{m_i n_i}$. Then, a symmetric matrix $Y_i \in \mathbb{R}^{n_i \times n_i}$ and a matrix $Z_i \in \mathbb{R}^{m_i \times n_i}$ can be uniquely determined, such that $\nu(Y_i) = Y_v^i$ and $\mathrm{vec}(Z_i) = Z_v^i$.

By (7.59), we have

$$\Theta_i^{(k)} X_i = I_{xx}^i \mathrm{vec}(M_i) + 2I_{xu}^i \mathrm{vec}(N_i) \tag{4.30}$$

where

$$\begin{aligned} M_i &= (A_i^{(k)})^T Y + Y A_i^{(k)} + (K_i^{(k)})^T (B_i^T Y_i - R_i Z_i) \\ &\quad + (Y_i B_i - Z_i^T R_i) K_i^{(k)}, \end{aligned} \tag{4.31}$$

$$N_i = B_i^T Y_i - R_i Z_i. \tag{4.32}$$

Notice that since M_i is symmetric, we have

$$I_{xx}^i \mathrm{vec}(M_i) = I_{\bar{x}}^i \nu(M_i) \tag{4.33}$$

where $I_{\bar{x}}^i \in \mathbb{R}^{l \times \frac{1}{2}n_i(n_i+1)}$ is defined as:

$$I_x^i = \left[\int_{t_{0,i}}^{t_{1,i}} \bar{x}_i d\tau, \quad \int_{t_{1,i}}^{t_{2,i}} \bar{x}_i d\tau, \quad \cdots, \quad \int_{t_{l-1,i}}^{t_{l,i}} \bar{x}_i d\tau \right]^T. \tag{4.34}$$

Then, (4.29) and (4.30) imply the following matrix form of linear equations

$$\left[\begin{array}{cc} I_x^i, & 2I_{xu}^i \end{array} \right] \left[\begin{array}{c} \nu(M_i) \\ \mathrm{vec}(N_i) \end{array} \right] = 0. \tag{4.35}$$

Under Assumption 5.2.1, we know $\left[\begin{array}{cc} I_x^i, & 2I_{xu}^i \end{array} \right]$ has full column rank. Therefore, the only solution to (4.35) is $\nu(M_i) = 0$ and $\mathrm{vec}(N_i) = 0$. As a result, we have $M_i = 0$ and $N_i = 0$.

Now, by (4.32) we know $Z_i = R_i^{-1} B_i^T Y_i$, and (4.31) is reduced to the following Lyapunov equation

$$(A_i^{(k)})^T Y_i + Y_i A_i^{(k)} = 0. \tag{4.36}$$

Since $A_i^{(k)}$ is Hurwitz for all $k \in \mathbb{Z}_+$, the only solution to (4.36) is $Y_i = 0$. Finally, by (4.32) we have $Z_i = 0$. Also, we have $X_i = 0$. But it contradicts with the assumption that $X_i \neq 0$. Therefore, $\Theta_i^{(k)}$ must have full column rank for all $k \in \mathbb{Z}_+$.

Step 2): Given a stabilizing feedback gain matrix $K_i^{(k)}$, if $P_i^{(k)} = (P_i^{(k)})^T$ is the solution of (4.25), $K_i^{(k+1)}$ is uniquely determined by $K_i^{(k+1)} = R_i^{-1} B_i^T P_i^{(k)}$. By (7.59), we know that $P_i^{(k)}$ and $K_i^{(k+1)}$ satisfy (4.28). On the other hand, let $P = P^T \in \mathbb{R}^{n_i \times n_i}$ and $K \in \mathbb{R}_i^{m_i \times n_i}$, such that

$$\Theta_i^{(k)} \begin{bmatrix} \nu(P) \\ \mathrm{vec}(K) \end{bmatrix} = \Xi_i^{(k)}.$$

Then, we immediately have $\nu(P) = \nu(P)_i^{(k)}$ and $\mathrm{vec}(K) = \mathrm{vec}(K_i^{(k+1)})$. By Step 1), $P = P^T$ and K are unique. In addition, by the definitions of $\nu(P)$ and $\mathrm{vec}(K)$, $P_i^{(k)} = P$ and $K_i^{(k+1)} = K$ are uniquely determined.

Therefore, the policy iteration (4.28) is equivalent to (4.25) and (4.26). By Theorem in [89], the convergence is thus proved. □

In summary, we give the following robust ADP algorithm for practical online implementation. Notice that the algorithm can be implemented to each subsystem in parallel without affecting each other. The learning system implemented for each subsystem only needs to use the state and input information of the subsystem.

Remark 4.2.1. *The algorithm can be implemented to each subsystem in parallel without affecting each other. The learning system implemented for each subsystem*

Algorithm 4.2.1 Robust ADP algorithm for large-scale systems

1: Select appropriate matrices Q_i and R_i such that the condition (4.8) is satisfied. $k \leftarrow 0$.
2: For the i-th subsystem, employ $u_i = -K_i^{(0)} x_i + e_i$, with e_i the exploration noise, as the input. Record δ_{xx}^i, I_{xx}^i and I_{xu}^i until Assumption 5.2.1 is satisfied.
3: Solve $P_i^{(k)}$ and $K_i^{(k+1)}$ from (4.28).
4: Let $k \leftarrow k+1$, and repeat Step 3 until
5: $|P_i^{(k)} - P_i^{(k-1)}| \leq \epsilon$ for all $k \geq 1$, where the constant $\epsilon > 0$ can be any predefined small threshold.
6: Use $u_i = -K_i^{(k)} x_i$ as the approximated optimal control policy to the i-th subsystem.

only needs to use the state and input information of the subsystem.

4.3 Application to a ten machine power system

4.3.1 System model

Consider the classical multimachine power system with governor controllers [97]

$$\dot{\delta}_i(t) = \omega_i(t), \tag{4.37}$$

$$\dot{\omega}_i(t) = -\frac{D_i}{2H_i}\omega_i(t) + \frac{\omega_0}{2H_i}\left[P_{mi}(t) - P_{ei}(t)\right], \tag{4.38}$$

$$\dot{P}_{mi}(t) = \frac{1}{T_i}\left[-P_{mi}(t) + u_{gi}(t)\right], \tag{4.39}$$

$$P_{ei}(t) = E'_{qi}\sum_{j=1}^{N} E'_{qj}\left[B_{ij}\sin\delta_{ij}(t) + G_{ij}\cos\delta_{ij}(t)\right] \tag{4.40}$$

where $\delta_i(t)$ is the angle of the i-th generator, $\delta_{ij} = \delta_i - \delta_j$; $\omega_i(t)$ is the relative rotor speed; $P_{mi}(t)$ and $P_{ei}(t)$ are the mechanical power and the electrical power; E'_{qi} is the transient EMF in quadrature axis, and is assumed to be constant under high-gain SCR controllers; D_i, H_i, and T_i are the damping constant, the inertia constant and the governor time constant; B_{ij}, G_{ij} are constants for $1 \leq i, j \leq N$.

Similarly as in [49], system (4.37)-(4.39) can be put into the following form,

$$\begin{aligned}
\Delta\dot{\delta}_i(t) &= \Delta\omega_i(t), && (4.41)\\
\Delta\dot{\omega}_i(t) &= -\frac{D_i}{2H_i}\Delta\omega_i(t) + \frac{\omega_0}{2H_i}\Delta P_{mi}(t), && (4.42)\\
\Delta\dot{P}_{mi}(t) &= \frac{1}{T_i}\left[-\Delta P_{mi}(t) + u_i(t) - d_i(t)\right] && (4.43)
\end{aligned}$$

where

$$\begin{aligned}
\Delta\delta_i(t) &= \delta_i(t) - \delta_{i0},\\
\Delta\omega_i(t) &= \omega_i(t) - \omega_{i0},\\
\Delta P_{mi}(t) &= P_{mi}(t) - P_{ei}(t),\\
u_i(t) &= u_{gi}(t) - P_{ei}(t),\\
d_i(t) &= E'_{qi}\sum_{j=1,j\neq i}^{N} E'_{qj}\left[B_{ij}\cos\delta_{ij}(t) - G_{ij}\sin\delta_{ij}(t)\right]\left[\Delta\omega_i(t) - \Delta\omega_j(t)\right].
\end{aligned}$$

Assume there exists a constant $\beta > 0$ such that $\max_{1\leq i,j\leq N}\left[E'_{qi}E'_{qj}(|B_{ij}| + |G_{ij}|)\right] < \beta$. Then,

$$|d_i(t)| \leq (N-1)\beta\sum_{j=1}^{N}|\Delta\omega_i - \Delta\omega_j| \leq (N-1)^2\beta\sum_{j=1}^{N}|\Delta\omega_j|.$$

Therefore, the model (4.41)-(4.43) is in the form (4.1), if we define $x_i = [\Delta\delta_i(t)$ $\Delta\omega_i(t)$ $\Delta P_{ei}(t)]^T$ and $y_i = \Delta\omega_i(t)$.

4.3.2 Numerical simulation

A ten-machine power system is considered for numerical studies. In the simulation, Generator 1 is used as the reference machine. Governor controllers and ADP-based learning systems are installed on Generators 2-10.

Simulation parameters for the ten-machine power system are shown in Tables 1-3.

Also the steady state frequency is set to be $\omega_0 = 314.15$rad/s. The initial feedback policies are

$$K_i^{(0)} = \begin{bmatrix} 10 & 50 & 0 \end{bmatrix}, \quad 1 \leq i \leq 10. \tag{4.44}$$

Table 4.1: Parameters for the generators

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
$H_i(p.u.)$	∞	6.4	3	5.5	5.2	4.7	5.4	4.9	5.1	3.4
$D_i(p.u.)$	-	1	1.5	2	2.2	2.3	2.6	1.8	1.7	2.9
$T_i(s)$	-	6	6.3	4.9	6.6	5.8	5.9	5.5	5.4	5.5
$E_{qi}(p.u.)$	1	1.2	1.5	0.8	1.3	0.9	1.1	0.6	1.5	1
$\delta_{i0}(^\circ)$	0	108.86	97.4	57.3	68.75	74.48	45.84	68.75	40.11	63.03

The admittance matrices for the transmission lines are

Table 4.2: Imaginary parts of the admittance matrix

B_{ij}	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$	$j=9$	$j=10$
$i=1$	0.25	0.19	0.41	0.30	0.29	0.48	0.24	0.09	0.21	0.23
$i=2$	0.19	0.39	0.25	0.53	0.28	0.29	0.38	0.33	0.24	0.38
$i=3$	0.41	0.25	0.05	0.27	0.25	0.22	0.27	0.21	0.33	0.49
$i=4$	0.30	0.53	0.27	0.57	0.33	0.33	0.13	0.29	0.49	0.13
$i=5$	0.29	0.28	0.25	0.32	0.21	0.37	0.30	0.03	0.24	0.31
$i=6$	0.48	0.29	0.22	0.33	0.37	0.46	0.28	0.41	0.35	0.14
$i=7$	0.24	0.38	0.27	0.13	0.30	0.28	0.12	0.43	0.14	0.53
$i=8$	0.09	0.33	0.21	0.29	0.03	0.41	0.43	0.33	0.16	0.44
$i=9$	0.21	0.24	0.33	0.49	0.24	0.35	0.14	0.16	0.36	0.21
$i=10$	0.23	0.38	0.49	0.13	0.31	0.14	0.53	0.44	0.21	0.37

All the parameters, except for the operating points, are assumed to be unknown to the learning system. The weighting matrices are set to be $Q_i = 1000I_3$, $R_i = 1$, for $i = 2, 3, \cdots, 10$.

From $t = 0s$ to $t = 1s$, all the generators were operating at the steady state. At $t = 1s$, an impulse disturbance on the active power was added to the network. As a result, the power angles and frequencies started to oscillate. In order to stabilize

Table 4.3: Real parts of the admittance matrix

G_{ij}	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$	$j=9$	$j=10$
$i=1$	0.21	0.05	0.00	0.14	0.04	-0.01	0.00	0.28	-0.08	-0.00
$i=2$	0.05	0.29	0.03	0.02	0.13	-0.17	-0.00	-0.01	-0.11	-0.04
$i=3$	0.00	0.03	0.15	0.18	-0.27	-0.10	0.15	-0.05	0.23	0.28
$i=4$	0.13	0.01	0.18	0.05	-0.11	-0.07	0.22	-0.11	0.24	-0.15
$i=5$	0.04	0.13	-0.27	-0.11	0.08	-0.20	-0.07	-0.19	0.03	-0.15
$i=6$	-0.01	-0.17	-0.10	-0.07	-0.20	0.02	-0.03	0.04	0.12	-0.07
$i=7$	0.00	-0.00	0.15	0.22	-0.07	-0.03	0.04	0.06	0.06	-0.10
$i=8$	0.28	-0.01	-0.05	-0.11	-0.19	0.04	0.06	0.27	0.13	0.11
$i=9$	-0.08	-0.11	0.23	0.23	0.03	0.12	0.06	0.13	0.19	-0.17
$i=10$	-0.00	-0.04	0.28	-0.15	-0.15	-0.07	-0.10	0.11	-0.17	0.21

the system and improve its performance, the learning algorithm is conducted from $t = 4s$ to $t = 5s$. Robust ADP-based control policies for the generators are applied from $t = 5s$ to the end of the simulation. Trajectories of the angles and frequencies of each generators are shown in Figures 4.1-4.6.

4.4 Conclusions

This chapter has studied the decentralized control of large-scale complex systems with uncertain system dynamics. We have developed an RADP-based online learning method for a class of large-scale systems, and a novel decentralized controller implementation algorithm is presented. The obtained controller globally asymptotically stabilizes the large-scale system, and at the same time, preserves suboptimality properties. In addition, the effectiveness of the proposed methodology is demonstrated via its application to the online learning control of a tem-machine power system with governor controllers. The methodology can be combined with the techniques in Chapter 3 to study large-scale systems with unmatched disturbance input (see [16] for some preliminary results). It will be interesting to further applied the proposed methodology to study complex real-world power networks ([27, 25, 26, 134, 152, 201, 201]), wind conversion systems [143], and demand response management problems [29].

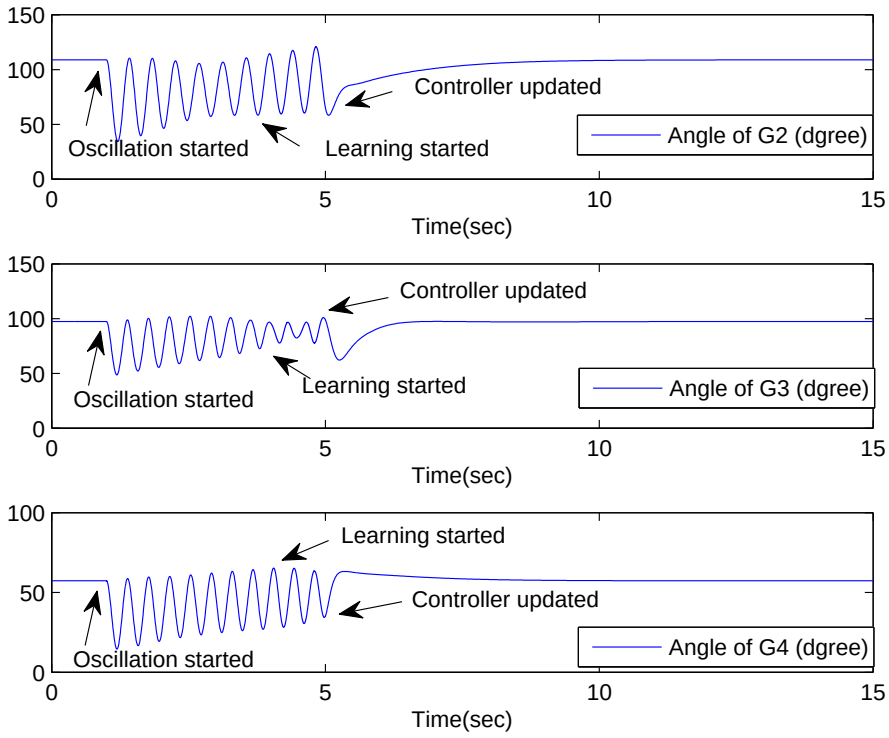

Figure 4.1: Power angle deviations of Generators 2-4.

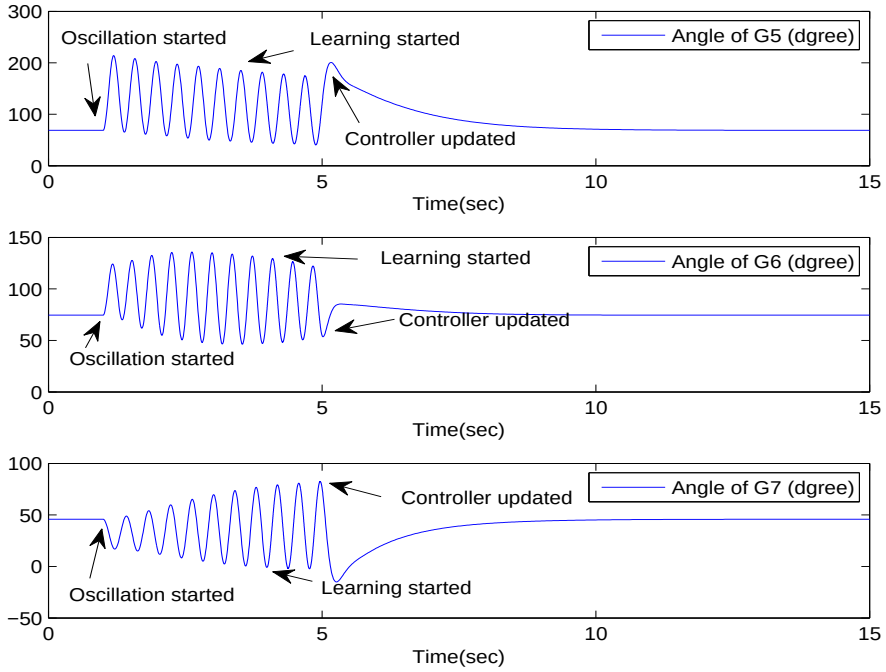

Figure 4.2: Power angle deviations of Generators 5-7.

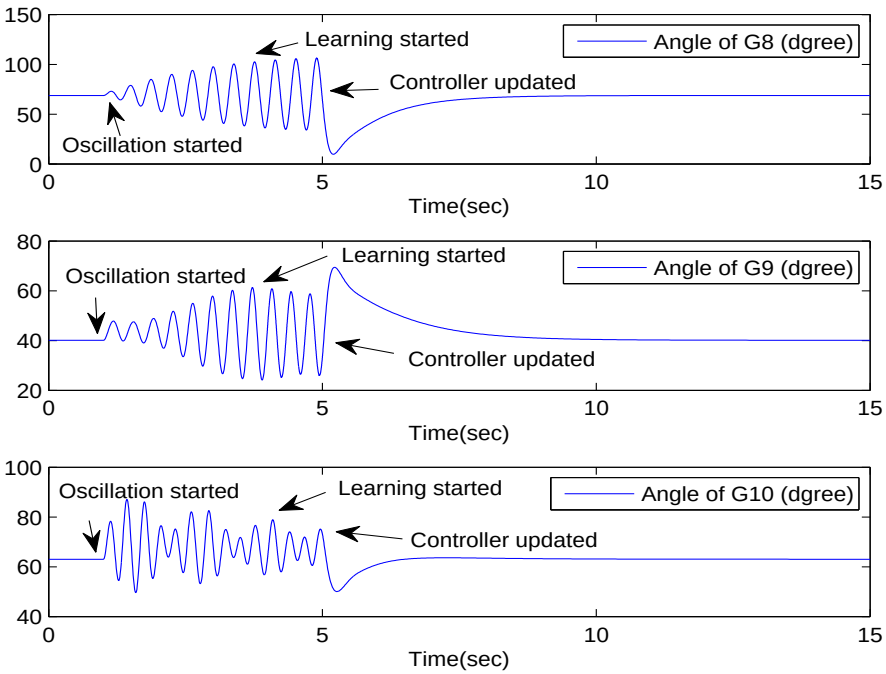

Figure 4.3: Power angle deviations of Generators 8-10.

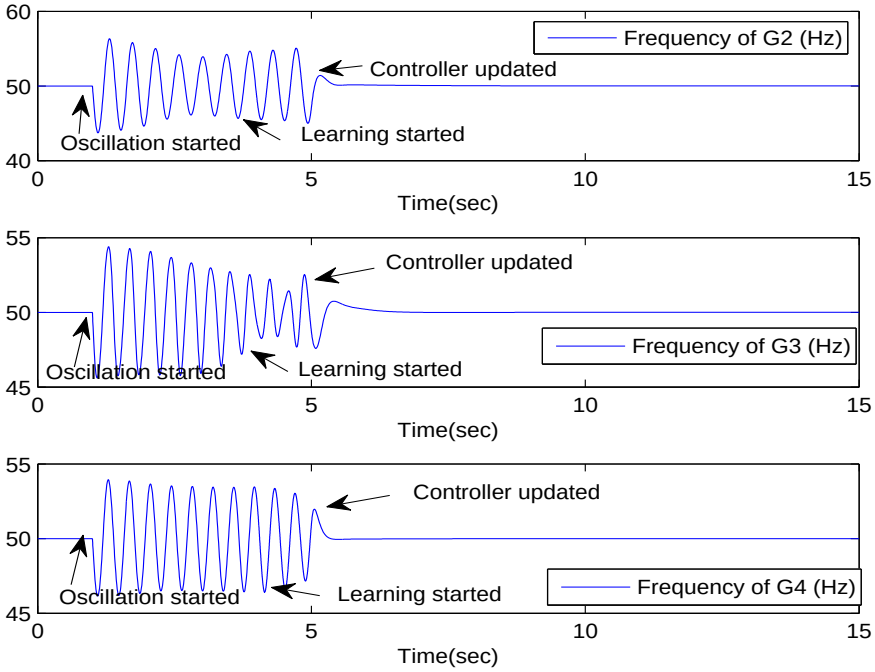

Figure 4.4: Power frequencies of Generators 2-4.

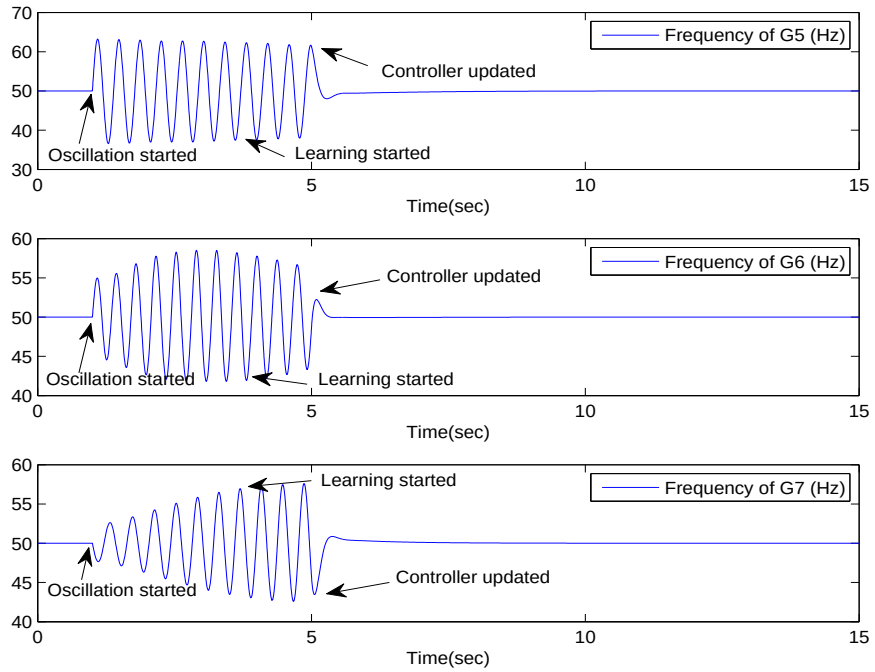

Figure 4.5: Power frequencies of Generators 5-7.

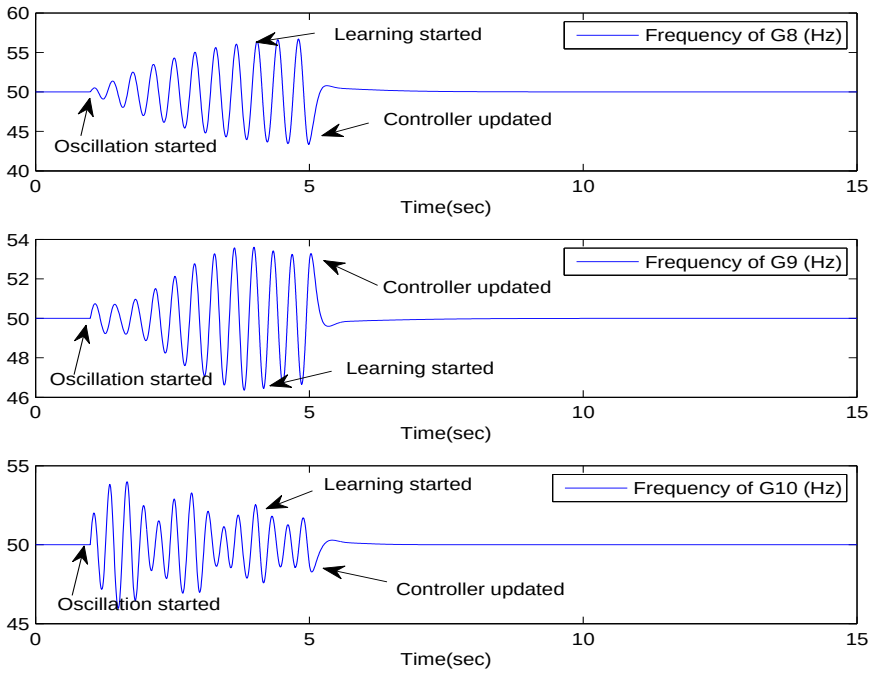

Figure 4.6: Power frequencies of Generators 8-10.

Chapter 5

Neural-networks-based RADP for nonlinear systems

As mentioned in Chapter 3, a common assumption in the past literature of ADP is that the system order is known and the state variables are either fully available or reconstructible from the output [101]. This problem, often formulated in the context of robust control theory, cannot be viewed as a special case of output feedback control. In addition, the ADP methods developed in the past literature may fail to guarantee not only optimality, but also the stability of the closed-loop system when dynamic uncertainty occurs. The RADP framework is developed to bridge the above-mentioned gap in the past literature of ADP, and it can be viewed as an extension of ADP to linear and partially linear systems [161], [164] with dynamic uncertainties.

This chapter studies RADP designs for genuinely nonlinear systems in the presence of dynamic uncertainties. We first decompose the open-loop system into two parts: The *system model* (ideal environment) with known system order and fully accessible state, and the *dynamic uncertainty*, with unknown system order, dynamics, and unmeasured states, interacting with the ideal environment. In order to handle the dynamic interaction between two systems, we resort to the gain assignment idea

[83], [129]. More specifically, we need to assign a suitable gain for the system model with disturbance in the sense of Sontag's input-to-state stability (ISS) [149], [151]. The backstepping, robust redesign, and small-gain techniques in modern nonlinear control theory are incorporated into the RADP theory, such that the system model is made ISS with an arbitrarily small gain. To perform stability analysis for the interconnected systems, we apply the nonlinear small-gain theorem [83], which has been proved to be an efficient tool for nonlinear system analysis and synthesis. As a consequence, the proposed RADP method can be seen as a nonlinear variant of [68]. Moreover, it solves the semi-global stabilization problem [161] in the sense that the domain of attraction for the closed-loop system can be made as large as possible.

The remainder of the paper is organized as follows. Section 5.1 reviews the online policy iteration technique for affine nonlinear systems. Section 5.2 studies the methodology of robust optimal design and gives a practical algorithm. Section 5.3 extends the RADP theory to nonlinear systems with unmatched dynamic uncertainties. Two numerical examples, including the controller designs for a jet engine and for a synchronous power generator, are provided in Section 5.4. Finally, concluding remarks are given in Section 5.5.

5.1 Problem formulation and preliminarlies

5.1.1 Nonlinear optimal control

Consider the system

$$\dot{x} = f(x) + g(x)u \tag{5.1}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}$ is the control input, $f, g : \mathbb{R}^n \to \mathbb{R}^n$ are locally Lipschitz functions. The to-be-minimized cost associated with (5.1) is defined

as

$$J(x_0, u) = \int_0^\infty \left[Q(x) + ru^2\right] dt, \quad x(0) = x_0 \tag{5.2}$$

where $Q(\cdot)$ is a positive definite function, and $r > 0$ is a constant. In addition, assume there exists an *admissible* control policy $u = u_0(x)$ in the sense that, under this policy, the system (5.1) is globally asymptotically stable and the cost (5.2) is finite. By [99], the control policy that minimizes the cost (5.2) can be solved from the following Hamilton-Jacobi-Bellman (HJB) equation:

$$0 = \nabla V(x)^T f(x) + Q(x) - \frac{1}{4r}\left[\nabla V(x)^T g(x)\right]^2 \tag{5.3}$$

with the boundary condition $V(0) = 0$. Indeed, if the solution $V^*(x)$ of (5.3) exists, the optimal control policy is given by

$$u^*(x) = -\frac{1}{2r} g(x)^T \nabla V^*(x). \tag{5.4}$$

In general, the analytical solution of (5.3) is difficult to be obtained. However, if $V^*(x)$ exists, it can be approximated using the policy iteration technique [136]:

Algorithm 5.1.1 Nonlinear policy iteration algorithm

1: Find an admissible control policy $u_0(x)$.
2: For any integer $i \geq 0$, solve for $V_i(x)$, with $V_i(0) = 0$, from

$$0 = \nabla V_i(x)^T \left[f(x) + g(x)u_i(x)\right] + Q(x) + ru_i(x)^2. \tag{5.5}$$

3: Update the control policy by

$$u_{i+1}(x) = -\frac{1}{2r} g(x)^T \nabla V_i(x). \tag{5.6}$$

Convergence of the policy iteration (5.5) and (5.6) is concluded in the following theorem, the proof of which follows the same lines of reasoning as in the proof of [136,

Theorem 4].

Theorem 5.1.1. *Consider $V_i(x)$ and $u_{i+1}(x)$ defined in (5.5) and (5.6). Then, for all $i = 0, 1, \cdots$,*

$$0 \leq V_{i+1}(x) \leq V_i(x), \quad \forall x \in R^n \tag{5.7}$$

and $u_{i+1}(x)$ is admissible. In addition, if the solution $V^*(x)$ of (5.3) exists, then for each fixed x, $\{V_i(x)\}_{i=0}^{\infty}$ and $\{u_i(x)\}_{i=0}^{\infty}$ converge pointwise to $V^*(x)$ and $u^*(x)$, respectively.

5.2 Online Learning via RADP

In this section, we develop the RADP methodology for nonlinear systems as follows:

$$\dot{w} = \Delta_w(w, x) \tag{5.8}$$

$$\dot{x} = f(x) + g(x)\left[u + \Delta(w, x)\right] \tag{5.9}$$

where $x \in \mathbb{R}^n$ is the measured component of the state available for feedback control, $w \in \mathbb{R}^p$ is the unmeasurable part of the state with unknown order p, $u \in \mathbb{R}$ is the control input, $\Delta_w : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^p$, $\Delta : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$ are unknown locally Lipschitz functions, f and g are defined the same as in (5.1) but are assumed to be unknown.

Our design objective is to find online the control policy which stabilizes the system at the origin. Also, in the absence of the dynamic uncertainty (i.e., $\Delta = 0$ and the w-subsystem is absent), the control policy becomes the optimal control policy that minimizes (5.2).

5.2.1 Online policy iteration

The iterative technique introduced in Section 5.1 relies on the knowledge of both $f(x)$ and $g(x)$. To remove this requirement, we develop a novel online policy iteration technique, which can be viewed as the nonlinear extension of [68].

To begin with, notice that (5.9) can be rewritten as

$$\dot{x} = f(x) + g(x)u_i(x) + g(x)v_i \tag{5.10}$$

where $v_i = u + \Delta - u_i$. For each $i \geq 0$, the time derivative of $V_i(x)$ along the solutions of (5.10) satisfies

$$\begin{aligned} \dot{V}_i &= \nabla V_i(x)^T \left[f(x) + g(x)u_i(x) + g(x)v_i\right] \\ &= -Q(x) - ru_i^2(x) + \nabla V_i(x)^T g(x)v_i \\ &= -Q(x) - ru_i^2(x) - 2ru_{i+1}(x)v_i. \end{aligned} \tag{5.11}$$

Integrating both sides of (5.11) on any time interval $[t, t+T]$, it follows that

$$\begin{aligned} & V_i(x(t+T)) - V_i(x(t)) \\ = & \int_t^{t+T} \left[-Q(x) - ru_i^2(x) - 2ru_{i+1}(x)v_i\right] d\tau. \end{aligned} \tag{5.12}$$

Notice that, if an admissible control policy $u_i(x)$ is given, the unknown functions $V_i(x)$ and $u_{i+1}(x)$ can be approximated using (5.12). To be more specific, for any given compact set $\Omega \subset \mathbb{R}^n$ containing the origin as an interior point, let $\{\phi_j(x)\}_{j=1}^{\infty}$ be an infinite sequence of linearly independent smooth basis functions on Ω, where $\phi_j(0) = 0$ for all $j = 1, 2, \cdots$. Then, by approximation theory [126], for each $i = 0, 1, \cdots$, the

cost function and the control policy can be approximated by:

$$\hat{V}_i(x) = \sum_{j=1}^{N_1} \hat{c}_{i,j}\phi_j(x), \tag{5.13}$$

$$\hat{u}_{i+1}(x) = \sum_{j=1}^{N_2} \hat{w}_{i,j}\phi_j(x). \tag{5.14}$$

where $N_1 > 0$, $N_2 > 0$ are two sufficiently large integers, and $\hat{c}_{i,j}$, $\hat{w}_{i,j}$ are constant weights to be determined.

Replacing $V_i(x)$, $u_i(x)$, and $u_{i+1}(x)$ in (5.12) with their approximations, we obtain

$$\begin{aligned} &\sum_{j=1}^{N_1} \hat{c}_{i,j}\left[\phi_j(x(t_{k+1})) - \phi_j(x(t_k))\right] \\ = &-\int_{t_k}^{t_{k+1}} 2r\sum_{j=1}^{N_2} \hat{w}_{i,j}\phi_j(x)\hat{v}_i dt \\ &-\int_{t_k}^{t_{k+1}} \left[Q(x) + r\hat{u}_i^2(x)\right] dt + e_{i,k} \end{aligned} \tag{5.15}$$

where $\hat{u}_0 = u_0$, $\hat{v}_i = u + \Delta - \hat{u}_i$, and $\{t_k\}_{k=0}^{l}$ is a strictly increasing sequence with $l > 0$ a sufficiently large integer. Then, the weights $\hat{c}_{i,j}$ and $\hat{w}_{i,j}$ can be solved in the sense of least-squares (i.e., by minimizing $\sum_{k=0}^{l} e_{i,k}^2$).

Now, starting from $u_0(x)$, two sequences $\{\hat{V}_i(x)\}_{i=0}^{\infty}$, and $\{\hat{u}_{i+1}(x)\}_{i=0}^{\infty}$ can be generated via the online policy iteration technique (5.15). Next, we show the convergence of the sequences to $V_i(x)$ and $u_{i+1}(x)$, respectively.

Assumption 5.2.1. *There exist $l_0 > 0$ and $\delta > 0$, such that for all $l \geq l_0$, we have*

$$\frac{1}{l}\sum_{k=0}^{l} \theta_{i,k}^T\theta_{i,k} \geq \delta I_{N_1+N_2} \tag{5.16}$$

where

$$\theta_{i,k}^T = \begin{bmatrix} \phi_1(x(t_{k+1})) - \phi_1(x(t_k)) \\ \phi_2(x(t_{k+1})) - \phi_2(x(t_k)) \\ \vdots \\ \phi_{N_1}(x(t_{k+1})) - \phi_{N_1}(x(t_k)) \\ 2r\int_{t_k}^{t_{k+1}} \phi_1(x)\hat{v}_i dt \\ 2r\int_{t_k}^{t_{k+1}} \phi_2(x)\hat{v}_i dt \\ \vdots \\ 2r\int_{t_k}^{t_{k+1}} \phi_{N_2}(x)\hat{v}_i dt \end{bmatrix} \in \mathbb{R}^{N_1+N_2}.$$

Assumption 5.2.2. *The closed-loop system composed of* (5.8), (5.9), *and*

$$u = u_0(x) + e \tag{5.17}$$

is ISS when e, the exploration noise, is considered as the input.

Remark 5.2.1. *The reason for imposing Assumption 5.2.2 is twofold. First, like in many other policy-iteration-based ADP algorithms, an initial admissible control policy is desired. Inspired by [205], we further assume the initial control policy is stabilizing in the presence of dynamic uncertainties. Such an assumption is feasible and realistic by means of the designs in [80], [129]. Second, by adding the exploration noise, we are able to satisfy Assumption 5.2.1, and at the same time keep the system solutions bounded.*

Under Assumption 5.2.2, we can find a compact set Ω_0 which is an invariant set of the closed-loop system composed of (5.8), (5.9), and $u = u_0(x)$. In addition, we can also let Ω_0 contain Ω_{i^*} as its subset. Then, the compact set for approximation can be selected as $\Omega = \{x : \exists w, \text{ s.t. } (w, x) \in \Omega_0\}$.

Theorem 5.2.1. *Under Assumptions 5.2.1 and 5.2.2, for each $i \geq 0$ and given $\epsilon > 0$, there exist $N_1^* >$ and $N_2^* > 0$, such that*

$$|\sum_{j=1}^{N_1} \hat{c}_{i,j}\phi_j(x) - V_i(x)| \quad < \epsilon, \tag{5.18}$$

$$|\sum_{j=1}^{N_2} \hat{w}_{i,j}\phi_j(x) - u_{i+1}(x)| < \epsilon, \tag{5.19}$$

for all $x \in \Omega.$, if $N_1 > N_1^*$ and $N_2 > N_2^*$.

Proof. To begin with, given $\hat{u}_i$, let $\tilde{V}_i(x)$ be the solution of the following equation with $\tilde{V}_i(0) = 0$.

$$\nabla\tilde{V}_i(x)^T\left(f(x) + g(x)\hat{u}_i(x)\right) + Q(x) + r\hat{u}_i^2(x) = 0 \tag{5.20}$$

and denote $\tilde{u}_{i+1}(x) = -\frac{1}{2r}g(x)^T\nabla\tilde{V}_i(x)^T$.

Lemma 5.2.1. *For each $i \geq 0$, we have $\lim_{N_1,N_2\to\infty} \hat{V}_i(x) = \tilde{V}_i(x)$, $\lim_{N_1,N_2\to\infty} \hat{u}_{i+1}(x) = \tilde{u}_{i+1}(x)$, $\forall x \in \Omega$.*

Proof. By definition

$$\begin{aligned} & \tilde{V}_i(x(t_{k+1})) - \tilde{V}_i(x(t_k)) \\ = \; & -\int_{t_k}^{t_{k+1}} [Q(x) + r\hat{u}_i^2(x) + 2r\tilde{u}_{i+1}(x)\hat{v}_i]dt \end{aligned} \tag{5.21}$$

Let $\tilde{c}_{i,j}$ and $\tilde{w}_{i,j}$ be the constant weights such that $\tilde{V}_i(x) = \sum_{j=1}^{\infty} \tilde{c}_{i,j}\phi_j(x)$ and $\tilde{u}_{i+1}(x) = \sum_{j=1}^{\infty} \tilde{w}_{i,j}\phi_j(x)$. Then, by (5.15) and (5.21), we have $e_{i,k} = \theta_{i,k}^T\bar{W}_i + \xi_{i,k}$,

where

$$\begin{aligned}
\bar{W}_i &= \begin{bmatrix} \tilde{c}_{i,1} & \tilde{c}_{i,2} & \cdots & \tilde{c}_{i,N_1} & \tilde{w}_{i,1} & \tilde{w}_{i,2} & \cdots & \tilde{w}_{i,N_2} \end{bmatrix}^T \\
&\quad - \begin{bmatrix} \hat{c}_{i,1} & \hat{c}_{i,2} & \cdots & \hat{c}_{i,N_1} & \hat{w}_{i,1} & \hat{w}_{i,2} & \cdots & \hat{w}_{i,N_2} \end{bmatrix}^T, \\
\xi_{i,k} &= \sum_{j=N_1+1}^{\infty} \tilde{c}_{i,j} \left[\phi_j(x(t_{k+1})) - \phi_j(x(t_k)) \right] \\
&\quad + \sum_{j=N_2+1}^{\infty} \tilde{w}_{i,j} \int_{t_k}^{t_{k+1}} 2r\phi_j(x)\hat{v}_i dt.
\end{aligned}$$

Since the weights are found using the least-squares method, we have

$$\sum_{k=1}^{l} e_{i,k}^2 \leq \sum_{k=1}^{l} \xi_{i,k}^2$$

Also, notice that,

$$\sum_{k=1}^{l} \bar{W}_i^T \theta_{i,k}^T \theta_{i,k} \bar{W}_i = \sum_{k=1}^{l} (e_{i,k} - \xi_{i,k})^2$$

Then, under Assumption 5.2.1, it follows that

$$|\bar{W}_i|^2 \leq \frac{4|\Xi_{i,l}|^2}{l\delta} = \frac{4}{\delta} \max_{1 \leq k \leq l} \xi_{i,k}^2.$$

Therefore, given any arbitrary $\epsilon > 0$, we can find $N_{10} > 0$ and $N_{20} > 0$, such that when $N_1 > N_{10}$ and $N_2 > N_{20}$, we have

$$\begin{aligned}
& |\hat{V}_i(x) - \tilde{V}_i(x)| \\
\leq\ & \sum_{j=1}^{N_1} |c_{i,j} - \hat{c}_{i,j}||\phi_j(x)| + \sum_{j=N_1+1}^{\infty} |c_{i,j}\phi_j(x)| \\
\leq\ & \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon, \quad \forall x \in \Omega.
\end{aligned} \tag{5.22}$$

Similarly, $|\hat{u}_{i+1}(x) - \tilde{u}_{i+1}(x)| \leq \epsilon$. The proof is complete. $\square$

We now prove Theorem 5.2.1 by induction:

1) If $i = 0$ we have $\tilde{V}_0(x) = V_0(x)$, and $\tilde{u}_1(x) = u_1(x)$. Hence, the convergence can directly be proved by Lemma A.1.

2) Suppose for some $i > 0$, we have $\lim_{N_1,N_2\to\infty} \hat{V}_{i-1}(x) = V_{i-1}(x)$, $\lim_{N_1,N_2\to\infty} \hat{u}_i(x) = u_i(x)$, $\forall x \in \Omega$. By definition, we have

$$\begin{aligned} & |V_i(x(t)) - \tilde{V}_i(x(t))| \\ = \;& r| \int_t^\infty \left[\hat{u}_i(x)^2 - u_i(x)^2\right] dt| \\ + \;& 2r| \int_t^\infty u_{i+1}(x) \left[\hat{u}_i(x) - u_i(x)\right] dt| \\ + \;& 2r| \int_t^\infty \left[\tilde{u}_{i+1}(x) - u_{i+1}(x)\right] \hat{v}_i dt|, \quad \forall x \in \Omega. \end{aligned}$$

By the induction assumptions, we know

$$\lim_{N_1,N_2\to\infty} \int_t^\infty \left[\hat{u}_i(x)^2 - u_i(x)^2\right] dt = 0 \tag{5.23}$$

$$\lim_{N_1,N_2\to\infty} \int_t^\infty u_{i+1}(x) \left[\hat{u}_i(x) - u_i(x)\right] dt = 0 \tag{5.24}$$

Also, by Assumption 5.2.1, we conclude

$$\lim_{N_1,N_2\to\infty} |u_{i+1}(x) - \tilde{u}_{i+1}(x)| = 0 \tag{5.25}$$

and

$$\lim_{N_1,N_2\to\infty} |V_i(x) - \tilde{V}_i(x)| = 0. \tag{5.26}$$

Finally, since

$$|\hat{V}_i(x) - V_i(x)| \leq |V_i(x) - \tilde{V}_i(x)| + |\tilde{V}_i(x) - \hat{V}_i(x)|$$

and by the induction assumption, we have

$$\lim_{N_1, N_2 \to \infty} |V_i(x) - \hat{V}_i(x)| = 0. \tag{5.27}$$

Similarly, we can show

$$\lim_{N_1, N_2 \to \infty} |u_{i+1}(x) - \hat{u}_i(x)| = 0. \tag{5.28}$$

The proof is thus complete. □

Corollary 5.2.1. *Assume $V^*(x)$ and $u^*(x)$ exist. Then, under Assumptions 5.2.1 and 5.2.2, for any arbitrary $\epsilon > 0$, there exist integers $i^* > 0$, $N_1^{**} > 0$ and $N_2^{**} > 0$, such that*

$$|\sum_{j=1}^{N_1} \hat{c}_{i^*,j}\phi_j(x) - V^*(x)| \leq \epsilon, \tag{5.29}$$

$$|\sum_{j=1}^{N_2} \hat{w}_{i^*,j}\phi_j(x) - u^*(x)| \leq \epsilon, \tag{5.30}$$

for all $x \in \Omega$, if $N_1 > N_1^{**}$, and $N_2 > N_2^{**}$.

Proof. By Theorem 6.1.1, there exists $i^* > 0$, such that

$$|V_{i^*}(x) - V^*(x)| \leq \frac{\epsilon}{2}, \tag{5.31}$$

$$|u_{i^*+1}(x) - u^*(x)| \leq \frac{\epsilon}{2}, \ \forall x \in \Omega. \tag{5.32}$$

By Theorem 5.2.1, there exist $N_1^{**} > 0$ and $N_2^{**} > 0$, such that for all $N_1 > N_1^{**}$

and $N_2 > N_2^{**}$,

$$|\sum_{j=1}^{N_1} \hat{c}_{i^*,j}\phi_j(x) - V_{i^*}(x)| \leq \frac{\epsilon}{2}, \tag{5.33}$$

$$|\sum_{j=1}^{N_2} \hat{w}_{i^*,j}\phi_j(x) - u_{i^*+1}(x)| \leq \frac{\epsilon}{2}, \ \forall x \in \Omega. \tag{5.34}$$

The corollary is thus proved by using the triangle inequality. □

5.2.2 Robust redesign

In the presence of the dynamic uncertainty, we redesign the approximated optimal control policy so as to achieve robust stability. The proposed method is an integration of optimal control theory [99] with the gain assignment technique [83], [129]. To begin with, let us make the following assumptions.

Assumption 5.2.3. *There exists a function $\underline{\alpha}$ of class $\mathcal{K}_\infty$, such that for $i = 0, 1, \cdots$,*

$$\underline{\alpha}(|x|) \leq V_i(x), \quad \forall x \in \mathbb{R}^n. \tag{5.35}$$

In addition, assume there exists a constant $\epsilon > 0$ such that $Q(x) - \epsilon^2|x|^2$ is a positive definite function.

Notice that, we can also find a class $\mathcal{K}_\infty$ function $\bar{\alpha}$, such that for $i = 0, 1, \cdots$,

$$V_i(x) \leq \bar{\alpha}(|x|), \quad \forall x \in \mathbb{R}^n. \tag{5.36}$$

Assumption 5.2.4. *Consider (5.8). There exist functions $\underline{\lambda}, \bar{\lambda} \in \mathcal{K}_\infty$, $\kappa_1, \kappa_2, \kappa_3 \in \mathcal{K}$, and positive definite functions W and κ_4, such that for all $w \in \mathbb{R}^p$ and $x \in \mathbb{R}^n$, we*

have

$$\underline{\lambda}(|w|) \leq W(w) \leq \bar{\lambda}(|w|), \tag{5.37}$$

$$|\Delta(w,x)| \leq \max\{\kappa_1(|w|), \kappa_2(|x|)\}, \tag{5.38}$$

together with the following implication:

$$W(w) \geq \kappa_3(|x|) \Rightarrow \nabla W(w)\Delta_w(w,x) \leq -\kappa_4(w). \tag{5.39}$$

Assumption 6.6.1 implies that the w-system (5.8) is input-to-state stable (ISS) [149], [151] when x is considered as the input.

Now, consider the following type of control policy

$$u_{ro}(x) = \left[1 + \frac{r}{2}\rho^2(|x|^2)\right] \hat{u}_{i^*+1}(x) \tag{5.40}$$

where $i^* > 0$ is a sufficiently large integer as defined in Corollary 5.2.1, ρ is a smooth, non-decreasing function, with $\rho(s) > 0$ for all $s \geq 0$. Notice that u_{ro} can be viewed as a robust redesign of the approximated optimal control policy $\hat{u}_{i^*+1}$.

As in [80], let us define a class $\mathcal{K}_\infty$ function γ by

$$\gamma(s) = \frac{1}{2}\epsilon\rho(s^2)s, \quad \forall s \geq 0. \tag{5.41}$$

In addition, define

$$\begin{aligned} e_{ro}(x) &= \frac{r}{2}\rho^2(|x|^2)\left[\hat{u}_{i^*+1}(x) - u_{i^*+1}(x)\right] \\ &\quad + \hat{u}_{i^*+1}(x) - u_{i^*}(x). \end{aligned} \tag{5.42}$$

Theorem 5.2.2. *Under Assumptions 5.2.3 and 6.6.1, suppose*

$$\gamma > \max\{\kappa_2, \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \circ \bar{\alpha}\}, \tag{5.43}$$

and the following implication holds for some constant $d > 0$:

$$0 < V_{i^*}(x) \leq d \Rightarrow |e_{ro}(x)| < \gamma(|x|). \tag{5.44}$$

Then, the closed-loop system composed of (5.8), (5.9), and (5.40) is asymptotically stable at the origin. In addition, there exists $\sigma \in \mathcal{K}_\infty$, such that $\Omega_{i^*} = \{(w, x) : \max[\sigma(V_{i^*}(x)), W(w)] \leq \sigma(d)\}$ is an estimate of the region of attraction of the closed-loop system.

Proof. Define

$$\bar{e}_{ro}(x) = \begin{cases} e_{ro}(x), & V_{i^*}(x) \leq d \\ 0, & V_{i^*}(x) > d \end{cases} \tag{5.45}$$

and

$$u(x) = u_{i^*}(x) + \frac{r}{2}\rho^2(|x|^2)u_{i^*+1}(x) + \bar{e}_{ro}(x). \tag{5.46}$$

Then, along the solutions of (5.9), by completing the squares, we have

$$\begin{aligned} \dot{V}_{i^*} &\leq -Q(x) + \frac{1}{\rho^2(|x|^2)}(\Delta + \bar{e}_{ro}(x))^2 \\ &= -(Q(x) - \epsilon^2|x|^2) - \frac{4\gamma^2 - (\Delta + \bar{e}_{ro}(x))^2}{\rho^2(|x|^2)} \\ &\leq -Q_0(x) - 4\frac{\gamma^2 - \max\{\kappa_1^2(|w|), \kappa_2^2(|x|), \bar{e}_{ro}^2(|x|)\}}{\rho^2(|x|^2)} \end{aligned}$$

where $Q_0(x) = Q(x) - \epsilon^2|x|^2$ is a positive definite function of x.

Therefore, under Assumptions 5.2.3, 6.6.1 and the gain condition (5.43), we have the following implication:

$$\begin{aligned} & V_{i^*}(x) \geq \bar{\alpha} \circ \gamma^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1}(W(w)) \\ \Rightarrow \quad & |x| \geq \gamma^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1}(W(w)) \\ \Rightarrow \quad & \gamma(|x|) \geq \kappa_1(|w|) \\ \Rightarrow \quad & \gamma(|x|) \geq \max\{\kappa_1(|w|), \kappa_2(|x|), \bar{e}_{ro}(|x|)\} \\ \Rightarrow \quad & \dot{V}_{i^*}(x) \leq -Q_0(x). \end{aligned} \tag{5.47}$$

Also, under Assumption 6.6.1, we have

$$\begin{aligned} & W(w) \geq \kappa_3 \circ \underline{\alpha}^{-1}(V_{i^*}(x)) \\ \Rightarrow \quad & W(w) \geq \kappa_3(|x|) \\ \Rightarrow \quad & \nabla W(w) \Delta_w(w, x) \leq -\kappa_4(|w|). \end{aligned} \tag{5.48}$$

Finally, under the gain condition (5.43), it follows that

$$\begin{aligned} & \gamma(s) > \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \circ \bar{\alpha}(s) \\ \Rightarrow \quad & \gamma \circ \bar{\alpha}^{-1}(s') > \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1}(s') \\ \Rightarrow \quad & s' > \bar{\alpha} \circ \gamma^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1}(s') \end{aligned} \tag{5.49}$$

where $s' = \bar{\alpha}(s)$. Hence, the following small-gain condition holds:

$$\left[\bar{\alpha} \circ \gamma^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1}\right] \circ \left[\kappa_3 \circ \underline{\alpha}^{-1}(s)\right] < s, \quad \forall s > 0. \tag{5.50}$$

Denoting $\chi_1 = \bar{\alpha} \circ \gamma^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1}$, and $\chi_2 = \kappa_3 \circ \underline{\alpha}^{-1}$, by Theorem 9.2.1, the system composed of (5.8), (5.9), (5.46) is globally asymptotically stable at the origin.

In addition, by Theorem 9.2.1, there exists a continuously differentiable class $\mathcal{K}_\infty$ function $\sigma(s)$, such that the set

$$\Omega_{i^*} = \{(w, x) : \max[\sigma(V_{i^*}(x)), W(w)] \leq \sigma(d)\} \tag{5.51}$$

is an estimate of the region of attraction of the closed-loop system.

The proof is thus complete. □

Remark 5.2.2. *In the absence of the dynamic uncertainty (i.e., $\Delta = 0$ and the w-system is absent), the control policy* (5.40) *can be replaced by $\hat{u}_{i^*+1}(x)$, which is an approximation of the optimal control policy $u^*(x)$ that minimizes the following cost*

$$J(x_0, u) = \int_0^\infty \left[Q(x) + ru^2\right] dt, \quad x(0) = x_0. \tag{5.52}$$

Remark 5.2.3. *It is of interest to note that the constant d in* (5.44) *can be chosen arbitrarily large. So, the proposed control scheme solves the semi-global stabilization problem [161].*

5.2.3 The RADP Algorithm

The RADP algorithm is given in Algorithm 5.2.1, and a graphical illustration is provided in Figure 5.1.

Algorithm 5.2.1 RADP Algorithm

1. Let $(w(0), x(0)) \in \Omega_{i^*} \subset \Omega_0$, employ the initial control policy (5.17) and collect the system state and input information. Let $i \leftarrow 0$.

2. Solve $\hat{c}_{i,j}$ and $\hat{w}_{i,j}$ from (5.15).

3. Let $i \leftarrow i + 1$, and go to Step 2, until

$$\sum_{j=1}^{N_1} |\hat{c}_{i,j} - \hat{c}_{i-1,j}|^2 \leq \epsilon_1 \tag{5.53}$$

   where the constant $\epsilon_1 > 0$ is a sufficiently small predefined threshold.

4. Terminate the exploration noise e.

5. If $(w(t), x(t)) \in \Omega_{i^*}$, apply the approximate robust optimal control policy (5.40).

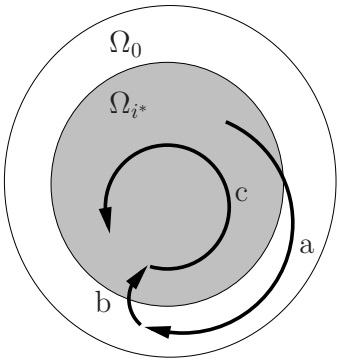

Figure 5.1: Illustration of Algorithm 5.2.1. **a**. The initial stabilizing control policy is employed, and online information of state variables of the x-subsystem, the input signal u, and the output of the dynamic uncertainty is utilized to approximate the optimal cost and the optimal control policy. **b.** The exploration noise is terminated after convergence to the optimal control policy is attained. **c.** The robust optimal control policy is applied as soon as the system trajectory enters the invariant set Ω_{i^*}.

5.3 RADP with unmatched dynamic uncertainty

In this section, we extend the RADP methodology to nonlinear systems with unmatched dynamic uncertainties. To begin with, consider the system:

$$\dot{w} = \Delta_w(w, x) \tag{5.54}$$

$$\dot{x} = f(x) + g(x)\left[z + \Delta(w, x)\right] \tag{5.55}$$

$$\dot{z} = f_1(x, z) + u + \Delta_1(w, x, z) \tag{5.56}$$

where $[x^T, z]^T \in \mathbb{R}^n \times \mathbb{R}$ is the measured component of the state available for feedback control; w, u, Δ_w, f, g, and Δ are defined in the same way as in (5.8)-(5.9); $f_1 : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ and $\Delta_1 : \mathbb{R}^p \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ are locally Lipschitz functions and are assumed to be unknown.

Assumption 5.3.1. *There exist class $\mathcal{K}$ functions $\kappa_5, \kappa_6, \kappa_7$, such that the following inequality holds:*

$$|\Delta_1(w, x, z)| \leq \max\{\kappa_5(|w|), \kappa_6(|x|), \kappa_7(|z|)\}. \tag{5.57}$$

5.3.1 Online learning

Let us define a virtual control policy $\xi = u_{ro}$, where u_{ro} is the same as in (5.40). Then, a state transformation can be performed as $\zeta = z - \xi$. Along the trajectories of (5.55)-(5.56), it follows that

$$\dot{\zeta} = \bar{f}_1(x, z) + u + \Delta_1 - \bar{g}_1(x)\Delta \tag{5.58}$$

where $\bar{f}_1(x,z) = f_1(x,z) - \frac{\partial \xi}{\partial x} f(x) - \frac{\partial \xi}{\partial x} g(x) z$, and $\bar{g}_1(x) = \frac{\partial \xi}{\partial x} g(x)$. By approximation theory [126], $\bar{f}_1(x,z)$ and $\bar{g}_1(x)$ can be approximated by:

$$\hat{f}_1(x,z) = \sum_{j=1}^{N_3} \hat{w}_{f,j} \psi_j(x,z), \tag{5.59}$$

$$\hat{g}_1(x) = \sum_{j=0}^{N_4-1} \hat{w}_{g,j} \phi_j(x) \tag{5.60}$$

where $\{\psi_j(x,z)\}_{j=1}^{\infty}$ is a sequence of linearly independent basis functions on some compact set $\Omega_1 \in \mathbb{R}^{n+1}$ containing the origin as its interior, $\phi_0(x) \equiv 1$, $\hat{w}_{f,j}$ and $\hat{w}_{g,j}$ are constant weights to be trained. As in the matched case, a similar assumption on the initial control policy is given as follows.

Assumption 5.3.2. *The closed-loop system composed of* (5.54)-(5.56)*, and*

$$u = \bar{u}_0(x,z) + e \tag{5.61}$$

is ISS when e, the exploration noise, is considered to be the input.

Under Assumption 5.3.2, we can find an invariant set $\Omega_{1,0}$ for the closed-loop system composed of (5.54)-(5.56) and (5.61), and approximate the unknown functions on the set

$$\Omega_1 = \{(x,z) : \exists w, \text{s.t.} (w,x,z) \in \Omega_{1,0}\}. \tag{5.62}$$

Then, we give the following two-phase learning scheme.

Phase-one learning

Similarly as in (5.15), to approximate the virtual control input ξ for the x-subsystem, we solve the weights $\hat{c}_{i,j}$ and $\hat{w}_{i,j}$ using least-squares method from

$$\begin{aligned} & \sum_{j=1}^{N_1} \hat{c}_{i,j} \left[\phi_j(x(t'_{k+1})) - \phi_j(x(t'_k))\right] \\ = & -\int_{t'_k}^{t'_{k+1}} 2r \sum_{j=1}^{N_2} \hat{w}_{i,j}\phi_j(x)\tilde{v}_i dt \\ & -\int_{t'_k}^{t'_{k+1}} \left[Q(x) + r\hat{u}_i^2(x)\right] dt + \tilde{e}_{i,k} \end{aligned} \tag{5.63}$$

where $\tilde{v}_i = z + \Delta - \hat{u}_i$, and $\{t'_k\}_{k=0}^{l}$ is a strictly increasing sequence with $l > l_0$ a sufficiently large integer, and $\tilde{e}_{i,k}$ is the approximation error.

Phase two learning

To approximate the unknown functions $\bar{f}_1$ and $\bar{g}_1$, The constant weights can be solved, in the sense of least-squares, from

$$\begin{aligned} & \frac{1}{2}\zeta^2(t'_{k+1}) - \frac{1}{2}\zeta^2(t'_k) \\ = & \int_{t'_k}^{t'_{k+1}} \left[\sum_{j=1}^{N_3} \hat{w}_{f,j}\psi_j(x,z) - \sum_{j=0}^{N_4-1} \hat{w}_{g,j}\phi_j(x)\Delta\right] \zeta dt \\ & + \int_{t'_k}^{t'_{k+1}} (u + \Delta_1)\zeta dt + \bar{e}_k \end{aligned} \tag{5.64}$$

where $\bar{e}_k$ denotes the approximation error. Similarly as in the previous section, let us introduce the following assumption:

Assumption 5.3.3. *There exist $l_1 > 0$ and $\delta_1 > 0$, such that for all $l \geq l_1$, we have*

$$\frac{1}{l}\sum_{k=0}^{l} \bar{\theta}_k^T \bar{\theta}_k \geq \delta_1 I_{N_3+N_4} \tag{5.65}$$

where

$$\bar{\theta}_k^T = \begin{bmatrix} \int_{t'_k}^{t'_{k+1}} \psi_1(x,z)\zeta dt \\ \int_{t'_k}^{t'_{k+1}} \psi_2(x,z)\zeta dt \\ \vdots \\ \int_{t'_k}^{t'_{k+1}} \psi_{N_3}(x,z)\zeta dt \\ \int_{t'_k}^{t'_{k+1}} \phi_0(x)\Delta\zeta dt \\ \int_{t'_k}^{t'_{k+1}} \phi_1(x)\Delta\zeta dt \\ \vdots \\ \int_{t'_k}^{t'_{k+1}} \phi_{N_4-1}(x)\Delta\zeta dt \end{bmatrix} \in \mathbb{R}^{N_3+N_4}.$$

Theorem 5.3.1. *Consider* $(x(0), z(0)) \in \Omega_1$. *Then, under Assumption 5.3.3 we have*

$$\lim_{N_3,N_4\to\infty} \hat{f}(x,z) = \bar{f}_1(x,z), \tag{5.66}$$

$$\lim_{N_3,N_4\to\infty} \hat{g}(x) = \bar{g}_1(x), \quad \forall (x,z) \in \Omega_1. \tag{5.67}$$

5.3.2 Robust redesign

Next, we study the robust stabilization of the system (5.54)-(5.56). To this end, let κ_8 be a function of $\mathcal{K}$ such that

$$\kappa_8(|x|) \geq |\xi(x)|, \quad \forall x \in \mathbb{R}^n. \tag{5.68}$$

Then, Assumption 5.3.1 implies

$$\begin{aligned} |\Delta_1| &\leq \max\{\kappa_5(|w|), \kappa_6(|x|), \kappa_7(|z|)\} \\ &\leq \max\{\kappa_5(|w|), \kappa_6(|x|), \kappa_7(|\xi| + \kappa_8(|x|))\} \\ &\leq \max\{\kappa_5(|w|), \kappa_9(|X_1|)\} \end{aligned}$$

where $\kappa_9(s) = \max\{\kappa_6, \kappa_7 \circ \kappa_8 \circ (2s), \kappa_7 \circ (2s)\}$, $\forall s \geq 0$. In addition, we denote $\tilde{\kappa}_1 = \max\{\kappa_1, \kappa_5\}$, $\tilde{\kappa}_2 = \max\{\kappa_2, \kappa_9\}$, $\gamma_1(s) = \frac{1}{2}\epsilon\rho(\frac{1}{2}s^2)s$, and

$$U_{i^*}(X_1) = V_{i^*}(x) + \frac{1}{2}\zeta^2. \tag{5.69}$$

Notice that, under Assumptions 5.2.3 and 6.6.1, there exist $\bar{\alpha}_1, \underline{\alpha}_1 \in \mathcal{K}_\infty$, such that $\underline{\alpha}_1(|X_1|) \leq U_{i^*}(X_1) \leq \bar{\alpha}_1(|X_1|)$.

The control policy can be approximated by

$$\begin{aligned} u_{ro1} &= -\hat{f}_1(x,z) + 2r\hat{u}_{i^*+1}(x) \\ &\quad -\frac{\hat{g}^2(x)\rho_1^2(|X_1|^2)\zeta}{4} - \epsilon^2\zeta \\ &\quad -\frac{\rho_1^2(|X_1|^2)\zeta}{4} - \frac{\epsilon^2\rho^2(\zeta^2)\zeta}{2\rho^2(|x|^2)} \end{aligned} \tag{5.70}$$

where $X_1 = [x^T, \zeta]^T$, and $\rho_1(s) = 2\rho(\frac{1}{2}s)$.

Next, define the approximation error as

$$\begin{aligned} e_{ro1}(X_1) &= -\bar{f}_1(x,z) + \hat{f}_1(x,z) \\ &\quad +2r\left[u_{i^*+1}(x) - \hat{u}_{i^*+1}(x)\right] \\ &\quad -\frac{\left[\bar{g}_1^2(x) - \hat{g}_1^2(x)\right]\rho_1^2(|X_1|^2)\zeta}{4} \end{aligned} \tag{5.71}$$

Then, the conditions for asymptotic stability are summarized in the following Theorem:

Theorem 5.3.2. *Under Assumptions 5.2.3, 6.6.1, and 5.3.1, if*

$$\gamma_1 > \max\{\tilde{\kappa}_2, \tilde{\kappa}_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}_1^{-1} \circ \bar{\alpha}_1\}, \tag{5.72}$$

and if the following implication holds for some constant $d_1 > 0$*:*

$$0 < U_{i^*}(X_1) \leq d_1 \Rightarrow \max\{|e_{ro1}(X_1)|, |e_{ro}(x)|\} < \gamma_1(|X_1|),$$

then the closed-loop system comprised of (5.54)-(5.56)*, and* (5.70) *is asymptotically stable at the origin. In addition, there exists* $\sigma_1 \in \mathcal{K}_\infty$*, such that*

$$\Omega_{1,i^*} = \{(w, X_1) : \max\left[\sigma_1(U_{i^*}(X_1)), W(w)\right] \leq \sigma_1(d_1)\}$$

is an estimate of the region of attraction.

Proof. Define

$$\bar{e}_{ro1}(X_1) = \begin{cases} e_{ro1}(X_1), & U_{i^*}(X_1) \leq d_1, \\ 0, & U_{i^*}(X_1) > d_1, \end{cases}$$

$$\bar{\bar{e}}_{ro}(x) = \begin{cases} e_{ro}(x), & U_{i^*}(X_1) \leq d_1, \\ 0, & U_{i^*}(X_1) > d_1, \end{cases}$$

Along the solutions of (5.54)-(5.56) with the control policy

$$\begin{aligned} u = & -\bar{f}_1(x, z) + 2r\hat{u}_{i^*+1}(x) - \frac{\bar{g}^2(x)\rho_1^2(|X_1|^2)\zeta}{4} \\ & -\frac{\rho_1^2(|X_1|^2)\zeta}{4} - \frac{\epsilon^2\rho^2(\zeta^2)\zeta}{2\rho^2(|x|^2)} - \epsilon^2\zeta - \bar{e}_{ro1}(X_1), \end{aligned}$$

it follows that

$$\begin{aligned}\dot{U}_{i^*} \;\leq\; & -Q_0(x) - \frac{1}{2}\epsilon^2\zeta^2 \\ & - \frac{\gamma_1^2(|X_1|) - \max\{\tilde{\kappa}_1^2(|w|), \tilde{\kappa}_2^2(|X_1|), \bar{\bar{e}}_{ro}^2(x)\}}{\frac{1}{4}\rho^2(|x|^2)} \\ & - \frac{\gamma_1^2(|X_1|) - \max\{\tilde{\kappa}_1^2(|w|), \tilde{\kappa}_2^2(|X_1|), \bar{\bar{e}}_{ro}^2(x)\}}{\frac{1}{4}\rho_1^2(|X_1|^2)} \\ & - \frac{\gamma_1^2(|X_1|) - \max\{\tilde{\kappa}_1^2(|w|), \tilde{\kappa}_2^2(|X_1|), \bar{e}_{ro1}^2(X_1)\}}{\frac{1}{4}\rho_1^2(|X_1|^2)}\end{aligned}$$

As a result,

$$\begin{aligned}U_{i^*}(X_1) \geq \bar{\alpha}_1 \circ \gamma_1^{-1} \circ \tilde{\kappa}_1 \circ \underline{\lambda}^{-1}(W(w)) \\ \Rightarrow \dot{U}_{i^*} \leq -Q_0(x) - \tfrac{1}{2}\epsilon^2|\zeta|^2.\end{aligned}$$

The rest of the proof follows the same reasoning as in the proof of Theorem 5.2.2. □

Remark 5.3.1. *In the absence of the dynamic uncertainty (i.e., $\Delta = 0$, $\Delta_1 = 0$ and the w-system is absent), the smooth functions ρ and ρ_1 can all be replaced by 0, and the system becomes*

$$\dot{X}_1 = F_1(X_1) + G_1 u_{o1} \tag{5.73}$$

where $F_1(X_1) = \begin{bmatrix} f(x) + g(x)\zeta + g(x)\xi \\ -\nabla V_{i^*}(x)^T g(x) \end{bmatrix}$, $G_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, *and* $u_{o1} = -\epsilon^2\zeta^2$. *As a result, it can be concluded that the control policy $u = u_{o1}$ is an approximate optimal control policy with respect to the cost functional*

$$J_1(X_1(0); u) = \int_0^\infty \left[Q_1(x, \zeta) + \frac{1}{2\epsilon^2}u^2\right] dt \tag{5.74}$$

with $X_1(0) = [x_0^T, z_0 - u_{i^*}(x_0)]^T$ *and* $Q_1(x, \zeta) = Q(x) + \frac{1}{4r}\left[\nabla V_{i^*}(x)^T g(x)\right]^2 + \frac{\epsilon^2}{2}\zeta^2$.

Remark 5.3.2. *Like in the matched case, by selecting large enough* d_1 *in Theorem 5.3.2, semi-global stabilization is achieved.*

5.3.3 The RADP algorithm with unmatched dynamic uncertainty

The RADP algorithm with unmatched dynamic uncertainty is given in Algorithm 5.3.1.

Algorithm 5.3.1 RADP Algorithm

1. Let $(w(0), x(0), z(0)) \in \Omega_{1,i^*} \subset \Omega_{1,0}$, employ the initial control policy (5.61) and collect the system state and input information. Let $i \leftarrow 0$.
2. Solve $\hat{c}_{i,j}$ and $\hat{w}_{i,j}$ from (5.63).
3. Let $i \leftarrow i + 1$, and go to Step 2, until
$$\sum_{j=1}^{N_1} |\hat{c}_{i,j} - \hat{c}_{i-1,j}|^2 \leq \epsilon_1 \tag{5.75}$$
where the constant $\epsilon_1 > 0$ is a sufficiently small predefined threshold.
4. Solve $\hat{w}_{f,j}$ and $\hat{w}_{g,j}$ from (5.64).
5. Terminate the exploration noise e.
6. If $(w(t), x(t), z(t)) \in \Omega_{1,i^*}$, apply the approximate robust optimal control policy (5.70).

5.4 Numerical examples

In this section, we apply the proposed online RADP schemes to the design of robust optimal control policies for a jet engine and a one-machine power system.

5.4.1 Jet engine

Consider the following dynamic model for jet engine surge and stall [119], [95] described by

$$\dot{\Phi} = -\Psi + \Psi_C(\Phi) - 3\Phi R \tag{5.76}$$

$$\dot{\Psi} = \frac{1}{\beta^2}\left(\Phi - \Phi_T(\Psi)\right) \tag{5.77}$$

$$\dot{R} = \sigma R\left(1 - \Phi^2 - R\right) \tag{5.78}$$

where Φ is the scaled annulus-averaged flow, Ψ is the plenum pressure rise, and $R > 0$ is the normalized rotating stall amplitude. Functions $\Phi_C(\Phi)$ and $\Phi_T(\Psi)$ are the compressor and throttle characteristics, respectively. According to [119], Φ_T is assumed to take the form of

$$\Phi_T(\Psi) = \gamma\sqrt{\Psi} - 1 \tag{5.79}$$

and $\Psi_C(\Phi)$ is assumed to be satisfying $\Psi_C(0) = 1 + \Psi_{C0}$ with Ψ_{C_0} a constant describing the shut-off pressure rise. The equilibrium of the system is

$$R^{\mathrm{e}} = 0, \Phi^{\mathrm{e}} = 1, \Psi^{\mathrm{e}} = \Psi_C(\Phi^{\mathrm{e}}) = \Psi^{\mathrm{e}}_{C0} + 2.$$

Performing the following state and input transformations [95]

$$\phi = \Phi - \Phi^{\mathrm{e}} \tag{5.80}$$

$$\psi = \Psi - \Psi^{\mathrm{e}} \tag{5.81}$$

$$u = \frac{1}{\beta^2}\left[\phi - \beta^2\gamma\sqrt{\psi + \Psi_{C0} + 2} + 2\right], \tag{5.82}$$

system (5.76)-(5.78) can be converted to

$$\begin{aligned}
\dot{R} &= -\sigma R^2 - \sigma R\left(2\phi + \phi^2\right), \quad R(0) \geq 0 && (5.83)\\
\dot{\phi} &= \Psi_C(\phi + 1) - 2 - \Psi_{C0} \\
&\quad - (\psi + 3R\phi + 3R) && (5.84)\\
\dot{\psi} &= u && (5.85)
\end{aligned}$$

Notice that this system is in the form of (5.54)-(5.56), if we choose $w = R$, $x = \phi$, and $z = \psi$. The function Ψ_C and the constant σ are assumed to be uncertain, but satisfy

$$-\frac{1}{2}\phi^3 - 2\phi^2 \leq \Psi_C(s + 1) - 2 - \Psi_{C0} \leq -\frac{1}{2}\phi^3 \tag{5.86}$$

and $0.1 < \sigma < 1$. The initial stabilizing policy and the initial virtual control policies are selected to be $u = 6\phi - 2\psi$ and $\psi = 3\phi$, with $V(R, \phi, \psi) = R + \frac{1}{2}\phi^2 + \frac{1}{2}(\psi - 3\phi)^2$ a Lyapunov function of the closed-loop system.

For simulation purpose, we set $\sigma = 0.3$, $\beta = 0.702$, $\Psi_{C0} = 1.7$, and $\Psi_C(\phi + 1) = \Psi_{C0} + 2 - \frac{3}{2}\phi^2 - \frac{1}{2}\phi^3$ [119]. We set $Q(\phi) = 4(\phi^2 + \phi^3 + \phi^4)$, and $r = 1$. For robust redesign, we set $\rho(s) = 0.01s$. The basis functions are selected to be polynomials of ϕ and ψ with order less or equal to four. The exploration noise is set to be $e = 10\cos(0.1t)$.

The RADP learning started from the beginning of the simulation and finished at $t = 10s$, when the control policy is updated after six iterations and the convergence criterion (5.75) in Algorithm 1 with $\epsilon_1 = 10^{-6}$ is satisfied. The approximated cost functions before and after phase-one learning are shown in Figure 5.2. The plots of state trajectories of the closed-loop system are shown in Figures 5.3-5.5.

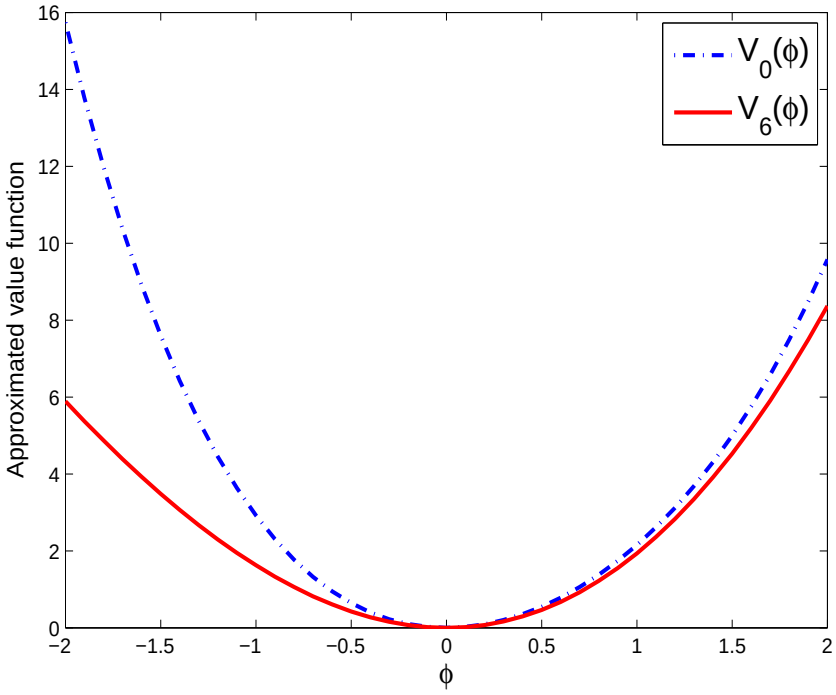

Figure 5.2: Approximated cost function.

5.4.2 One-machine infinite-bus power system

The power system considered in this chapter is a synchronous generator connected to an infinite-bus as shown in Figure 5.6. A model for the generator with both excitation and power control loops can be written as follows [97]:

$$\dot{\delta} = \omega \tag{5.87}$$

$$\dot{\omega} = -\frac{D}{2H}\omega + \frac{\omega_0}{2H}\left(P_m - P_e\right) \tag{5.88}$$

$$\dot{E}'_q = -\frac{1}{T'_d}E'_q + \frac{1}{T_{d0}}\frac{x_d - x'_d}{x'_{d\Sigma}}V_s\cos\delta + \frac{1}{T_{d0}}E_f \tag{5.89}$$

$$\dot{P}_m = -\frac{1}{T_G}P_m - K_G\omega + u \tag{5.90}$$

where δ, ω, P_e, P_m, E'_q, and u are the incremental changes of the rotor angle, the relative rotor speed, the active power delivered to the infinite-bus, the mechanical

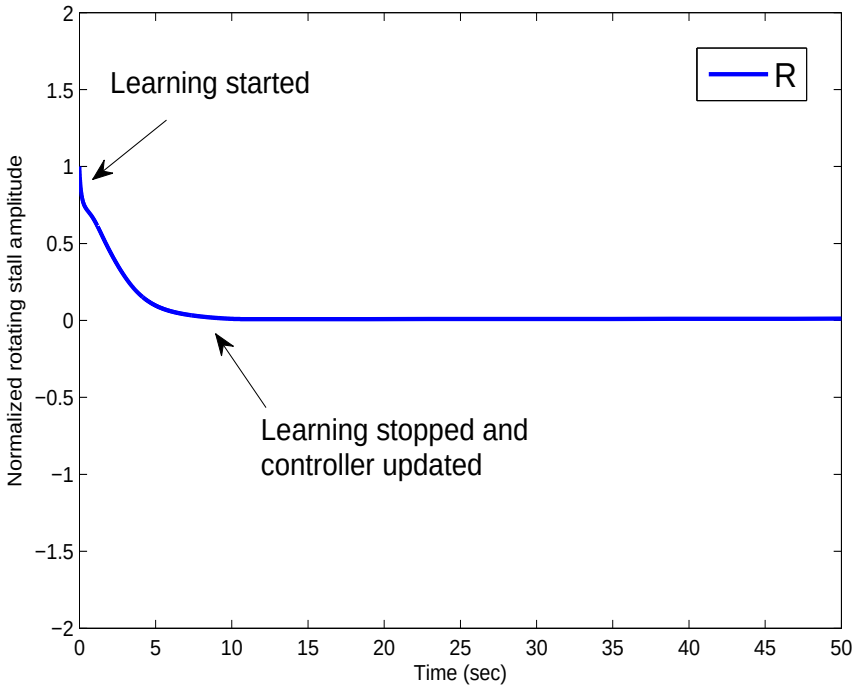

Figure 5.3: Trajectory of the normalized rotating stall amplitude.

input power, the EMF in the quadrature axis, and the control input to the system, respectively; x_d, x_T, and x_L are the reactance of the direct axis, the transformer, and the transmission line, respectively. x'_d is the direct axis transient reactance, K_G is the regulation constant, H is the inertia constant, and T'_{d0} is the direct axis transient short-circuit time constant, V_s is the voltage on the infinite-bus.

Define the following transformations

$$w = \frac{V_s}{x_{d\Sigma}}\left(E'_q - E_{q_0}\right) \tag{5.91}$$

$$x_1 = \delta - \delta_0 \tag{5.92}$$

$$x_2 = \omega \tag{5.93}$$

$$z = P_m - P_0 \tag{5.94}$$

where constants δ_0, P_0, and E_{q_0} denote the steady-state values of the rotor angle, the

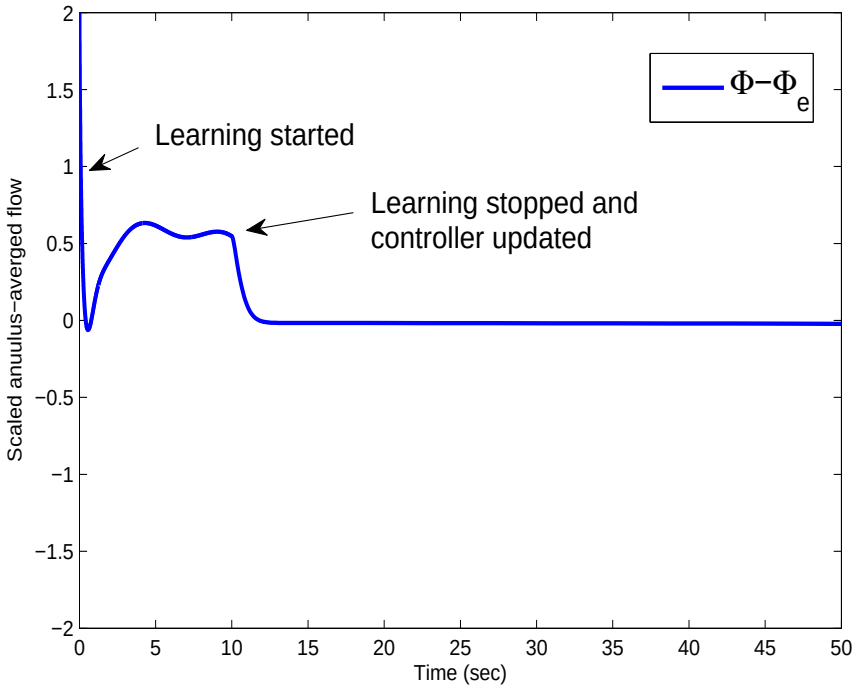

Figure 5.4: Trajectory of the mass flow.

mechanical power input, and the EMF, respectively.

This system can be converted to

$$\dot{w} = -a_1 w + a_2 \sin(\frac{x_1}{2} + a_3) \sin\frac{x_1}{2} \tag{5.95}$$

$$\dot{x}_1 = x_2 \tag{5.96}$$

$$\dot{x}_2 = -b_1 x_2 - b_2 \cos(\frac{x_1}{2} + a_3) \sin\frac{x_1}{2} + b_3 \left[z - w \sin(x_1 + a_3)\right] \tag{5.97}$$

$$\dot{z} = -c_1 z - c_2 x_2 + u \tag{5.98}$$

where $a_1 = \frac{1}{T'_d}$, $a_2 = \frac{x_d - x'_d}{T_{d0}} \frac{V_s^2}{x'_{d\Sigma} x_{d\Sigma}}$, $a_3 = \delta_0$, $b_1 = \frac{D}{2H}$, $b_2 = \frac{\omega_0}{H} \frac{V_s}{x_{d\Sigma}} E_{q_0}$, $b_3 = \frac{\omega_0}{2H}$, $c_1 = \frac{1}{T_G}$, $c_2 = K_G$.

For simulation purpose, the parameters are specified as follows: $D = 5$, $H = 4$, $\omega_0 = 314.159$ rad/s, $x_T = 0.127$, $x_L = 0.4853$, $x_d = 1.863$, $x'_d = 0.257$, $T'_{d0} = 0.5s$,

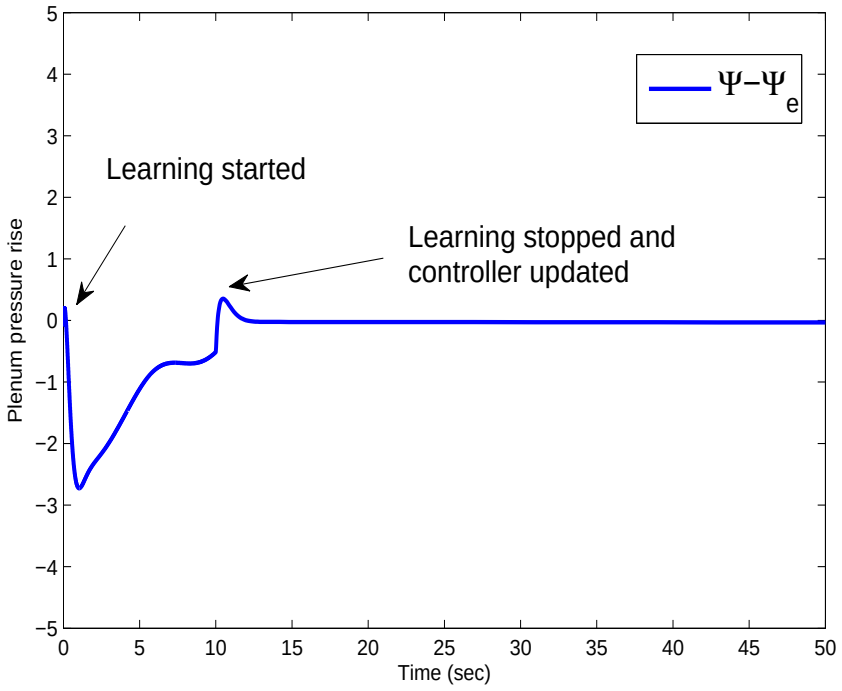

Figure 5.5: Trajectory of the plenum pressure rise.

$\delta_0 = 1.2566$ rad, $V_s = 1$p.u., $T_T = 2s$, $K_G = 1$, $K_T = 1$, $T_G = 0.2s$. In addition, notice that $x_{ds} = x_T + x_L + x_d$, $x'_{ds} = x_T + x_L + x'_d$. We set $Q(x_1, x_2) = 10x_1^2 + x_2^2$, $r = 1$, and we pick $\rho(s) = 1$ for robust redesign. The basis functions are selected to be polynomials of x_1, x_2, and z with order less or equal to four.

Suppose the bounds of the parameters are given as $0.5 < a_1 \leq 1$, $0 < a_2, 1.5$, $0 < a_3 \leq 1$, $0.5 < b_1 \leq 1$, $0 < b_2 \leq 150$, $0 < b_3 \leq 50$, $0 < c_1 \leq 1$, and $0 < c_2 \leq 0.1$. Then, we select the initial control policy to be $u = -x_1$, with $V(w, x_1, x_2, z) = w^2 + x_1^2 + x_2^2 + z^2$ the Lyapunov function of the closed-loop system. The exploration noise is set to be $e = 0.001\sin(t)$. The initial virtual control policy is $z = -x_1$. The algorithm stopped after nine iterations, when the stopping criterion in (5.75) with $\epsilon_1 = 0.01$ is satisfied. The RADP learning is finished within two seconds. The initial cost function and the cost function we obtained from phase-one learning are shown in Figure 5.11.

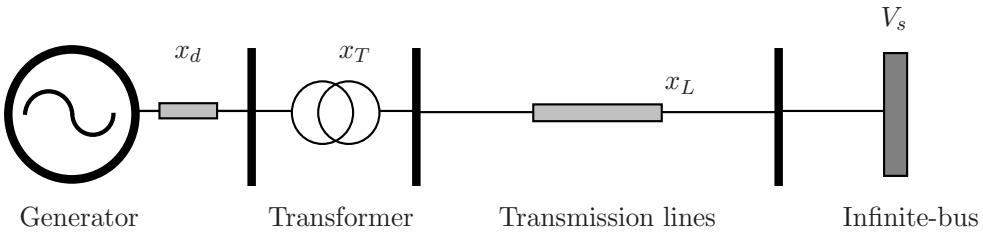

Figure 5.6: One-machine infinite-bus synchronous generator with speed governor

It is worth pointing out that, attenuating the oscillation in the power frequency is an important issue in power system control. From the simulation results shown in Figures 5.7-5.10, we see that the post-learning performance of the system is remarkably improved and the oscillation is attenuated.

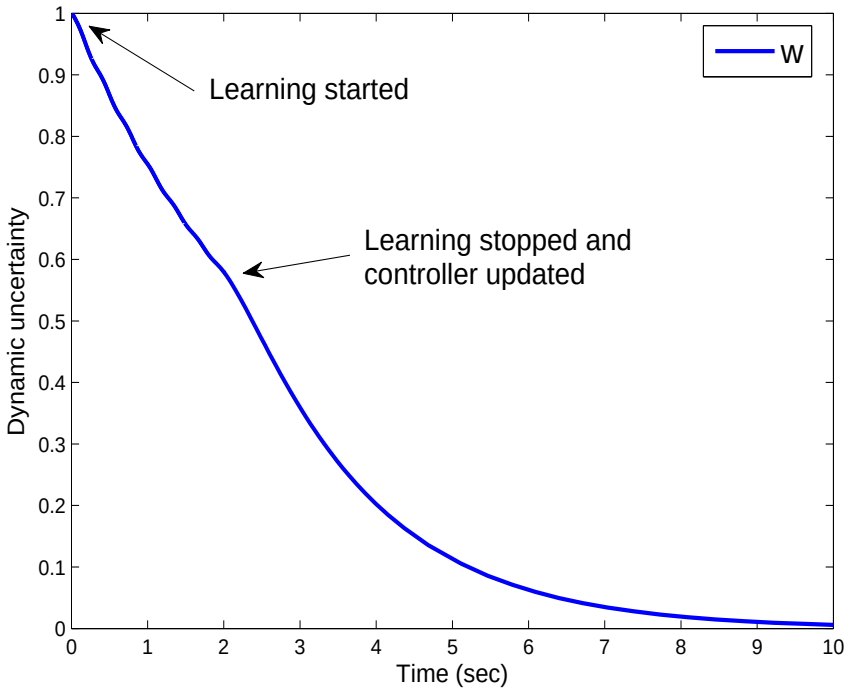

Figure 5.7: Trajectory of the dynamic uncertainty.

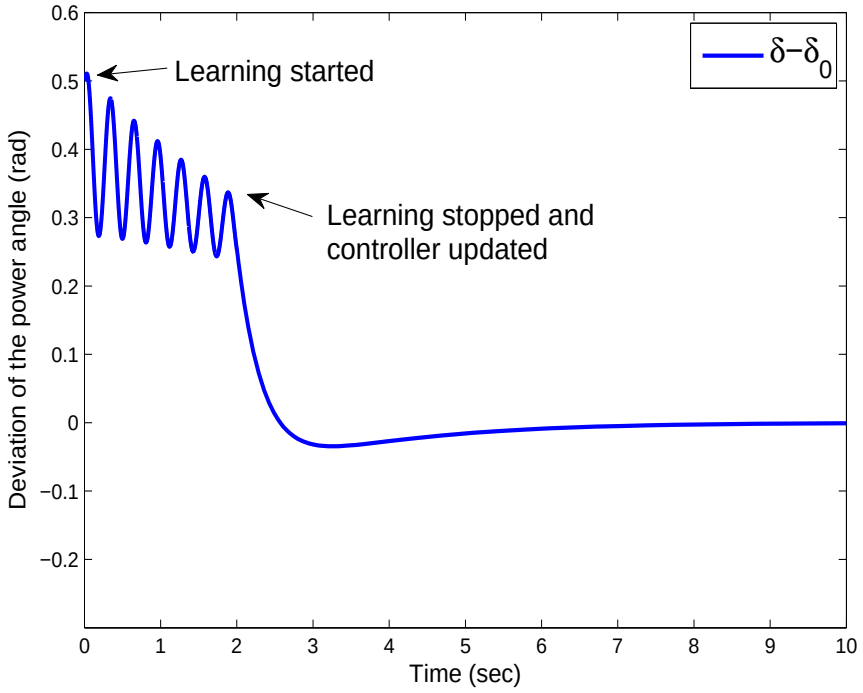

Figure 5.8: Trajectory of the deviation of the rotor angle.

5.5 Conclusions

In this chapter, neural-network-based robust and adaptive optimal control design has been studied for nonlinear systems with dynamic uncertainties. Both the matched and the unmatched cases are studied. We have presented for the first time a recursive, online, adaptive optimal controller design when dynamic uncertainties, characterized by input-to-state stable systems with unknown order and states/dynamics, are taken into consideration. We have achieved this goal by integration of approximate/adaptive dynamic programming (ADP) theory and tools recently developed within the nonlinear control community. Systematic RADP-based online learning algorithms have been developed to obtain semi-globally stabilizing controllers with optimality properties. The effectiveness of the proposed methodology has been validated by its application to the robust optimal control policy designs for a jet engine and a one-machine power system.

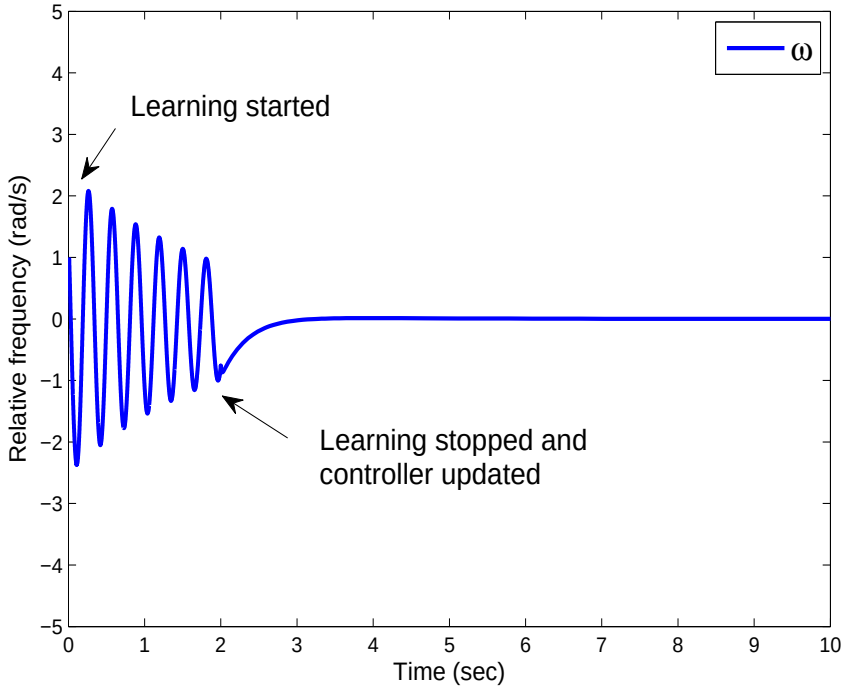

Figure 5.9: Trajectory of the relative frequency.

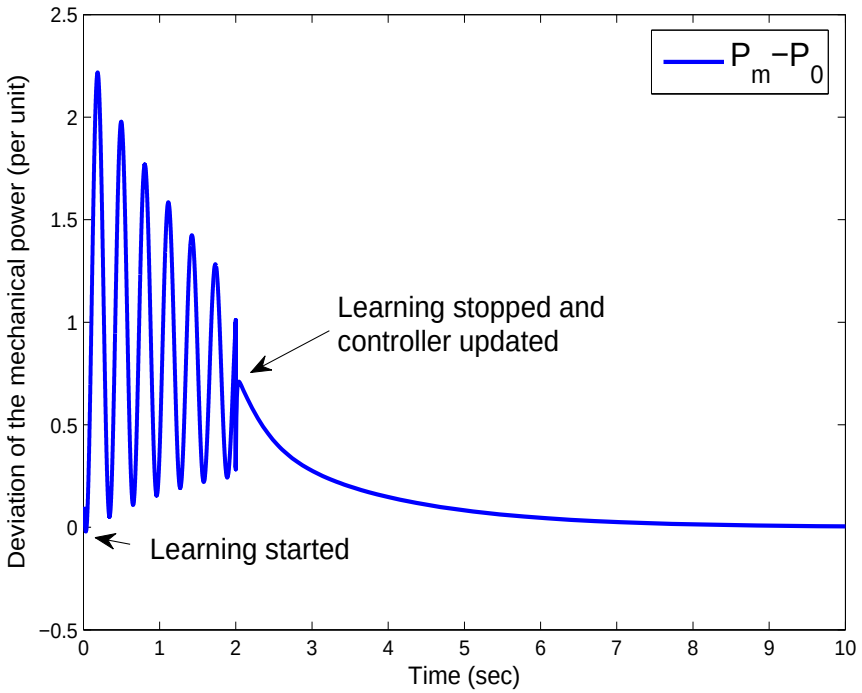

Figure 5.10: Trajectory of the deviation of the mechanical power.

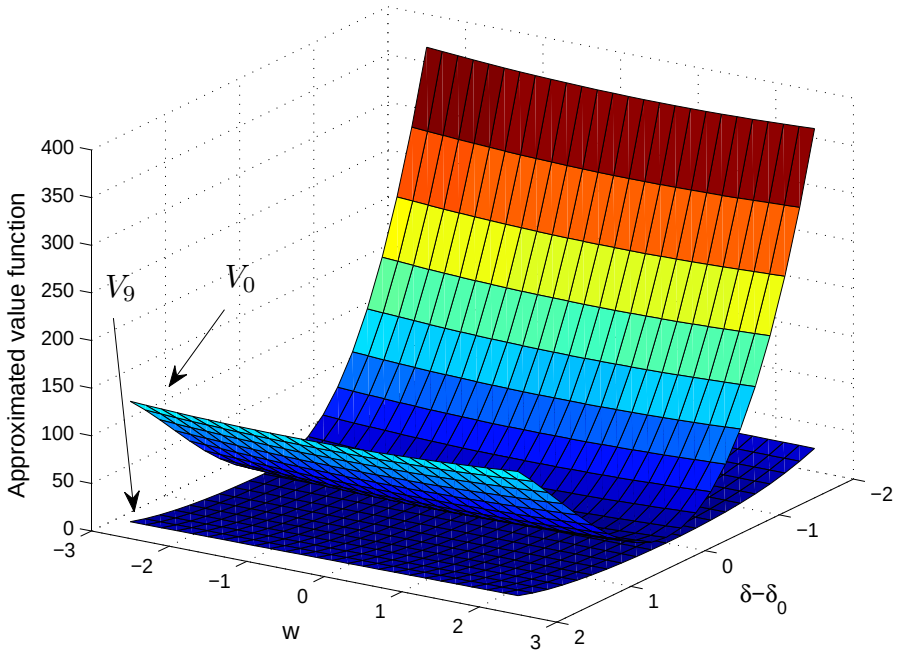

Figure 5.11: Approximated cost function.

Chapter 6

Global robust adaptive dynamic programming via sum-of-squares-programming

In the previous section, neural networks are employed to achieve online approximation of the cost function and the control policy. Actually, neural networks are widely used in the previous ADP architecture. However, although they can be used as universal approximators [57], [124], there are at least two major limitations for ADP-based online implementations. First, in order to approximate unknown functions with high accuracy, a large number of basis functions comprising the neural network are usually required. Hence, it may incur a huge computational burden for the learning system. Besides, it is not trivial to specify the type of basis functions, when the target function to be approximated is unknown. Second, neural network approximations generally are effective only on some compact sets, but not in the entire state space. Therefore, the resultant control policy may not provide global asymptotic stability for the closed-loop system. In addition, the compact set, on which the uncertain functions of interest are to be approximated, has to be carefully quantified before one applies the online

learning method, such that stability can be assured during the learning process [75].

The main purpose of this chapter is to develop a novel RADP methodology to achieve global and adaptive suboptimal stabilization of uncertain continuous-time nonlinear system via online learning. As the first contribution of this chapter, an optimization problem, of which the solutions can be easily parameterized, is proposed to relax the problem of solving the Hamilton-Jacobi-Bellman (HJB) equation. This approach is similar to the relaxation method used in approximate dynamic programming for Markov decision processes (MDPs) with finite state space [32], and more generalized discrete-time systems [106, 178, 179, 137, 139, 153]. However, methods developed in these papers cannot be trivially extended to the continuous-time setting, or achieve global asymptotic stability of general nonlinear systems. The idea of relaxation was also used in nonlinear $\mathcal{H}_\infty$ control, where Hamilton-Jacobi inequalities are used for nonadaptive systems [52, 169].

The second contribution of the chapter is to propose a relaxed policy iteration method. For polynomial systems, we formulate each iteration step of the proposed policy iteration as a sum of squares (SOS) program [125, 17], and give its equivalent semidefinite programming (SDP) problem [170]. For nonlinear functions that cannot be parameterized using a basis of polynomials, a less conservative sufficient condition is derived to decide their non-negativity by examining the coefficients. Thus, the proposed policy iteration is formulated as a more general SDP problem. It is worth pointing out that, different from the inverse optimal control design [96], the proposed method finds directly a suboptimal solution to the original optimal control problem.

The third contribution is an online learning method that implements the proposed iterative schemes using only the real-time online measurements, when the perfect system knowledge is not available. This method can be regarded as a nonlinear variant of our recent work for continuous-time linear systems with completely unknown system dynamics [68]. This method distinguishes from previously known nonlinear ADP

methods in that the neural network approximation is avoided for computational benefits and that the resultant control policy is globally stabilizing, instead of semiglobally or locally stabilizing.

The fourth contribution of this chapter is the robust redesign of the approximate suboptimal control policy, such that the overall system can be globally asymptotically stable in the presence of dynamic uncertainties. As in the previous chapter, the key strategy is to integrate the idea of gain assignment [83, 129] and the Lyapunov-based small-gain theorem [80] in nonlinear control theory.

The remainder of this chapter is organized as follows. 6.1 formulates the problem and introduces some basic results in nonlinear optimal control and nonlinear optimization. Section 6.2 relaxes the problem of solving an HJB equation to an optimization problem. Section 6.3 develops a relaxed policy iteration technique for polynomial systems based on sum of squares (SOS) programming [17]. Section 6.4 develops an online learning method for applying the proposed policy iteration, when the system dynamics are not known exactly. Section 6.5 extends the proposed method to deal with more generalized nonlinear systems. Section 6.6 develops a way to redesign the suboptimal control policy. Section 6.7 examines three numerical examples to validate the efficiency and effectiveness of the proposed method. Section 6.8 gives concluding remarks.

6.1 Problem formulation and preliminaries

In this section, we first formulate the control problem to be studied in the chapter. Then, we introduce some basic tools in nonlinear optimal control and optimization theories, based on which our main results in this chapter will be developed.

6.1.1 Problem formulation

Consider the nonlinear system

$$\dot{x} = f(x) + g(x)u \tag{6.1}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the control input, $f(x)$ and $g(x)$ are locally Lipschitz functions with $f(0) = 0$.

In conventional optimal control theory [103], the common objective is to find a control policy u that minimizes certain performance index. In this chapter, it is specified as follows.

$$J(x_0, u) = \int_0^\infty r(x(t), u(t))dt, \quad x(0) = x_0 \tag{6.2}$$

where $r(x, u) = Q(x) + u^T Ru$, with $Q(x)$ a positive definite function, and R is a symmetric positive definite matrix. Notice that, the purpose of specifying $r(x, u)$ in this form is to guarantee that an optimal control policy can be explicitly determined, if it exists.

Assumption 6.1.1. *Consider system* (6.1). *There exist a function* $V_0 \in \mathcal{P}$ *and a feedback control policy* u_1, *such that*

$$\mathcal{L}(V_0(x), u_1(x)) \geq 0, \quad \forall x \in \mathbb{R}^n \tag{6.3}$$

where, for any $V \in \mathcal{C}^1$ *and* $u \in \mathbb{R}^m$,

$$\mathcal{L}(V, u) = -\nabla V^T(x)(f(x) + g(x)u) - r(x, u). \tag{6.4}$$

Under Assumption 6.1.1, the closed-loop system composed of (6.1) and $u = u_1(x)$ is globally asymptotically stable at the origin, with a well-defined Lyapunov function

V_0. With this property, u_1 is also known as an *admissible* control policy [6], implying that the cost $J(x_0, u_1)$ is finite, $\forall x_0 \in \mathbb{R}^n$. Indeed, integrating both sides of (6.3) along the trajectories of the closed-loop system composed of (6.1) and $u = u_1(x)$ on the interval $[0, +\infty)$, it is easy to show that

$$J(x_0, u_1) \leq V_0(x_0), \quad \forall x_0 \in \mathbb{R}^n. \tag{6.5}$$

6.1.2 Optimality and stability

Here, we recall a basic result connecting optimality and global asymptotic stability in nonlinear systems [141]. To begin with, let us give the following assumption.

Assumption 6.1.2. *There exists $V^{\mathrm{o}} \in \mathcal{P}$, such that the Hamilton-Jacobi-Bellman (HJB) equation holds*

$$\mathcal{H}(V^{\mathrm{o}}) = 0 \tag{6.6}$$

where

$$\mathcal{H}(V) \;=\; \nabla V^T(x) f(x) + Q(x) - \frac{1}{4} \nabla V^T(x) g(x) R^{-1} g^T(x) \nabla V(x).$$

Under Assumption 6.1.2, it is easy to see that V^{o} is a well-defined Lyapunov function for the closed-loop system comprised of (6.1) and

$$u^{\mathrm{o}}(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla V^{\mathrm{o}}(x). \tag{6.7}$$

Hence, this closed-loop system is globally asymptotically stable at $x = 0$ [86]. Then, according to [141, Theorem 3.19], u^{o} is the optimal control policy, and the

value function $V^{\mathrm{o}}(x_0)$ gives the optimal cost at the initial condition $x(0) = x_0$, i.e.,

$$V^{\mathrm{o}}(x_0) = \min_u J(x_0, u) = J(x_0, u^{\mathrm{o}}), \quad \forall x_0 \in \mathbb{R}^n. \tag{6.8}$$

It can also be shown that V^{o} is the unique solution to the HJB equation (9.8) with $V^{\mathrm{o}} \in \mathcal{P}$. Indeed, let $\hat{V} \in \mathcal{P}$ be another solution to (9.8). Then, by Theorem 3.19 in [141], along the solutions of the closed-loop system composed of (6.1) and $u = \hat{u} = -\frac{1}{2}R^{-1}g^T\nabla\hat{V}$, it follows that

$$\hat{V}(x_0) \;\; = \;\; V^{\mathrm{o}}(x_0) - \int_0^\infty |u^{\mathrm{o}} - \hat{u}|_R^2 dt, \qquad \forall x_0 \in \mathbb{R}^n. \tag{6.9}$$

Finally, comparing (6.8) and (6.9), we conclude that $V^{\mathrm{o}} = \hat{V}$.

6.1.3 Conventional policy iteration

The above-mentioned result implies that, if there exists a class-$\mathcal{P}$ function which solves the HJB equation (9.8), an optimal control policy can be obtained. However, the nonlinear HJB equation (9.8) is almost impossible to be solved analytically in general. As a result, numerical methods are developed to approximate the solution. In particular, the following policy iteration method is widely used [136].

Algorithm 6.1.1 Conventional policy iteration

1: *Policy evaluation:* For $i = 1, 2, \cdots$, solve for the cost function $V_i(x) \in \mathcal{C}^1$, with $V_i(0) = 0$, from the following partial differential equation.

$$\mathcal{L}(V_i(x), u_i(x)) = 0. \tag{6.10}$$

2: *Policy improvement:* Update the control policy by

$$u_{i+1}(x) = -\frac{1}{2}R^{-1}g^T(x)\nabla V_i(x). \tag{6.11}$$

The following result is a trivial extension of [136, Theorem 4], in which $g(x)$ is a constant matrix and only stabilization over compact set is considered.

Theorem 6.1.1. *Suppose Assumptions 6.1.1 and 6.1.2 hold, and the solution* $V_i(x) \in \mathcal{C}^1$ *satisfying* (6.10) *exists, for* $i = 1, 2, \cdots$. *Let* $V_i(x)$ *and* $u_{i+1}(x)$ *be the functions generated from (6.10) and (6.11). Then, the following properties hold,* $\forall i = 0, 1, \cdots$.

1) $V^o(x) \leq V_{i+1}(x) \leq V_i(x)$, $\forall x \in \mathbb{R}^n$;

2) u_{i+1} *is globally stabilizing;*

3) $J(x_0, u_i)$ *is finite,* $\forall x_0 \in \mathbb{R}^n$;

4) $\{V_i(x)\}_{i=0}^{\infty}$ *and* $\{u_i(x)\}_{i=1}^{\infty}$ *converge pointwise to* $V^o(x)$ *and* $u^o(x)$, *respectively.*

Notice that finding the analytical solution to (6.10) is still non-trivial. Hence, in practice, the solution is approximated using, for example, neural networks or Galerkin's method [6]. When the precise knowledge of f or g is not available, ADP-based online approximation method can be applied to compute numerically the cost functions via online data [172], [75].

In general, approximation methods can only give acceptable results on some compact set in the state space, but cannot be used to achieve global stabilization. In addition, in order to reduce the approximation error, huge computational complexity is almost inevitable. These facts may affect the effectiveness of the previously developed ADP-based online learning methods.

6.1.4 Semidefinite programming and sum-of-squares programming

A standard semidefinite programming (SDP) problem can be formulated as the following problem of minimizing a linear function of a variable $y \in \mathbb{R}^{n_0}$ subject to a linear matrix inequality.

Problem 6.1.1 (Semidefinite programming [170])**.**

$$\min_{y} \quad c^T y \tag{6.12}$$

$$F_0 + \sum_{i=1}^{n_0} y_i F_i \geq 0 \tag{6.13}$$

where $c \in \mathbb{R}^{n_0}$ is a constant column vector, and $F_0, F_1, \cdots, F_{n_0} \in \mathbb{R}^{m_0 \times m_0}$ are $n_0 + 1$ symmetric constant matrices.

SDPs can be solved using several commercial or non-commercial software packages, such as the Matlab-based solver CVX [48].

Definition 6.1.1 (Sum of squares [17])**.** *A polynomial $p(x) \in \mathbb{R}[x]_{0,2d}$ is a sum of squares (SOS) if there exist $q_1, q_2, \cdots, q_{m_0} \in \mathbb{R}[x]_{0,d}$ such that*

$$p(x) = \sum_{i=1}^{m} q_i^2(x). \tag{6.14}$$

An SOS programming problem is a convex optimization problem of the following form

Problem 6.1.2 (SOS programming [17])**.**

$$\min_{y} \quad b^T y \tag{6.15}$$

$$\text{s.t.} \quad p_i(x; y) \text{ are SOS}, \ i = 1, 2, \cdots, k_0 \tag{6.16}$$

where $p_i(x; y) = a_{i0}(x) + \sum_{j=1}^{n_0} a_{ij}(x) y_j$, and $a_{ij}(x)$ are given polynomials in $\mathbb{R}[x]_{0,2d}$.

In [17, p.74], it has been pointed out that SOS programs are in fact equivalent to SDPs. Indeed, the constraints (6.16) are equivalent to the existence of symmetric matrices $Q_i \geq 0$ satisfying

$$p_i(x; y) = [x]_{0,d}^T \, Q_i \, [x]_{0,d}, \quad i = 1, 2, \cdots, k_0. \tag{6.17}$$

Then, by extending and matching the coefficients of (6.17), the equations (6.17) reduce to linear equations between y and the entries of Q_i. As a result, Problem 6.1.1 is equivalent to an SDP problem in the variables of y and all the distinct entries of Q_i. This equivalence implies that SOS programs can be reformulated and solved as SDPs. The conversion from an SOS to an SDP can be performed either manually, or automatically using, for example, the Matlab toolbox SOSTOOLS [128], YALMIP [112], and Gloptipoly [53].

6.2 Suboptimal control with relaxed HJB equation

In general, it is extremely difficult to obtain the analytical solution to the HJB equation (9.8). Therefore, in this section we consider an auxiliary optimization problem, which allows us to obtain a suboptimal solution to the minimization problem (9.7) subject to (6.1). For simplicity, we will omit the arguments of functions whenever there is no confusion in the context.

Problem 6.2.1 (Relaxed optimal control problem)**.**

$$\min_{V} \quad \int_{\mathbb{R}^n} s(x)V(x)dx \tag{6.18}$$

$$\text{s.t.} \quad \mathcal{H}(V) \leq 0 \tag{6.19}$$

$$V \in \mathcal{P} \tag{6.20}$$

where $s(x)$ is a positive semidefinite function taking positive values only on some predefined compact set $\Omega \subset \mathbb{R}^n$.

Remark 6.2.1. *Notice that Problem 6.2.1 is called a relaxed problem of* (9.8)*. Indeed, if we restrict this problem by replacing the inequality constraint* (6.19) *with the equality constraint* (9.8)*, there will be only one feasible solution left and the objective function*

can thus be neglected. As a result, Problem 6.2.1 reduces to the problem of solving (9.8).

Remark 6.2.2. *The function $s(x)$ can also be recognized as the state-relevance weighting function [32]. It is easy to see that better approximation to the optimal cost function V^{o} in a particular region of state space can be achieved by assigning relatively higher weights to the region.*

Some useful facts about Problem 6.2.1 are given as follows.

Theorem 6.2.1. *Under Assumptions 6.1.1 and 6.1.2, the following hold.*

1) Problem 6.2.1 has a nonempty feasible set.

2) Let V be a feasible solution to Problem 6.2.1. Then, the control policy

$$\bar{u} = -\frac{1}{2}R^{-1}g^T\nabla V \tag{6.21}$$

is globally stabilizing.

3) For any $x_0 \in \mathbb{R}^n$, an upper bound of the cost of the closed-loop system comprised of (6.1) *and* (6.21) *is given by $V(x_0)$, i.e.,*

$$J(x_0, \bar{u}) \leq V(x_0). \tag{6.22}$$

4) Along the trajectories of the closed-loop system (6.1) *and* (6.21)*, the following inequalities hold for any $x_0 \in \mathbb{R}^n$:*

$$V(x_0) + \int_0^\infty \mathcal{H}(V(x(t)))dt \leq V^{\mathrm{o}}(x_0) \leq V(x_0). \tag{6.23}$$

5) V^{o} as defined in (6.8) *is a global optimal solution to Problem 6.2.1.*

Proof. 1) Define $u_0 = -\frac{1}{2}R^{-1}g^T\nabla V_0$. Then,

$$\begin{aligned}
\mathcal{H}(V_0) &= \nabla V_0^T (f + gu_0) + r(x, u_0) \\
&= \nabla V_0^T (f + gu_1) + r(x, u_1) \\
&\quad + \nabla V_0^T g(u_0 - u_1) + |u_0|_R^2 - |u_1|_R^2 \\
&= \nabla V_0^T (f + gu_1) + r(x, u_1) - |u_0 - u_1|_R^2 \\
&\leq 0
\end{aligned}$$

Hence, V_0 is a feasible solution to Problem 6.2.1.

2) To show global asymptotic stability, we only need to prove that V is a well-defined Lyapunov function for the closed-loop system composed of (6.1) and (6.21). Indeed, along the solutions of the closed-loop system, it follows that

$$\dot{V} = \nabla V^T(f + g\bar{u}) = \mathcal{H}(V) - r(x, \bar{u}) \leq -Q(x)$$

Therefore, the system is globally asymptotically stable at the origin [86].

3) Along the solutions of the closed-loop system comprised of (6.1) and (6.21), we have

$$\begin{aligned}
V(x_0) &= -\int_0^T \nabla V^T(f + g\bar{u})dt + V(x(T)) \\
&= \int_0^T [r(x, \bar{u}) - \mathcal{H}(V)]\, dt + V(x(T)) \\
&\geq \int_0^T r(x, \bar{u})dt + V(x(T)) \qquad (6.24)
\end{aligned}$$

By 2), $\lim_{T\to+\infty} V(x(T)) = 0$. Therefore, letting $T \to +\infty$, by (6.24) and (9.7), we have

$$V(x_0) \geq J(x_0, \bar{u}). \qquad (6.25)$$

4) By 3), we have

$$V(x_0) \geq J(x_0, \bar{u}) \geq \min_u J(x_0, \bar{u}) = V^{\mathrm{o}}(x_0). \tag{6.26}$$

Hence, the second inequality in (6.23) is proved.

On the other hand,

$$\begin{aligned}
\mathcal{H}(V) &= \mathcal{H}(V) - \mathcal{H}(V^{\mathrm{o}}) \\
&= (\nabla V - \nabla V^{\mathrm{o}})^T (f + g\bar{u}) + r(x, \bar{u}) - (\nabla V^{\mathrm{o}})^T g(u^{\mathrm{o}} - \bar{u}) - r(x, u^{\mathrm{o}}) \\
&= (\nabla V - \nabla V^{\mathrm{o}})^T (f + g\bar{u}) + |\bar{u} - u^{\mathrm{o}}|_R^2 \\
&\geq (\nabla V - \nabla V^{\mathrm{o}})^T (f + g\bar{u})
\end{aligned}$$

Integrating the above equation along the solutions of the closed-loop system (6.1) and (6.21) on the interval $[0, +\infty)$, we have

$$V(x_0) - V^{\mathrm{o}}(x_0) \leq -\int_0^{\infty} \mathcal{H}(V(x))dt. \tag{6.27}$$

5) By 3), for any feasible solution V to Problem 6.2.1, we have $V^{\mathrm{o}}(x) \leq V(x)$. So,

$$\int_{\mathbb{R}^n} s(x)V^{\mathrm{o}}(x)dx \leq \int_{\mathbb{R}^n} s(x)V(x)dx \tag{6.28}$$

which implies that V^{o} is a global optimal solution.

The proof is therefore complete. □

Remark 6.2.3. *A feasible solution V to Problem 6.2.1 may not necessarily be the true cost function associated with the control policy $\bar{u}$ defined in* (6.21)*. However, by Theorem 6.2.1, we see V can be viewed as an upper bound or an overestimate of the actual cost, inspired by the concept of underestimator in [178]. Further, V serves as a Lyapunov function for the closed-loop system and can be more easily parameterized*

than the actual cost function. For simplicity, V is still called the cost function, in the remainder of the chapter.

6.3 SOS-based policy iteration for polynomial systems

The inequality constraint (6.19) contained in Problem 6.2.1 provides us the freedom of specifying desired analytical forms of the cost function. However, solving (6.19) is non-trivial in general, even for polynomial systems (see, for example, [28], [44], [121], [147], [197]). Indeed, for any polynomial with degree no less than four, deciding its non-negativity is an NP-hard problem [125]. Fortunately, due to the developments in sum of squares (SOS) programming [17, 125], the computational burden can be significantly reduced, if inequality constraints can be restricted to SOS constraints. The purpose of this section is to develop a novel policy iteration method for polynomial systems using SOS-based methods [17, 125].

6.3.1 Polynomial parametrization

To study polynomial systems, let us first give the following Assumption.

Assumption 6.3.1. *There exist integers $d > 0$, $d_1 \geq 0$, and $r > 0$, such that*

1. *all entries of $f(x)$ belong to $\mathbb{R}[x]_{1,d}$ and all entries of $g(x)$ belong to $\mathbb{R}[x]_{0,d_1}$;*
2. *in addition to being positive definite, the weighting function $Q(x)$ satisfies $Q(x) \in \mathbb{R}[x]_{2,2d}$;*
3. *there exist a nonlinear mappings $V_0 : \mathbb{R}^n \to \mathbb{R}$ and a feedback control policy $u_1 : \mathbb{R}^n \to \mathbb{R}^m$, such that $V_0 \in \mathbb{R}[x]_{2,2r} \cap \mathcal{P}$, $u_1 \in \mathbb{R}[x]^m_{1,d}$, and $\mathcal{L}(V_0, u_1)$ is SOS; and*

4. the inequality holds:

$$d \geq (2r-1) + d_1. \tag{6.29}$$

Remark 6.3.1. *It is easy to see that, Assumption 6.3.1 holds only if Assumption 6.1.1 holds. In addition, under Assumption 6.3.1, we know that* $\mathcal{L}(V_0, u_1) \in \mathbb{R}[x]_{2,2d}$. *Indeed, by* (6.29)*, it follows that* $\mathcal{L}(V_0, u_1) \in \mathbb{R}[x]_{2,\max\{(2r-1)+d+(d_1+d),2d\}} = \mathbb{R}[x]_{2,2d}$.

Remark 6.3.2. *Notice that the inequality* (6.29) *can be assumed without loss of generality. Indeed, if it does not hold, we can always find* $\tilde{d} > \max\{d, (2r-1)+d_1\}$. *As a result, Assumption 6.3.1 holds with d replaced by* $\tilde{d}$.

For notational simplicity, we denote the dimensions of $[x]_{1,r}$, $[x]_{1,d}$, $[x]_{2,2r}$, and $[x]_{2,2d}$ by n_r, n_d, n_{2r}, and n_{2d}, respectively. By [17], we know $n_r = \binom{n+r}{r} - 1$, $n_d = \binom{n+d}{d} - 1$, $n_{2r} = \binom{n+2r}{2r} - n - 1$, and $n_{2d} = \binom{n+2d}{2d} - d - 1$.

6.3.2 SOS-programming-based policy iteration

Now, we are ready to propose a relaxed policy iteration scheme. Similar as in other policy-iteration-based iterative schemes, an initial globally stabilizing (and admissible) control policy has been assumed in Assumption 6.3.1.

Remark 6.3.3. *The optimization problem* (6.30)-(6.33) *is a well defined SOS program [17]. Indeed, the objective function* (6.30) *is linear in p, since for any* $V = p^T[x]_{2,2r}$, *we have* $\int_{\mathbb{R}^n} s(x)V(x)dx = c^T p$, *with* $c = \int_{\mathbb{R}^n} s(x)[x]_{2,2r}dx$. *In addition, notice that since the objective function is nonnegative, its optimal value must be finite.*

Theorem 6.3.1. *Under Assumptions 6.1.2 and 6.3.1, the following are true, for* $i = 1, 2, \cdots$.

1) The SOS program (6.30)-(6.33) *has a nonempty feasible set.*

Algorithm 6.3.1 SOS-based policy iteration

1: *Policy evaluation:* For $i = 1, 2, \cdots$, solve for an optimal solution $p_i \in \mathbb{R}^{n_{2r}}$ to the following optimization program, and denote $V_i = p_i^T [x]_{2,2r}$.

$$\begin{aligned} \min_{p \in \mathbb{R}^{n_{2r}}} \quad & \int_{\mathbb{R}^n} s(x) V(x) dx && (6.30) \\ \text{s.t.} \quad V &:= p^T [x]_{2,2r} && (6.31) \\ \mathcal{L}(V, u_i) &\in \Sigma_{2,2d} && (6.32) \\ V_{i-1} - V &\in \Sigma_{2,2r} && (6.33) \end{aligned}$$

where $\Sigma_{2,2d}$ and $\Sigma_{2,2r}$ denote the sets of all SOS polynomials in $\mathbb{R}[x]_{2,2d}$ and $\mathbb{R}[x]_{2,2r}$, respectively.

2: *Policy improvement:* Update the control policy by

$$u_{i+1} = -\frac{1}{2} R^{-1} g^T \nabla V_i. \tag{6.34}$$

Then, go to Step 1) with i replaced by $i + 1$.

2) The closed-loop system comprised of (6.1) *and* $u = u_i$ *is globally asymptotically stable at the origin.*

3) $V_i \in \mathcal{P}$*. In particular, the following inequalities hold:*

$$V^{\mathrm{o}}(x_0) \leq V_i(x_0) \leq V_{i-1}(x_0), \quad \forall x_0 \in \mathbb{R}^n. \tag{6.35}$$

4) There exists $V^*(x)$ *satisfying* $V^*(x) \in \mathbb{R}[x]_{2,2r} \cap \mathcal{P}$*, such that, for any* $x_0 \in \mathbb{R}^n$*,* $\lim_{i \to \infty} V_i(x_0) = V^*(x_0)$*.*

5) Along the solutions of the system (6.1) *with*

$$u^* = -\frac{1}{2} R^{-1} g^T \nabla V^*, \tag{6.36}$$

the following inequalities hold:

$$0 \leq V^*(x_0) - V^{\mathrm{o}}(x_0) \leq -\int_0^\infty \mathcal{H}(V^*(x(t))) dt. \tag{6.37}$$

Proof. 1) We prove by mathematical induction.

i) Suppose $i = 1$, under Assumption 6.3.1, we know $\mathcal{L}(V_0, u_1) \in \Sigma_{2,2d}$. Hence, $V = V_0$ is a feasible solution to the problem (6.30)-(6.33).

ii) Let $u_{j-1} \in \mathbb{R}[x]^m_{1,d}$, and $V = V_{j-1}$ be an optimal solution to the problem (6.30)-(6.33) with $i = j - 1 > 1$. We show that $V = V_{j-1}$ is a feasible solution to the same problem with $i = j$.

Indeed, by definition,

$$u_j = -\frac{1}{2}R^{-1}g^T\nabla V_{j-1} \in \mathbb{R}[x]^m_{1,d},$$

and

$$\begin{aligned}\mathcal{L}(V_{j-1}, u_j) &= -\nabla V_{j-1}^T(f + gu_j) - r(x, u_j)\\ &= \mathcal{L}(V_{j-1}, u_{j-1}) - \nabla V_{j-1}^T g(u_j - u_{j-1}) + u_{j-1}^T R u_{j-1} - u_j^T R u_j\\ &= \mathcal{L}(V_{j-1}, u_{j-1}) + |u_j - u_{j-1}|^2_R.\end{aligned}$$

Under the induction assumption, we know $V_{j-1} \in \mathbb{R}[x]_{2,2r}$, $u_{j-1} \in \mathbb{R}[x]^m_{1,d}$, and $\mathcal{L}(V_{j-1}, u_j) \in \Sigma_{2,2d}$. Hence, $\mathcal{L}(V_{j-1}, u_j) \in \Sigma_{2,2d}$. As a result, V_{j-1} is a feasible solution to the SOS program (6.30)-(6.33) with $i = j$.

2) Again, we prove by induction.

i) Suppose $i = 1$, under Assumption 6.3.1, u_1 is globally stabilizing. Also, we can show that $V_1 \in \mathcal{P}$. Indeed, for each $x_0 \in \mathcal{R}^n$ with $x_0 \neq 0$, we have

$$V_1(x_0) \geq \int_0^\infty r(x, u_1)dt > 0. \tag{6.38}$$

By (6.38) and the constraint (6.33), under Assumption 6.1.2 it follows that

$$V^{\mathrm{o}} \leq V_1 \leq V_0. \tag{6.39}$$

Since both V^{o} and V_0 are assumed to belong to $\mathcal{P}$, we conclude that $V_1 \in \mathcal{P}$.

ii) Suppose u_{i-1} is globally stabilizing, and $V_{i-1} \in \mathcal{P}$ for $i > 1$. Let us show that u_i is globally stabilizing, and $V_i \in \mathcal{P}$.

Indeed, along the solutions of the closed-loop system composed of (6.1) and $u = u_i$, it follows that

$$\dot{V}_{i-1} \;=\; \nabla V_{i-1}^T(f + gu_i) = -\mathcal{L}(V_{i-1}, u_i) - r(x, u_i) \leq -Q(x).$$

Therefore, u_i is globally stabilizing, since V_{i-1} is a well-defined Lyapunov function for the system. In addition, we have

$$V_i(x_0) \geq \int_0^\infty r(x, u_i)dt > 0, \quad \forall x_0 \neq 0. \tag{6.40}$$

Similarly as in (6.39), we can show

$$V^{\mathrm{o}}(x_0) \leq V_i(x_0) \leq V_{i-1}(x_0), \quad \forall x_0 \in \mathbb{R}^n, \tag{6.41}$$

and conclude that $V_i \in \mathcal{P}$.

3) The two inequalities have been proved in (6.41).

4) By 3), for each $x \in \mathbb{R}^n$, the sequence $\{V_i(x)\}_{i=0}^\infty$ is monotonically decreasing with 0 as its lower bound. Therefore, the limit exists, i.e., there exists $V^*(x)$, such that $\lim\limits_{i\to\infty} V_i(x) = V^*(x)$. Let $\{p_i\}_{i=1}^\infty$ be the sequence such that $V_i = p_i^T[x]_{2,2r}$. Then, we know $\lim\limits_{i\to\infty} p_i = p^* \in \mathbb{R}^{n_{2r}}$, and therefore $V^* = p^{*T}[x]_{2,2r}$. Also, it is easy to show $V^{\mathrm{o}} \leq V^* \leq V_0$. Hence, $V^* \in \mathbb{R}[x]_{2,2r} \cap \mathcal{P}$.

5) By 4), we know

$$\mathcal{H}(V^*) = -\mathcal{L}(V^*, u^*) \leq 0, \tag{6.42}$$

which implies that V^* is a feasible solution to Problem 6.2.1. Then, the inequalities in (5) can be obtained by the fourth property in Theorem 6.2.1.

The proof is thus complete. □

6.3.3 An equivalent SDP implementation

According to the equivalence between SOS programs and SDPs, the SOS-based policy iteration can be reformulated as SDPs. Notice that we can always find two linear mappings $\iota : \mathbb{R}^{n_{2r}} \times \mathbb{R}^{m\times n_r} \to \mathbb{R}^{n_{2d}}$ and $\kappa : \mathbb{R}^{n_{2r}} \to \mathbb{R}^{m\times n_r}$, such that given $p \in \mathbb{R}^{n_{2r}}$ and $K \in \mathbb{R}^{m\times n_r}$,

$$\iota(p,K)^T[x]_{2,2d} = \mathcal{L}(p^T[x]_{2,2r}, K[x]_{1,2r-1}) \tag{6.43}$$

$$\kappa(p)^T[x]_{1,2r-1} = -\frac{1}{2}R^{-1}g^T\nabla(p^T[x]_{2,2r}) \tag{6.44}$$

Then, by properties of SOS constraints [17], the polynomial $\iota(p,K)^T[x]_{2,2d}$ is SOS if and only if there exists a symmetric and positive semidefinite matrix $L \in \mathbb{R}^{n_d\times n_d}$, such that

$$\iota(p,K)^T[x]_{2,2d} = [x]_{1,d}^T L[x]_{1,d}. \tag{6.45}$$

Furthermore, there exist linear mappings $M_P : \mathbb{R}^{n_r\times n_r} \to \mathbb{R}^{n_{2r}}$ and $M_L : \mathbb{R}^{n_d\times n_d} \to \mathbb{R}^{n_{2d}}$, such that, for any vectors $p \in \mathbb{R}^{n_{2r}}$, $l \in \mathbb{R}^{n_{2d}}$, and symmetric matrices $P \in \mathbb{R}^{n_r\times n_r}$ and $L \in \mathbb{R}^{n_d\times n_d}$, the following implications are true.

$$p^T[x]_{2,2r} = [x]_{1,r}^T P[x]_{1,r} \Longleftrightarrow p = M_P(P) \tag{6.46}$$

$$l^T[x]_{2,2d} = [x]_{1,d}^T L[x]_{1,d} \Longleftrightarrow l = M_L(L) \tag{6.47}$$

Under Assumptions 6.1.2 and 6.3.1, the proposed policy iteration can be reformulated as follows.

Algorithm 6.3.2 SDP-based policy iteration

1: Let $i = 1$. Let $p_0 \in \mathbb{R}^{n_{2r}}$ and $K_1 \in \mathbb{R}^{m \times n_d}$ satisfy $V_0 = p_0^T[x]_{2,2r}$ and $u_1 = K_1[x]_{1,d}$.

2: Solve for an optimal solution $(p_i, P_i, L_i) \in \mathbb{R}^{n_{2r}} \times \mathbb{R}^{n_r \times n_r} \times \mathbb{R}^{n_d \times n_d}$ to the following problem.

$$\begin{aligned} & \min_{p,P,L} && c^T p && (6.48)\\ \text{s.t.} \quad & \iota(p, K_i) &=& M_L(L) && (6.49)\\ & p_{i-1} - p &=& M_P(P) && (6.50)\\ & P &=& P^T \geq 0 && (6.51)\\ & L &=& L^T \geq 0 && (6.52) \end{aligned}$$

where $c = \int_{\mathbb{R}^n} s(x)[x]_{2,2r} dx$.

3: Go to Step 2) with $K_{i+1} = \kappa(p_i)$ and i replaced by $i + 1$.

Remark 6.3.4. *The optimization problem* (6.48)-(6.52) *is a well-defined semidefinite programming (SDP) problem, since it has a linear objective function subject to linear equality and inequality constraints. It can be directly solved using, for example, Matlab-based solver CVX [48]. Also, it can be rewritten in the standard form* (6.12)-(6.13) *by equivalently replacing each equality constraint with two inequalities constraints, and by treating p and entries in P and L as the decision variables.*

Corollary 6.3.1. *Under Assumptions 6.1.2 and 6.3.1, the following are true.*

1. *The optimization problem* (6.48)-(6.52) *has at least one feasible solution, for $i = 1, 2, \cdots$.*

2. *Denote $V_i = p_i^T[x]_{2,2r}$, $u_{i+1} = K_i[x]_{1,d}$, for $i = 0, 1, \cdots$. Then, the sequences $\{V_i\}_{i=0}^{\infty}$ and $\{u_i\}_{i=1}^{\infty}$ satisfy the properties 2)-5) in Theorem 6.3.1.*

Proof. Given $p_i \in \mathbb{R}^{n_{2r}}$, there exist P_i and L_i such that (p_i, P_i, L_i) is a feasible solution to the optimization problem (6.48)-(6.52) if and only if p_i is a feasible solution to the SOS program (6.30)-(6.33). Therefore, by Theorem 6.3.1, 1) holds. In addition, since the two optimization problems share the identical objective function, we know that if (p_i, P_i, L_i) is a feasible solution to the optimization problem (6.48)-(6.52), p_i is also

an optimal solution to the SOS program (6.30)-(6.33). Hence, the corollary can be obtained from Theorem 6.3.1. □

6.4 Online learning via global adaptive dynamic programming

The proposed policy iteration method requires the perfect knowledge of the mappings ι and κ, which can be determined if f and g are known exactly. In practice, precise system knowledge may be difficult to obtain. Hence, in this section, we develop an online learning method based on the idea of ADP to implement the iterative scheme with real-time data, instead of identifying the system dynamics.

To begin with, consider the system

$$\dot{x} = f + g(u_i + e) \tag{6.53}$$

where u_i is a feedback control policy and e is a bounded time-varying function, known as the exploration noise, added for the learning purpose.

Lemma 6.4.1. *Consider system* (6.53). *Suppose u_i is a globally stabilizing control policy and there exists $V_{i-1} \in \mathcal{P}$, such that $\nabla V_{i-1}(f + gu_i) + u_i^T R u_i \leq 0$. Then, the system* (6.53) *is forward complete.*

Proof. Under Assumptions 6.1.2 and 6.3.1, by Theorem 6.3.1 we know $V_{i-1} \in \mathcal{P}$. Then, by completing the squares, it follows that

$$\begin{aligned}
\nabla V_{i-1}^T(f + gu_i + ge) &\leq -u_i^T R u_i - 2u_i^T R e \\
&= -|u_i + e|_R^2 + |e|_R^2 \\
&\leq |e|_R^2 \\
&\leq |e|_R^2 + V_{i-1}.
\end{aligned}$$

According to [3, Corollary 2.11], the system (6.53) is forward complete. □

By Lemma 6.4.1 and Theorem 6.3.1, we immediately have the following Proposition.

Proposition 6.4.1. *Under Assumptions 6.1.2 and 6.3.1, let u_i be a feedback control policy obtained at the i-th iteration step in the proposed policy iteration algorithm* (6.30)-(6.34) *and e be a bounded time-varying function. Then, the closed-loop system* (6.1) *with $u = u_i + e$ is forward complete.*

Suppose there exist $p \in \mathbb{R}^{n_{2r}}$ and $K_i \in \mathbb{R}^{m\times n_k}$ such that $V = p^T[x]_{2,2r}$ and $u_i = K_i[x]_{1,d}$. Then, along the solutions of the system (6.53), it follows that

$$\begin{aligned}
\dot{V} &= \nabla V^T (f + gu_i) + \nabla V^T Be \\
&= -r(x, u_i) - \mathcal{L}(V, u_i) + \nabla V^T ge \\
&= -r(x, u_i) - \mathcal{L}(V, u_i) + 2(\frac{1}{2}R^{-1}g^T\nabla V)^T Re \\
&= -r(x, u_i) - \iota(p, K_i)^T[x]_{2,2d} - 2[x]_{1,d}^T\kappa(p)^T Re \qquad (6.54)
\end{aligned}$$

where the last row is obtained by (6.43) and (6.44).

Now, integrating the terms in (6.99) over the interval $[t, t + \delta t]$, we have

$$\begin{aligned}
& p^T ([x(t)]_{2,2r} - [x(t + \delta t)]_{2,2r}) \\
= & \int_t^{t+\delta t} \left[r(x, u_i) + \iota(p, K_i)^T[x]_{2,2d} + 2[x]_{1,d}^T\kappa(p)^T Re\right] dt \qquad (6.55)
\end{aligned}$$

Eq. (6.55) implies that, given $p \in \mathbb{R}^{n_{2r}}$, $\iota(p, K_i)$ and $\kappa(p)$ can be directly calculated by using real-time online data, without knowing the precise knowledge of f and g.

Indeed, define

$$\begin{aligned}
\sigma_e &= -\left[\begin{array}{cc} [x]_{2,2d}^T & 2[x]_{1,d}^T \otimes e^T R \end{array}\right]^T \in \mathbb{R}^{n_{2d}+mn_d}, \\
\Phi_i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} \sigma_e dt & \int_{t_{1,i}}^{t_{2,i}} \sigma_e dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} \sigma_e dt \end{array}\right]^T \in \mathbb{R}^{q_i \times (n_{2d}+mn_d)}, \\
\Xi_i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} r(x,u_i) dt & \int_{t_{1,i}}^{t_{2,i}} r(x,u_i) dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} r(x,u_i) dt \end{array}\right]^T \in \mathbb{R}^{q_i}, \\
\Theta_i &= \left[\begin{array}{cccc} [x]_{2,2r}|_{t_{0,i}}^{t_{1,i}} & [x]_{2,2r}|_{t_{1,i}}^{t_{2,i}} & \cdots & [x]_{2,2r}|_{t_{q_i-1,i}}^{t_{q_i,i}} \end{array}\right]^T \in \mathbb{R}^{q_i \times n_{2r}}.
\end{aligned}$$

Then, (6.55) implies

$$\Phi_i \left[\begin{array}{c} \iota(p,K_i) \\ \mathrm{vec}(\kappa(p)) \end{array}\right] = \Xi_i + \Theta_i p. \tag{6.56}$$

Assumption 6.4.1. *For each $i = 1, 2, \cdots$, there exists an integer q_{i0}, such that when $q_i \geq q_{i0}$ the following rank condition holds.*

$$\mathrm{rank}(\Phi_i) = n_{2d} + mn_d. \tag{6.57}$$

Remark 6.4.1. *Such a rank condition* (6.57) *is in the spirit of persistency of excitation (PE) in adaptive control (e.g. [60, 158]) and is a necessary condition for parameter convergence.*

Given $p \in \mathbb{R}^{n_{2r}}$ and $K_i \in \mathbb{R}^{m \times n_d}$, suppose Assumption 6.4.1 is satisfied and $q_i \geq q_{i0}$ for all $i = 1, 2, \cdots$. Then, it is easy to see that the values of $\iota(p, K_i)$ and $\kappa(p)$ can be uniquely determined from

$$\left[\begin{array}{c} \iota(p,K_i) \\ \mathrm{vec}(\kappa(p)) \end{array}\right] = \left(\Phi_i^T \Phi_i\right)^{-1} \Phi_i^T \left(\Xi_i + \Theta_i p\right) \tag{6.58}$$

Now, we are ready to develop the ADP-based online implementation algorithm

for the proposed policy iteration method.

Algorithm 6.4.1 Global adaptive dynamic programming algorithm

1: *Initialization:* Let p_0 be the constant vector such that $V_0 = p_0^T[x]_{2,2r}$, and let $i = 1$.

2: *Collect online data:* Apply $u = u_i + e$ to the system and compute the data matrices Φ_i, Ξ_i, and Θ_i, until the rank condition (6.57) in Assumption 6.4.1 is satisfied.

3: *Policy evaluation and improvement:* Find an optimal solution (p_i, K_{i+1}, P_i, L_i) to the following optimization problem

$$\min_{p,K,P,L} \quad c^T p \tag{6.59}$$

$$\text{s.t.} \quad \begin{bmatrix} M_L(L) \\ \text{vec}(K) \end{bmatrix} = \left(\Phi_i^T \Phi_i\right)^{-1} \Phi_i^T \left(\Xi_i + \Theta_i p\right) \tag{6.60}$$

$$p_{i-1} - p = M_P(P) \tag{6.61}$$

$$P = P^T \geq 0 \tag{6.62}$$

$$L = L^T \geq 0 \tag{6.63}$$

Then, denote $V_i = p_i^T[x]_{2,2r}$, $u_{i+1} = K_{i+1}[x]_{1,d}$, and go to Step 2) with $i \leftarrow i + 1$.

Lemma 6.4.2. *Under Assumption 6.4.1, (p_i, K_{i+1}, P_i, L_i) is an optimal solution to the optimization problem* (6.59)-(6.63) *if and only if (p_i, P_i, L_i) is an optimal solution to the optimization problem* (6.48)-(6.52) *and $K_{i+1} = \kappa(p_i)$.*

Proof. If (p_i, K_{i+1}, P_i, L_i) is an optimal solution to the optimization problem (6.59)-(6.63), under Assumption 6.4.1, we must have $K_{i+1} = \kappa(p_i)$. Then, it is easy to check (p_i, P_i, L_i) is a feasible solution to the problem (6.48)-(6.52). On the other hand, if (p_i, P_i, L_i) is a feasible solution to the problem (6.48)-(6.52), $(p_i, \kappa(p_i), P_i, L_i)$ must be a feasible solution to the problem (6.59)-(6.63). Finally, since the two optimization problems share the same objective function, their optimal values are the same. □

By Lemma 6.4.2 and Theorem 6.3.1, we immediately have the following Theorem.

Theorem 6.4.1. *Under Assumptions 6.1.1, 6.3.1 and 6.4.1, the following properties hold.*

1. *The optimization problem* (6.59)-(6.63) *has a nonempty feasible set.*

2. The sequences $\{V_i\}_{i=1}^{\infty}$ and $\{u_i\}_{i=1}^{\infty}$ satisfy the properties 2)-5) in Theorem 6.3.1.

Remark 6.4.2. *Notice that the above-mentioned algorithm assumes that both V_0 and u_1 satisfying Assumption 6.3.1 are determined without knowing exactly f and g. In practice, upper and lower bounds of the coefficients in f and g are often available, i.e., there exist polynomial mappings $\bar{f}, \underline{f}, \bar{g}, \underline{g}$, such that $\underline{f} \leq f \leq \bar{f}$ and $\underline{g} \leq g \leq \bar{g}$. Thus, it is possible to find a globally stabilizing control policy for interval systems using robust nonlinear control methods [95, 159]. Then, we can use this control policy as a candidate of u_1 to solve for V_0 from the following robust feasibility problem in SOS programming*

$$-\nabla V_0^T(\tilde{f} + \tilde{g}u_1) - Q - u_1^T R u_1 \quad \in \quad \Sigma_{2,2d}, \tag{6.64}$$

for all $\tilde{f}$ and $\tilde{g}$ such that $\underline{f} \leq \tilde{f} \leq \bar{f}$ and $\underline{g} \leq \tilde{g} \leq \bar{g}$. This problem, if solvable, can be converted into a robust linear matrix inequality and efficiently solved using MATLAB-based solvers, such as the LMI control toolbox [45] or CVX [48].

Remark 6.4.3. *In practice, a stopping criterion can be set. For example, the exploration noise can be terminated and u_i can be applied as the actual control policy, if $|p_i - p_{i+1}| \leq \epsilon$ or $i = i_{max}$, with $\epsilon > 0$ is a pre-defined threshold and i_{max} a pre-defined maximum number of iterations.*

6.5 Extension to nonpolynomial systems

In this section, we extend the proposed global ADP method to deal with an enlarged class of nonlinear systems. First, we will give an illustrative example to show that the SOS condition is conservative for general nonlinear functions. Second, a generalized parametrization method is proposed. Third, a less conservative sufficient condition will be derived to assure the non-negativity of a given nonlinear function. Fourth,

an SDP-based implementation for the proposed policy iteration technique will be presented. Finally, an online learning method will be developed.

6.5.1 An illustrative example

The implementation method via SOS programs developed in the previous section can efficiently handle nonlinear polynomial systems. The results can also be trivially extended to real trigonometric polynomials [17]. However, the SOS-like constraint may be conservative to be used as a sufficient condition for non-negativity of general nonlinear functions. To see this, consider the following illustrative example:

$$f(x) = ax^2 + bx\sin x + c\sin^2 x = \begin{bmatrix} x \\ \sin x \end{bmatrix}^T P \begin{bmatrix} x \\ \sin x \end{bmatrix}. \tag{6.65}$$

Apparently, a symmetric matrix P can be uniquely determined from the constants a, b, and c. Similar to the polynomial case, we know $f(x)$ is positive semidefinite, if P is positive semidefinite. Unfortunately, this condition is very conservative in general. For example, for the cases of $a = 1,\ b = 0,\ c = -0.5$, or $a = 0,\ b = 1,\ c = 0$, it is easy to check $f(x)$ is positive semidefinite. But in both cases, we have either $P = \begin{bmatrix} 1 & 0 \\ 0 & -0.5 \end{bmatrix}$ or $P = \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix}$, which are not positive semidefinite matrices.

This illustrative example shows that, instead of searching for a positive semidefinite matrix P, a less conservative sufficient condition for the non-negativity of more general nonlinear functions is desired. Deriving this condition and developing a global ADP method for more general nonlinear systems are the main objectives of this section.

6.5.2 Generalized parametrization

Assumption 6.5.1. *The function f considered in system* (6.1) *can be decomposed as*

$$f = A\sigma \tag{6.66}$$

where $A \in \mathbb{R}^{n\times l}$ is an uncertain constant matrix, and $\sigma = [\sigma_1(x), \sigma_2(x), \cdots, \sigma_l(x)]^T$ is a vector of locally Lipschitz, piecewise-continuous, and linearly independent functions, satisfying $\sigma_i(0) = 0$, $\forall i = 1, 2, \cdots, l$.

Now, we restrict each feasible solution to Problem 6.2.1 to take the form of $V(x) = \phi^T(x)P\phi(x)$, where $P \in \mathbb{R}^{l\times l}$ is a constant matrix and $\phi = [\phi_1(x), \phi_2(x), \cdots, \phi_N(x)]^T$ is a vector of continuously differentiable, linearly independent, functions vanishing at the origin.

Assumption 6.5.2. *The following are true.*

1. *For each $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, N$, and $k = 1, 2, \cdots, n$, we have*

$$\frac{\partial(\phi_i\phi_j)}{\partial x_k} \in \text{span}\{\sigma_1, \sigma_2, \cdots, \sigma_l\},$$

2. *Let g_i be the i-th column of $g(x)$, with $i = 1, 2, \cdots, m$. Then,*

$$g_i^T\nabla(\phi_i\phi_j) \in \text{span}\{\sigma_1, \sigma_2, \cdots, \sigma_l\}.$$

3. *The weighting function $Q(x)$ defined in* (9.7) *is positive definite and satisfies $Q(x) \in \text{span}\{\sigma_1^2, \sigma_1\sigma_2, \cdots, \sigma_i\sigma_j, \cdots, \sigma_l^2\}$.*

Notice that Assumption 6.5.2 is not restrictive, and can be satisfied by expanding the basis functions. Indeed, if 1) and 2) in Assumption 6.5.2 are not satisfied,

we can always find locally Lipschitz functions $\sigma_{l+1}(x), \sigma_{l+2}(x), \cdots, \sigma_{l+s}(x)$, such that $\sigma_1, \sigma_2, \cdots, \sigma_{l+s}$ are linearly independent and vanish at the origin, satisfying $\frac{\partial(\phi_i\phi_j)}{\partial x_k} \in \text{span}\{\sigma_1, \sigma_2, \cdots, \sigma_{l+s}\}$ and $g_i^T\nabla(\phi_i\phi_j) \in \text{span}\{\sigma_1, \sigma_2, \cdots, \sigma_{l+s}\}$. Then, the decomposition (6.66) can be rewritten as

$$f(x) = \tilde{A}\tilde{\sigma} \tag{6.67}$$

where $\tilde{A} = [A \;\; \mathbf{0}_{n\times s}]$ and $\tilde{\sigma} = [\sigma_1, \sigma_2, \cdots, \sigma_{l+s}]^T$.

Also, if the intersection between $\text{span}\{\sigma_1^2, \sigma_1\sigma_2, \cdots, \sigma_i\sigma_j, \cdots, \sigma_l^2\}$ and the set of all positive definite functions is empty, we can select $Q(x)$ such that $\sqrt{Q(x)}$ is locally Lipschitz and positive definite. Define $\hat{\sigma} = [\sigma_1, \sigma_2, \cdots, \sigma_l, \sqrt{Q(x)}]$. Then, clearly, all the elements in $\hat{\sigma}$ are linearly independent, and the decomposition (6.66) can be rewritten as $f = \hat{A}\hat{\sigma}$, where $\hat{A} = \begin{bmatrix} A & \mathbf{0}_{n\times 1} \end{bmatrix}$.

6.5.3 A sufficient condition for non-negativity

Define $\{\bar{\sigma}_1, \bar{\sigma}_2, \cdots, \bar{\sigma}_{l_1}\}$ as the largest linearly independent subset of $\{\sigma_1^2, \sigma_1\sigma_2, \cdots, \sigma_i\sigma_j, \cdots, \sigma_l^2\}$, and $\{\bar{\phi}_1, \bar{\phi}_2, \cdots, \bar{\phi}_{N_1}\}$ as the largest linearly independent subset of $\{\phi_1^2, \phi_1\phi_2, \cdots, \phi_i\phi_j, \cdots, \phi_N^2\}$.

Then, if $W \in \text{span}\{\phi_1^2, \phi_1\phi_2, \cdots, \phi_N^2\}$ and $\delta \in \text{span}\{\phi_1^2, \sigma_1\sigma_2, \cdots, \sigma_l^2\}$, there exist uniquely constant vectors $p \in \mathbb{R}^{N_1}$ and $h \in \mathbb{R}^{l_1}$, such that $W = p^T\bar{\phi}$ and $\delta = h^T\bar{\sigma}$, where $\bar{\phi} = [\bar{\phi}_1, \bar{\phi}_2, \cdots, \bar{\phi}_{N_1}]^T$ and $\bar{\sigma} = [\bar{\sigma}_1, \bar{\sigma}_2, \cdots, \bar{\sigma}_{l_1}]^T$.

Using the above-mentioned parametrization method, we now show that it is possible to decide if W and δ are positive semidefinite functions, by studying the coefficient vectors p and h.

Without loss of generality, we assume the following properties of $\bar{\phi}_i$:

1) For $i = 1, 2, \cdots, N_2$, we have $\bar{\phi}_i \geq 0$, with N_2 an integer satisfying $1 \leq N_2 \leq N_1$.

2) There exist integers i_r and j_r with $r = 1, 2, \cdots, N_3$, such that $1 \leq i_r, j_r \leq N_2$,

$i_r \neq j_r$ and $\bar{\phi}_{i_r} \geq \bar{\phi}_{j_r}$.

Definition 6.5.1. *For any $p \in \mathbb{R}^{N_1}$, we say $p \in \mathbb{S}_\phi^+$ if and only if there exist constants $\gamma_1, \gamma_2, \cdots, \gamma_{N_2} \geq 0$, $\alpha_1, \alpha_2, \cdots, \alpha_{N_3} \geq 0$, $\beta_1, \beta_2, \cdots, \beta_{N_3}$, and a symmetric positive semidefinite matrix $P \in \mathbb{R}^{N \times N}$, such that $\alpha_i + \beta_i \geq 0$, for $i = 1, 2, \cdots, N_3$, and*

$$p = M_\phi^T \mathrm{vec}(P) + \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_{N_2} \\ \mathbf{0}_{N_1 - N_2} \end{bmatrix} + \sum_{r=1}^{N_3} \left(\begin{bmatrix} \mathbf{0}_{i_r - 1} \\ \alpha_r \\ \mathbf{0}_{N_1 - i_r} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{j_r - 1} \\ \beta_r \\ \mathbf{0}_{N_1 - j_r} \end{bmatrix} \right) \tag{6.68}$$

where $M_\phi \in \mathbb{R}^{N^2 \times N_1}$ is a constant matrix satisfying $M_\phi \bar{\phi} = \phi \otimes \phi$.

In addition, W is said to belong to the set $\mathbb{S}_\phi^+[x]$ if and only if there exists $p \in \mathbb{S}_\phi^+$, such that $W = p^T \bar{\phi}$.

Lemma 6.5.1. *If $p \in \mathbb{S}_\phi^+$, then $p^T \bar{\phi}$ is positive semidefinite.*

Proof. By definition, if $p \in \mathbb{S}_\phi^+$, it follows that

$$\begin{aligned} p^T \bar{\phi} &= \phi^T P \phi + \sum_{i=1}^{N_2} \gamma_2 \bar{\phi}_i + \sum_{r=1}^{N_3} \left(\alpha_r \bar{\phi}_{i_r} + \beta_r \bar{\phi}_{j_r} \right) \\ &\geq \sum_{r=1}^{N_3} \left(\alpha_r \bar{\phi}_{i_r} - |\beta_r| \bar{\phi}_{j_r} \right) = \sum_{r=1}^{N_3} \left(\alpha_r - |\beta_r| \right) \bar{\phi}_{i_r} \\ &\geq 0. \end{aligned}$$

The proof is complete. □

In the same way, we can find two sets $\mathbb{S}_\sigma^+$ and $\mathbb{S}_\sigma^+[x]$, such that the following implications hold

$$h \in \mathbb{S}_\sigma^+ \Leftrightarrow h^T \bar{\phi} \in \mathbb{S}_\sigma^+[x] \Rightarrow h^T \bar{\phi} \geq 0. \tag{6.69}$$

6.5.4 Generalized policy iteration

Assumption 6.5.3. *There exist $p_0 \in \mathbb{R}^{N_1}$ and $K_1 \in \mathbb{R}^{m\times l_1}$, such that $V_0 = p_0^T\bar{\phi}$, $u_1 = K_1\sigma$, and $\mathcal{L}(V_0, u_1) \in \mathbb{S}_\phi^+$.*

Remark 6.5.1. *Under Assumptions 6.5.1, 6.5.2, and 6.5.3, Assumption 6.1.1 is satisfied.*

Now, let us show how the proposed policy iteration can be practically implemented. First of all, given $p \in \mathbb{R}^{N_1}$, since $u_i = K_i\sigma$, we can always find two linear mappings $\bar{\iota} : \mathbb{R}^{N_1} \times \mathbb{R}^{ml} \to \mathbb{R}^{l_1}$ and $\bar{\kappa} : \mathbb{R}^{N_1} \to \mathbb{R}^{l_1\times ml}$, such that

$$\bar{\iota}(p, K)^T\bar{\sigma} = \mathcal{L}(p^T\bar{\phi}, K_i\sigma) \tag{6.70}$$

$$\bar{\kappa}(p)^T\bar{\sigma} = -\frac{1}{2}R^{-1}g^T\nabla(p^T\bar{\phi}) \tag{6.71}$$

Then, under Assumptions 6.1.2, 6.5.1, 6.5.2, and 6.5.3, the proposed policy iteration can be implemented as follows.

Algorithm 6.5.1 SDP-based policy iteration for nonpolynomial systems

1: *Initialization:*
2: Find $p_0 \in \mathbb{R}^{N_1}$ and $K_1 \in \mathbb{R}^{m\times l_1}$ satisfying Assumption 6.5.3, and let $i = 1$.
3: *Policy evaluation and improvement:*
4: Solve for an optimal solution (p_i, K_{i+1}) of the following problem.

$$\min_{p,K} \quad c^T p \tag{6.72}$$

$$\text{s.t.} \quad \bar{\iota}(p, K_i) \in \mathbb{S}_\sigma^+ \tag{6.73}$$

$$p_{i-1} - p \in \mathbb{S}_\phi^+ \tag{6.74}$$

$$K = \bar{\kappa}(p) \tag{6.75}$$

where $c = \int_{\mathbb{R}^n} s(x)\bar{\phi}(x)dx$. Then, denote $V_i = p_i^T\bar{\phi}$ and $u_{i+1} = K_{i+1}\sigma$.
5: Go to Step 2) with i replaced by $i + 1$.

Some useful facts about the above-mentioned policy iteration algorithm are summarized in the following theorem, of which the proof is omitted, because it is nearly identical to the proof of Theorem 6.3.1.

Theorem 6.5.1. *Under Assumptions 6.1.2, 6.5.1, 6.5.2, and 6.5.3 the following are true, for* $i = 1, 2, \cdots$.

1) *The optimization problem* (6.72)-(6.75) *has a nonempty feasible set.*

2) *The closed-loop system comprised of* (6.1) *and* $u = u_i(x)$ *is globally asymptotically stable at the origin.*

3) $V_i \in \mathcal{P}$. *In addition,* $V^{\mathrm{o}}(x_0) \leq V_i(x_0) \leq V_{i-1}(x_0), \quad \forall x_0 \in \mathbb{R}^n$.

4) *There exists* $p^* \in \mathbb{R}^{N_1}$, *such that* $\lim_{i\to\infty} V_i(x_0) = p^{*T}\bar{\phi}(x_0)$, $\forall x_0 \in \mathbb{R}^n$.

5) *Along the solutions of the system* (6.1) *with* $u^* = -\frac{1}{2}R^{-1}g^T\nabla(p^{*T}\bar{\phi})$, *it follows that*

$$0 \leq p^{*T}\bar{\phi}(x_0) - V^{\mathrm{o}}(x_0) \leq -\int_0^\infty \mathcal{H}(p^{*T}\bar{\phi}(x(t)))dt. \tag{6.76}$$

6.5.5 Online implementation via global adaptive dynamic programming

Let $V = p^T\bar{\phi}$. Similar as in Section 6.4, over the interval $[t, t+\delta t]$, we have

$$\begin{aligned} & p^T\left[\bar{\phi}(x(t)) - \bar{\phi}(x(t+\delta t))\right] \\ = & \int_t^{t+\delta t} \left[r(x,u_i) + \bar{\iota}(p,K_i)^T\bar{\sigma} + 2\sigma^T\bar{\kappa}(p)^T Re\right] dt \end{aligned} \tag{6.77}$$

Therefore, (6.77) shows that, given $p \in \mathbb{R}^{N_1}$, $\bar{\iota}(p, K_i)$ and $\bar{\kappa}(p)$ can be directly obtained by using real-time online data, without knowing the precise knowledge of f and g.

Indeed, define

$$\begin{aligned}
\bar{\sigma}_e &= -\left[\begin{array}{cc} \bar{\sigma}^T & 2\sigma^T \otimes e^T R \end{array}\right]^T \in \mathbb{R}^{l_1+ml}, \\
\bar{\Phi}_i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} \bar{\sigma}_e dt & \int_{t_{1,i}}^{t_{2,i}} \bar{\sigma}_e dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} \bar{\sigma}_e dt \end{array}\right]^T \in \mathbb{R}^{q_i \times (l_1+ml)}, \\
\bar{\Xi}_i &= \left[\begin{array}{cccc} \int_{t_{0,i}}^{t_{1,i}} r(x,u_i) dt & \int_{t_{1,i}}^{t_{2,i}} r(x,u_i) dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} r(x,u_i) dt \end{array}\right]^T \in \mathbb{R}^{q_i}, \\
\bar{\Theta}_i &= \left[\begin{array}{cccc} \bar{\phi}(x)|_{t_{0,i}}^{t_{1,i}} & \bar{\phi}(x)|_{t_{1,i}}^{t_{2,i}} & \cdots & \bar{\phi}(x)|_{t_{q_i-1,i}}^{t_{q_i,i}} \end{array}\right]^T \in \mathbb{R}^{q_i \times N_1}.
\end{aligned}$$

Then, (6.77) implies

$$\bar{\Phi}_i \left[\begin{array}{c} \bar{\iota}(p, K_i) \\ \mathrm{vec}(\bar{\kappa}(p)) \end{array}\right] = \bar{\Xi}_i + \bar{\Theta}_i p. \tag{6.78}$$

Assumption 6.5.4. *For each $i = 1, 2, \cdots$, there exists an integer q_{i0}, such that, when $q_i \geq q_{i0}$, the following rank condition holds.*

$$\mathrm{rank}(\bar{\Phi}_i) = l_1 + ml. \tag{6.79}$$

Let $p \in \mathbb{R}^{N_1}$ and $K_i \in \mathbb{R}^{m \times l}$. Suppose Assumption 6.5.4 holds and assume $q_i \geq q_{i0}$, for $i = 1, 2, \cdots$. Then, $\bar{\iota}(p, K_i)$ and $\bar{\kappa}(p)$ can be uniquely determined by

$$\left[\begin{array}{c} h \\ \mathrm{vec}(K) \end{array}\right] = \left(\bar{\Phi}_i^T \bar{\Phi}_i\right)^{-1} \bar{\Phi}_i^T \left(\bar{\Xi}_i + \bar{\Theta}_i p\right). \tag{6.80}$$

Now, we are ready to develop the ADP-based online implementation algorithm for the proposed policy iteration method.

Properties of the above algorithm are summarized in the following corollary.

Corollary 6.5.1. *Under Assumptions 6.1.2, 6.5.1, 6.5.2, 6.5.3, and 6.5.4, the algorithm enjoys the following properties.*

Algorithm 6.5.2 Global adaptive dynamic programming algorithm for nonpolynomial systems

1: *Initialization:* Let p_0 and K_1 satisfying Assumption 6.5.3, and let $i = 1$.
2: *Collect online data:* Apply $u = u_i + e$ to the system and compute the data matrices $\bar{\Phi}_i$, $\bar{\Xi}_i$, and $\bar{\Theta}_i$, until the rank condition (6.79) is satisfied.
3: *Policy evaluation and improvement:* Find an optimal solution (p_i, h_i, K_{i+1}) to the following optimization problem

$$\min_{p,h,K} \; c^T p \tag{6.81}$$

$$\text{s.t.} \quad \begin{bmatrix} h \\ \text{vec}(K) \end{bmatrix} = \left(\bar{\Phi}_i^T \bar{\Phi}_i\right)^{-1} \bar{\Phi}_i^T \left(\bar{\Xi}_i + \bar{\Theta}_i p\right) \tag{6.82}$$

$$h \in \mathbb{S}_\sigma^+ \tag{6.83}$$

$$p_{i-1} - p \in \mathbb{S}_\phi^+ \tag{6.84}$$

Then, denote $V_i = p_i \bar{\phi}$ and $u_{i+1} = K_{i+1} \bar{\sigma}$.
4: Go to Step 2) with $i \leftarrow i + 1$.

1. *The optimization problem* (6.100)-(6.103) *has a feasible solution.*

2. *The sequences* $\{V_i\}_{i=1}^\infty$ *and* $\{u_i\}_{i=1}^\infty$ *satisfy the properties 2)-5) in Theorem 6.5.1.*

6.6 Robust redesign

Consider consider nonlinear system with dynamic uncertainties as follows

$$\dot{w} = q(w, x) \tag{6.85}$$

$$\dot{x} = f(x) + g(x)\left[u + \Delta(w, x)\right] \tag{6.86}$$

where $x \in \mathbb{R}^n$ is the system state, $w \in \mathbb{R}^{n_w}$ is the state of the dynamic uncertainty, $u \in \mathbb{R}^m$ is the control input, $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are unknown polynomial mappings with $f(0) = 0$.

Again, in the presence of the dynamic uncertainty, i.e., the w-subsystem, Algorithm 6.5.2 may not lead to an optimal or suboptimal control policy, since u_i obtained in Algorithm 6.5.2 may not be stabilizing for the overall system (6.85)-(6.86). There-

fore, to balance the tradeoff between global robust stability and optimality, here we develop a method to redesign the control policy. Similarly as in the previous chapter, the idea is inspired from the work by [83, 129].

To begin with, we define the cost functional as

$$\min \qquad J(x_0, u) = \int_0^\infty \left[Q(x) + u^T R u\right] dt, \tag{6.87}$$

where $Q(x) = Q_0(x) + \epsilon|x|^2$, with $Q_0(x)$ is a positive definite function, $\epsilon > 0$ is a constant, R is a symmetric and positive definite matrix.

Our design objective is twofold. First, we intend to minimize the cost (6.87) for the nominal system

$$\dot{x} = f(x) + g(x)u, \tag{6.88}$$

by finding online an optimal control policy u_o. Second, we want to guarantee the stability of the system comprised of (6.85) and (6.86) by redesigning the optimal control policy.

To this end, let us introduce the following Assumption.

Assumption 6.6.1. *Consider the system comprised of* (6.85) *and* (6.86)*. There exist functions* $\underline{\lambda}, \bar{\lambda} \in \mathcal{K}_\infty$*,* $\kappa_1, \kappa_2, \kappa_3 \in \mathcal{K}$*, and positive definite functions* W *and* κ_4*, such that for all* $w \in \mathbb{R}^p$ *and* $x \in \mathbb{R}^n$*, we have*

$$\underline{\lambda}(|w|) \leq W(w) \leq \bar{\lambda}(|w|), \tag{6.89}$$

$$|\Delta(w, x)| \leq \kappa_1(|w|) + \kappa_2(|x|), \tag{6.90}$$

together with the following implication:

$$W(w) \geq \kappa_3(|x|) \Rightarrow \nabla W(w)^T q(w, x) \leq -\kappa_4(w). \tag{6.91}$$

Assumption 6.6.1 implies that the w-system (6.85) is input-to-state stable (ISS) [149, 151] when x is considered as the input.

Let $V_i \in \mathcal{P}$ and u_i be the cost function and the control policy obtained from Algorithm 6.5.2. Then, we know that $\mathcal{L}(V_i, u_i) \geq 0$. Also, there exist $\underline{\alpha}, \bar{\alpha} \in K_\infty$, such that the following inequalities hold:

$$\underline{\alpha}(|x|) \leq V^{\mathrm{o}}(x) \leq V_i(x) \leq V_0(x) \leq \bar{\alpha}(|x|), \quad \forall x_0 \in \mathbb{R}^n; \tag{6.92}$$

The robustly redesigned control policy is given below:

$$u_{r,i} = \rho^2(|x|^2)u_i + e \tag{6.93}$$

where $\rho(\cdot)$ is a smooth and nondecreasing function with $\rho(s) \geq 1$, $\forall s > 0$, e denotes the time varying exploration noise added for the purpose of online learning.

Theorem 6.6.1. *Consider the closed-loop system comprised of* (6.85), (6.86), *and* (6.93). *Let $V_i \in \mathcal{P}$ and u_i be the cost function and the control policy obtained from Algorithm 6.5.2 at the i-th iteration step. Then, the closed-loop system is ISS with respect to e as the input, if the following gain condition holds:*

$$\gamma > \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \circ \bar{\alpha} + \kappa_2, \tag{6.94}$$

where $\gamma \in \mathcal{K}_\infty$ is defined by

$$\gamma(s) = \epsilon s \sqrt{\frac{\frac{1}{4} + \frac{1}{2}\rho^2(s^2)}{\lambda_{\min}(R)}}. \tag{6.95}$$

Proof. Let $\chi_1 = \kappa_3 \circ \underline{\alpha}^{-1}$. Then, under Assumption 6.6.1, we immediately have the

following implications

$$\begin{aligned} & W(w) \geq \chi_1(V_i(x)) \\ \Rightarrow \quad & W(w) \geq \kappa_3\left(\underline{\alpha}^{-1}(V_i(x))\right) \geq \kappa_3\left(|x|\right) \\ \Rightarrow \quad & \nabla W(w)^T q(w,x) \leq -\kappa_4(w) \end{aligned} \tag{6.96}$$

Define $\tilde{\rho}(x) = \sqrt{\frac{\frac{1}{4}+\frac{1}{2}\rho^2(|x|^2)}{\lambda_{\min}(R)}}$. Then, along solutions of the system comprised of (6.86), it follows that

$$\begin{aligned} & \nabla V_i^T\left[f + g\left(u_{r,i} + \Delta\right)\right] \\ \leq \quad & -Q(x) - |u_i|_R^2 + \nabla V_i^T g\left[\left(\rho^2(|x|^2) - 1\right)u_i + \Delta + e\right] \\ \leq \quad & -Q(x) - \tilde{\rho}^2\left|g^T \nabla V_i\right|^2 + \nabla V_i^T g\left(\Delta + e\right) \\ \leq \quad & -Q(x) - \left|\tilde{\rho} g^T \nabla V_i - \frac{1}{2}\tilde{\rho}^{-1}\Delta\right|^2 + \frac{1}{4}\tilde{\rho}^{-2}|\Delta + e|^2 \\ \leq \quad & -Q_0(x) - \epsilon^2|x|^2 + \tilde{\rho}^{-2}\max\{|\Delta|^2, |e|^2\} \\ \leq \quad & -Q_0(x) - \tilde{\rho}^{-2}\left(\gamma^2 - \max\{|\Delta|^2, |e|^2\}\right) \end{aligned}$$

Hence, by defining $\chi_2 = \bar{\alpha} \circ (\gamma - \kappa_2)^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1}$, it follows that

$$\begin{aligned} & V_i(x) \geq \max\{\chi_2(W(w)), \bar{\alpha} \circ (\gamma - \kappa_2)^{-1}\left(|e|\right)\} \\ \Leftrightarrow \quad & V_i(x) \geq \bar{\alpha} \circ (\gamma - \kappa_2)^{-1} \circ \max\{\kappa_1 \circ \underline{\lambda}^{-1}(W(w)), |e|\} \\ \Rightarrow \quad & (\gamma - \kappa_2) \circ \bar{\alpha}^{-1}\left(V_i(x)\right) \geq \max\{\kappa_1 \circ \underline{\lambda}^{-1}(W(w)), |e|\} \\ \Rightarrow \quad & \gamma(|x|) - \kappa_2(|x|) \geq \max\{\kappa_1 \circ \underline{\lambda}^{-1}(W(w)), |e|\} \\ \Rightarrow \quad & \gamma(|x|) - \kappa_2(|x|) \geq \max\{\kappa_1(|w|), |e|\} \\ \Rightarrow \quad & \gamma(|x|) \geq \max\{|\Delta(w,x)|, |e|\} \\ \Rightarrow \quad & \nabla V_i^T\left[f + g\left(u_{r,i+1} + \Delta\right)\right] \leq -Q_0(x) \end{aligned} \tag{6.97}$$

Finally, by the gain condition, we have

$$\begin{aligned}
&\gamma > \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \circ \bar{\alpha} + \kappa_2 \\
\Rightarrow \quad & Id > (\gamma - \kappa_2)^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \circ \bar{\alpha} \\
\Rightarrow \quad & Id > \bar{\alpha} \circ (\gamma - \kappa_2)^{-1} \circ \kappa_1 \circ \underline{\lambda}^{-1} \circ \kappa_3 \circ \underline{\alpha}^{-1} \\
\Rightarrow \quad & Id > \chi_2 \circ \chi_1. \qquad (6.98)
\end{aligned}$$

The proof is thus completed by the small-gain theorem [81]. □

Similarly as in the previous section, along the solution of the system (6.86) and (6.93), it follows that

$$\begin{aligned}
\dot{V} &= \nabla V^T (f + g u_{r,i}) \\
&= \nabla V^T (f + g u_i) + \nabla V^T g \tilde{e} \\
&= -r(x, u_i) - \mathcal{L}(V, u_i) + \nabla V^T g \tilde{e} \\
&= -r(x, u_i) - \mathcal{L}(V, u_i) + 2(\frac{1}{2} R^{-1} g^T \nabla V)^T R \tilde{e} \\
&= -r(x, u_i) - \iota(p, K_i)^T [x]_{2,2d} - 2[x]_{1,d}^T \kappa(p)^T R \tilde{e} \qquad (6.99)
\end{aligned}$$

where $\tilde{e} = (\rho^2(|x|^2) - 1)u_i + e$.

Therefore, we can redefine the data matrices as follows. Indeed, define

$$\begin{aligned}
\bar{\sigma}_e &= -\begin{bmatrix} \bar{\sigma}^T & 2\sigma^T \otimes e^T R \end{bmatrix}^T \in \mathbb{R}^{l_1 + ml}, \\
\bar{\Phi}_i &= \begin{bmatrix} \int_{t_{0,i}}^{t_{1,i}} \bar{\sigma}_e dt & \int_{t_{1,i}}^{t_{2,i}} \bar{\sigma}_e dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} \bar{\sigma}_e dt \end{bmatrix}^T \in \mathbb{R}^{q_i \times (l_1 + ml)}, \\
\bar{\Xi}_i &= \begin{bmatrix} \int_{t_{0,i}}^{t_{1,i}} r(x, u_i) dt & \int_{t_{1,i}}^{t_{2,i}} r(x, u_i) dt & \cdots & \int_{t_{q_i-1,i}}^{t_{q_i,i}} r(x, u_i) dt \end{bmatrix}^T \in \mathbb{R}^{q_i}, \\
\bar{\Theta}_i &= \begin{bmatrix} \bar{\phi}(x)|_{t_{0,i}}^{t_{1,i}} & \bar{\phi}(x)|_{t_{1,i}}^{t_{2,i}} & \cdots & \bar{\phi}(x)|_{t_{q_i-1,i}}^{t_{q_i,i}} \end{bmatrix}^T \in \mathbb{R}^{q_i \times N_1}.
\end{aligned}$$

Then, the global robust adaptive dynamic programming algorithm is given below.

Algorithm 6.6.1 The global robust adaptive dynamic programming algorithm algorithm

1: *Initialization:* Let p_0 and K_1 satisfying Assumption 6.5.3, and let $i = 1$.
2: *Collect online data:* Apply $u = u_{r,i} = \rho^2(|x|^2)u_i + e$ to the system and compute the data matrices Φ_i, Ξ_i, and Θ_i, until the rank condition in Assumption 6.5.4 is satisfied.
3: *Policy evaluation and improvement:* Find an optimal solution (p_i, h_i, K_{i+1}) to the following optimization problem

$$\min_{p,h,K} \ c^T p \tag{6.100}$$

$$\text{s.t.} \quad \begin{bmatrix} h \\ \text{vec}(K) \end{bmatrix} = \left(\bar{\Phi}_i^T \bar{\Phi}_i\right)^{-1} \bar{\Phi}_i^T \left(\bar{\Xi}_i + \bar{\Theta}_i p\right) \tag{6.101}$$

$$h \in \mathbb{S}_\sigma^+ \tag{6.102}$$

$$p_{i-1} - p \in \mathbb{S}_\phi^+ \tag{6.103}$$

Then, denote $V_i = p_i\bar{\phi}$ and $u_{i+1} = K_{i+1}\bar{\sigma}$.
4: Go to Step 2) with $i \leftarrow i + 1$.

Corollary 6.6.1. *Under Assumptions 6.1.2, 6.3.1 and 6.4.1, the following properties hold.*

1) The optimization problem (6.59)-(6.63) *has a nonempty feasible set.*

2) The sequences $\{V_i\}_{i=1}^\infty$ *and* $\{u_i\}_{i=1}^\infty$ *satisfy the properties 2)-5) in Theorem 6.3.1.*

3) Suppose the gain condition (6.94) *holds. Then, the closed-loop system comprised of* (6.85), (6.86), *and* (6.93) *is ISS with respect to* e *as the input.*

6.7 Numerical examples

This section provides three numerical examples to illustrate the effectiveness of the proposed algorithms.

6.7.1 A scalar nonlinear polynomial system

Consider the following polynomial system

$$\dot{x} = ax^2 + bu \tag{6.104}$$

where $x \in \mathbb{R}$ is the system state, $u \in \mathbb{R}$ is the control input, a and b, satisfying $a \in [0, 0.05]$ and $b \in [0.5, 1]$, are uncertain constants. The cost to be minimized is defined as

$$J(x_0, u) = \int_0^\infty (0.01x^2 + 0.01x^4 + u^2)dt. \tag{6.105}$$

An initial stabilizing control policy can be selected as $u_1 = -0.1x - 0.1x^3$, which globally asymptotically stabilizes system (6.104), for any a and b satisfying the given range. Further, it is easy to see that $V_0 = 10(x^2 + x^4)$ and u_1 satisfy Assumption 6.3.1 with $r = 2$. In addition, in the present case, we set $d = 3$ and $d_1 = 0$ in Assumption 6.3.1.

Only for the purpose of simulation, set $a = 0.01$, $b = 1$, and $x(0) = 2$. The proposed global ADP method is applied with the control policy updated after every five seconds, and convergence is attained after five iterations, when $|p_i - p_{i-1}| \leq 10^{-3}$. The coefficient in the objective function (6.48) is defined as $c = [x(1)]_{2,4} + [x(-1)]_{2,4}$, i.e., the weighting function is set to be $s(x) = \delta(x-1) + \delta(x+1)$ with $\delta(\cdot)$ denoting the impulse function. The exploration noise is set to be $e = 0.01(\sin(10t) + \sin(3t) + \sin(100t))$, which is turned off after the fifth iteration.

The suboptimal control policy and the cost function obtained after five iterations are

$$V^* = 0.1020x^2 + 0.007x^3 + 0.0210x^4, \tag{6.106}$$

$$u^* = -0.2039x - 0.02x^2 - 0.0829x^3. \tag{6.107}$$

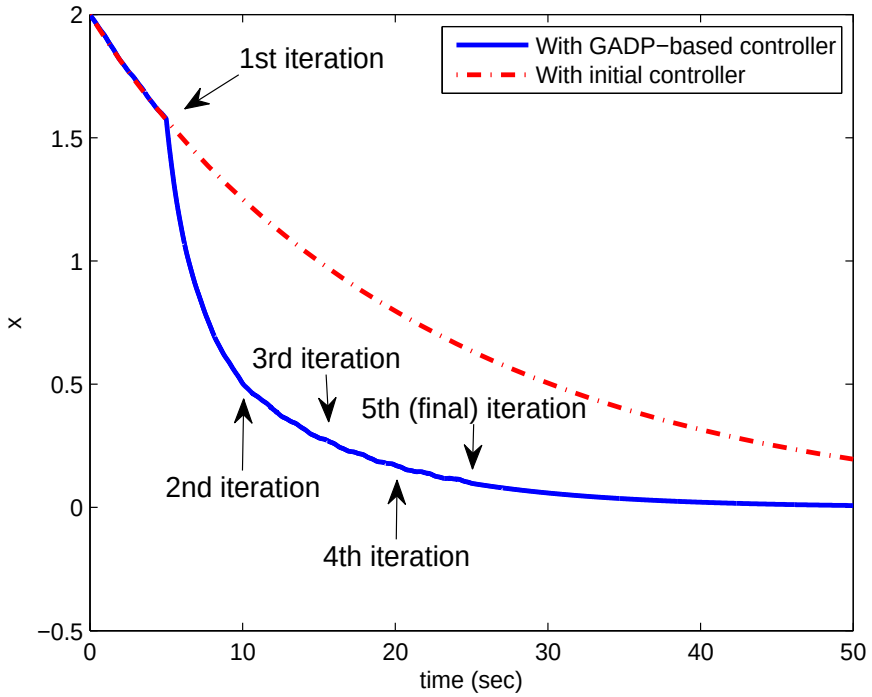

Figure 6.1: Simulation of the scalar system: State trajectory

For comparison purpose, the exact optimal cost and the control policy are given below.

$$V^o = \frac{x^3}{150} + \frac{(\sqrt{101x^2+100})^3}{15150} - \frac{20}{303} \tag{6.108}$$

$$u^o = -\frac{x^2\sqrt{101x^2+100} + 101x^4 + 100x^2}{100\sqrt{101x^2+100}} \tag{6.109}$$

Figures 6.1-6.4 shows the comparison of the suboptimal control policy with respect to the exact optimal control policy and the initial control policy.

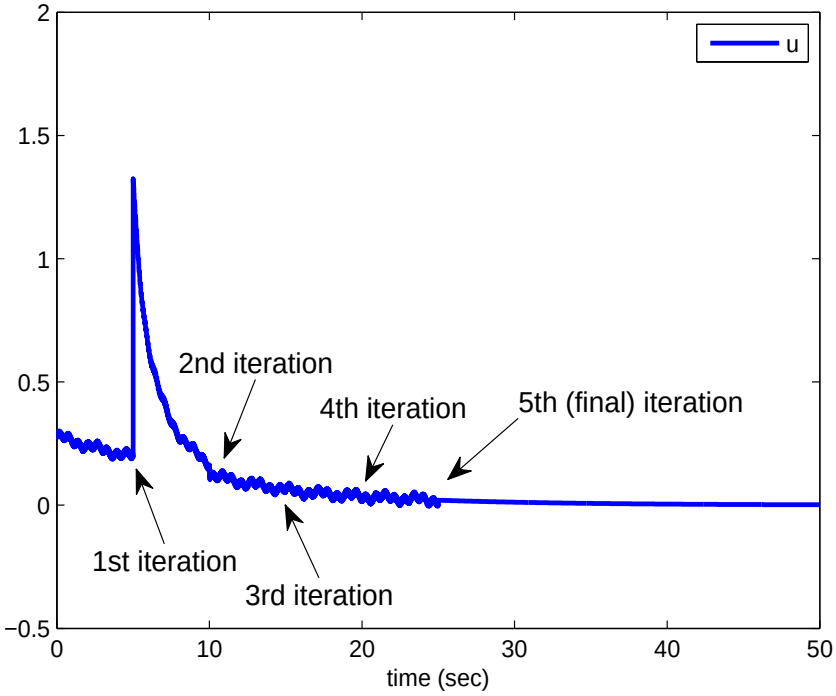

Figure 6.2: Simulation of the scalar system: Control input

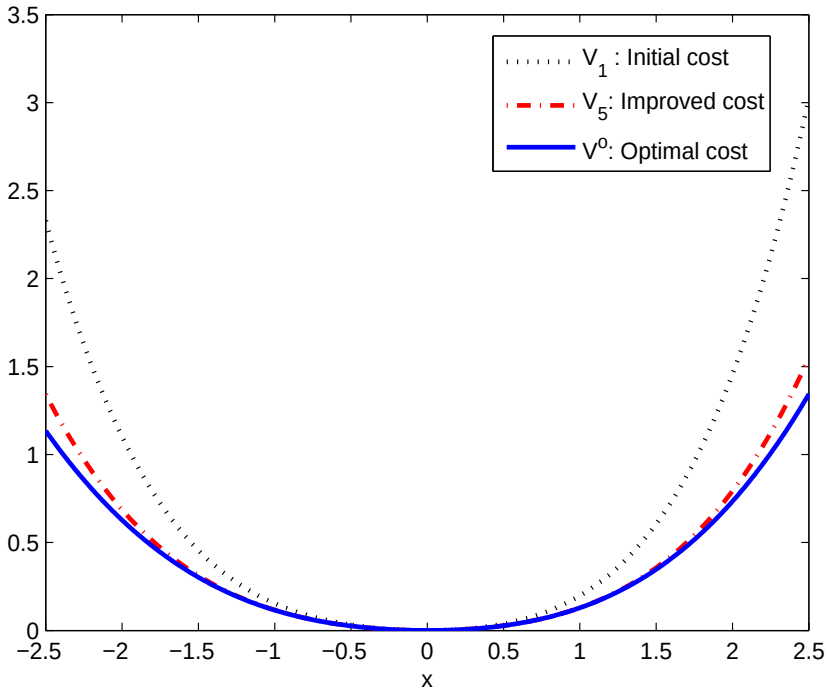

Figure 6.3: Simulation of the scalar system: Cost functions

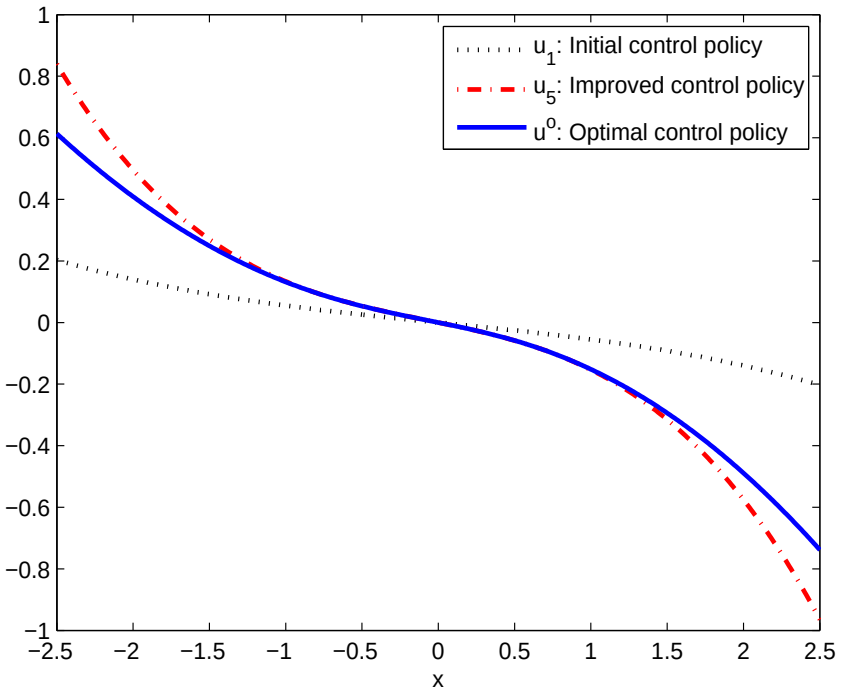

Figure 6.4: Simulation of the scalar system: Control policies

6.7.2 Inverted pendulum

Consider the following differential equations which are used to model an inverted pendulum:

$$\dot{x}_1 = x_2 \tag{6.110}$$

$$\dot{x}_2 = -\frac{kl}{m}x_2 + g\sin(x_1) + \frac{1}{m}u \tag{6.111}$$

where x_1 is the angular position of the pendulum, x_2 is the angular velocity, u is the control input, g is the gravity constant, l is the length of the pendulum, k is the coefficient of friction, and m is the mass. The design objective is to find a suboptimal and globally stabilizing control policy that can drive the state to the origin. Assume the parameters are not precisely known, but they satisfy $0.5 \leq k \leq 1.5$, $0.5 \leq m \leq 1.5$, $0.8 \leq l \leq 1.2$, and $9 \leq g \leq 10$.

Notice that we can select $\phi = [x_1,\ x_2]^T$ and $\sigma = [x_1,\ x_2,\ \sin x_1]^T$. The cost is selected as $J(x_0, u) = \int_0^\infty (10x_1^2 + 10x^2 + u^2)dt$.

Further, set $\bar{\phi} = [x_1^2,\ x_1x_2,\ x_2^2]^T$ and $\bar{\sigma} = [x_1^2,\ x_2^2,\ x_1 \sin x_1,\ \sin^2 x_1,\ x_2 \sin x_1,\ x_1x_2]^T$. Then, based on the range of the system parameters, a pair (V_0, u_1) satisfying Assumption 6.5.3 can be obtained as $u_1 = -10x_1^2 - x_2 - 15 \sin x_1$, and $V_0 = 320.1297x_1^2 + 46.3648x_1x_2 + 22.6132x_2^2$. The coefficient vector c is defined as $c = \bar{\phi}(1, -1) + \bar{\phi}(1, 1)$.

The initial condition for the system is set to be $x_1(0) = -1.5$ and $x_2(0) = 1$. The control policy is updated after 0.5 seconds, until convergence is attained after 4 iterations. The exploration noise we use is the sum of sinusoidal waves with different frequencies, and it is terminated once the convergence is attained.

The resultant control policy and the cost function are $u^* = -20.9844x_1 - 7.5807x_2$ and $V^* = 86.0463x_1^2 + 41.9688x_1x_2 + 7.5807x_2^2$. Simulation results are provided in Figures 6.5-6.6. It can be seen that the system performance is significantly improved under the proposed ADP scheme.

6.7.3 Jet engine surge and stall dynamics

Consider the following system, which is inspired by the jet engine surge and stall dynamics in [94, 119]

$$\dot{r} = -\sigma r^2 - \sigma r \left(2\phi + \phi^2\right) \tag{6.112}$$

$$\dot{\phi} = -a\phi^2 - b\phi^3 - (u + 3r\phi + 3r) \tag{6.113}$$

where $r > 0$ is the normalized rotating stall amplitude, ϕ is the deviation of the scaled annulus-averaged flow, u is the deviation of the plenum pressure rise and is treated as the control input, $\sigma \in [0.2, 0.5]$, $a \in [1.2, 1.6]$, $b \in [0.3, 0.7]$ are uncertain constants.

In this example, we assume the variable r is not available for real-time feedback

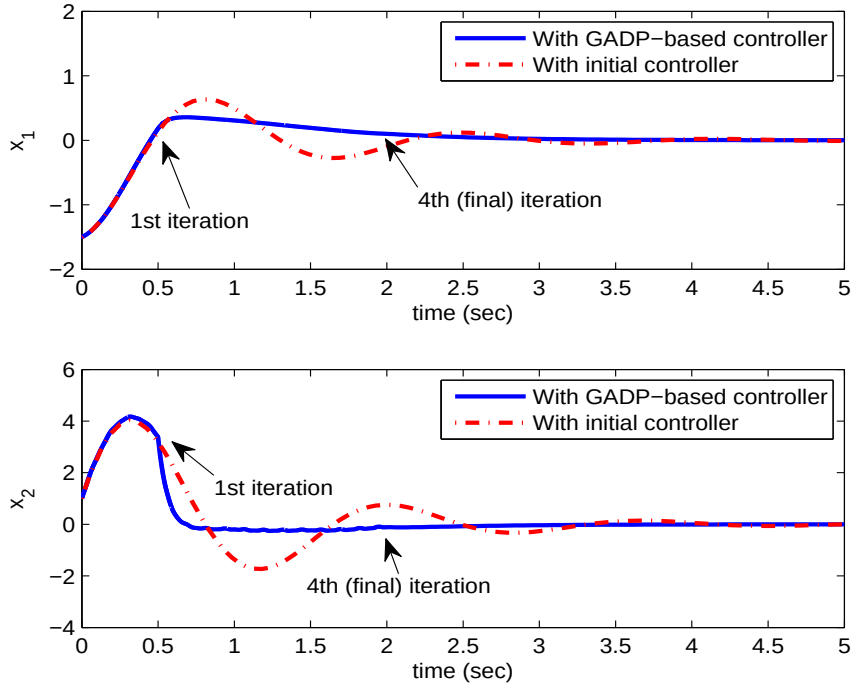

Figure 6.5: Simulation of the inverted pendulum: State trajectories

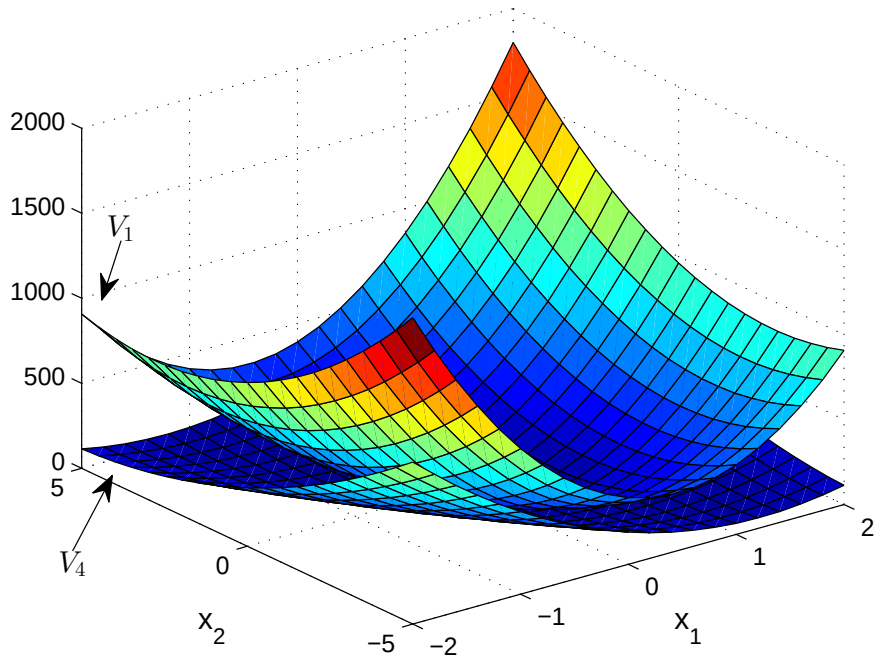

Figure 6.6: Simulation of the inverted pendulum: Cost functions

control due to a $0.2s$ time-delay in measuring it. Hence, the objective is to find a control policy that only relies on ϕ.

The cost function we used here is

$$J = \int_0^\infty \left(5\phi^2 + u^2\right) dt \tag{6.114}$$

and an initial control policy is chosen as

$$u_{r,1} = -\frac{1}{2}\rho^2(\phi^2)\left(2x - 1.4x^2 - 0.45x^3\right) \tag{6.115}$$

with $\rho(s) = \sqrt{2}$.

Only for the purpose of simulation, we set $\sigma = 0.3$, $a = 1.5$, and $b = 0.5$. The control policy is updated every $0.25s$ until the convergence criterion, $|p_i - p_{i-1}| < 0.1$ is satisfied. The simulation results are provided in Figures 6.7-6.9. It can be seen that the system performance has been improved via online learning.

6.8 Conclusions

This chapter has proposed a global robust adaptive dynamic programming method. In particular, a new policy iteration scheme has been developed. Different from conventional policy iteration, the new iterative technique does not attempt to solve a partial differential equation but a convex optimization problem at each iteration step. It has been shown that, this method can find a suboptimal solution to continuous-time nonlinear optimal control problems [103]. In addition, the resultant control policy is globally stabilizing. Also, the method can be viewed as a computational strategy to solve directly Hamilton-Jacobi inequalities, which are used in H_∞ control problems [52, 169]. In the presence of dynamic uncertainties, robustification of the proposed algorithms and their online implementations has been addressed, by integration with

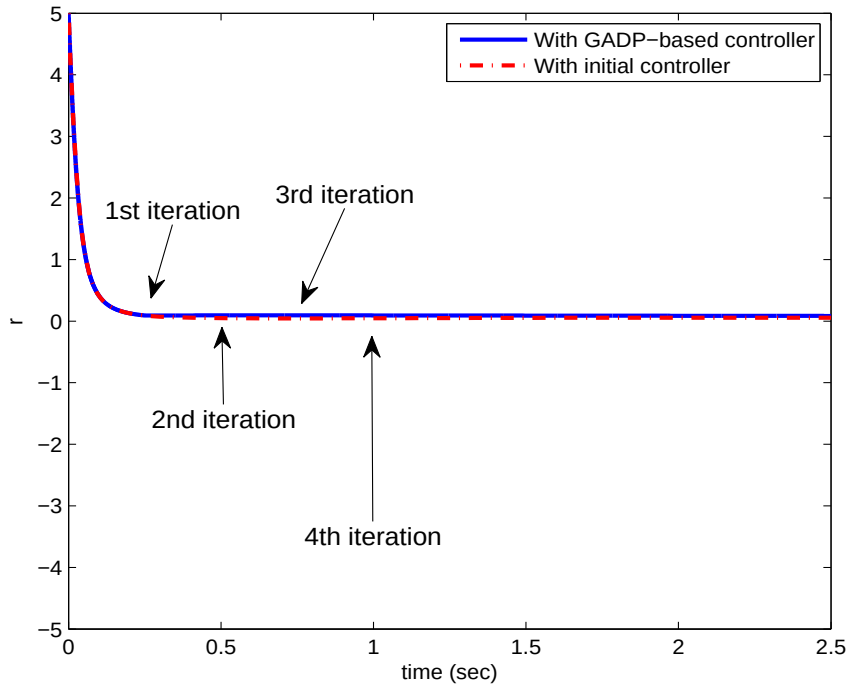

Figure 6.7: Simulation of the jet engine: Trajectories of r.

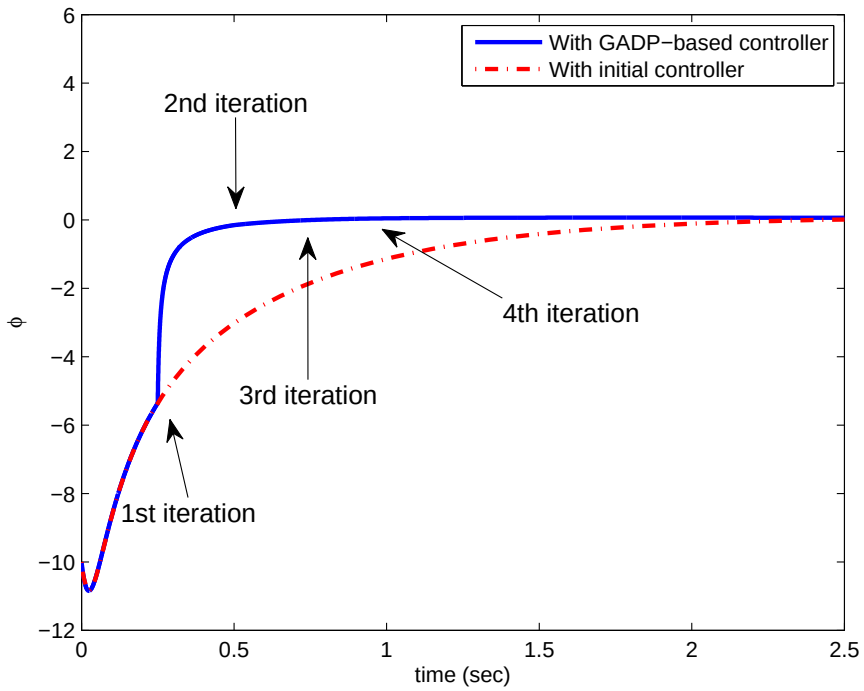

Figure 6.8: Simulation of the jet engine: Trajectories of ϕ.

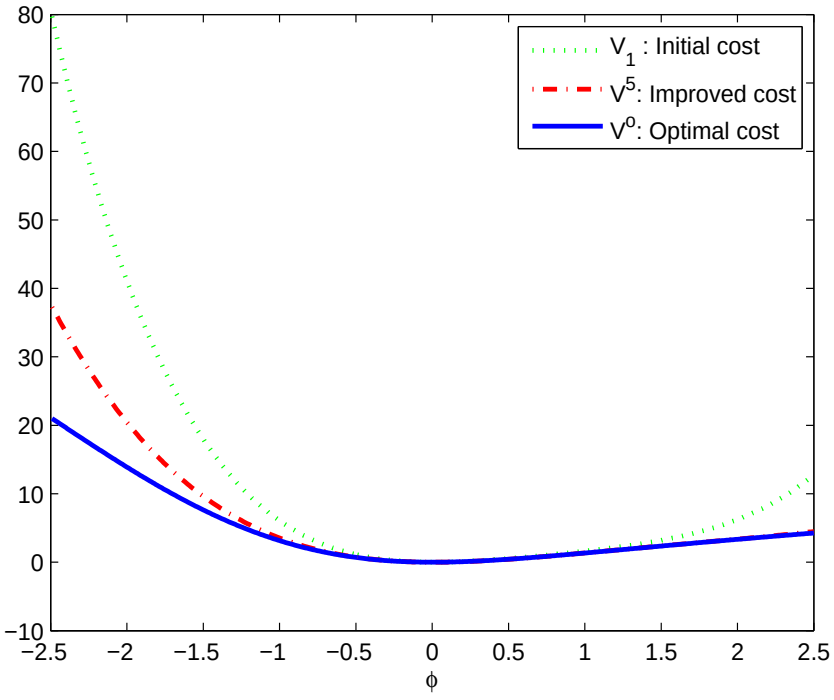

Figure 6.9: Simulation of the jet engine: Value functions.

the ISS property [149, 151] and the nonlinear small-gain theorem [83, 81].

When the system parameters are unknown, conventional ADP methods utilize neural networks to approximate online the optimal solution, and a large number of basis functions are required to assure high approximation accuracy on some compact sets. Thus, neural-network-based ADP schemes may result in slow convergence and loss of global asymptotic stability for the closed-loop system. Here, the proposed GRADP method has overcome the two above-mentioned shortcomings, and it yields computational benefits.

Chapter 7

RADP as a theory of sensorimotor control

Many tasks that humans perform in our daily lives involve different sources of uncertainties. However, it is interesting and surprising to notice how the central nervous system can coordinate gracefully our movements to deal with these uncertainties. For example, one may be clumsy in moving an object with uncertain mass and unknown friction at the first time, but after several trials, the movements will gradually become smooth. Although extensive research by many authors has been made, the underlying computational mechanisms in sensorimotor control require further investigations.

From different aspects, many theories have been proposed to explain the computational nature of sensorimotor control; see the review article [43]. One widely accepted view is that the central nervous system (CNS) prefers trajectories produced by minimizing some cost function. This perspective has inspired quite a few optimization-based models for motor control (see [33, 40, 50, 55, 77, 131, 140, 163, 162, 165], and references therein). These models can explain many characteristics of motor control, such as approximately straight movement trajectories and the bell-shaped velocity curves reported by [120]. However, these models assume the CNS knows and uses the

dynamics of both the motor system and the interactive environment. Consequently, an indirect control scheme is assumed. Namely, the CNS first identifies all the system dynamics, and then finds the optimal control policies based on the identified information. This identification-based idea has also been used to study motor adaptation under external perturbations [9, 13, 31, 92, 142, 195]. Nevertheless, this viewpoint is difficult to be justified theoretically and has not been convincingly validated by experiments. Using self-generated perturbation, [59] reported that disturbance may not be identified by the CNS, and the control policy may not necessarily be optimal in the presence of uncertainties. Indeed, when uncertainties, especially dynamic uncertainties, occur, it becomes difficult to maintain not only optimality, but also stability. Since the optimization-based model may not be suitable to study the behavior and stability of motor systems, developing a new theoretical modeling framework is not only necessary but also of great importance.

The primary objective of this chapter is to study sensorimotor control with static and dynamic uncertainties under the framework of RADP [68, 69, 72, 73, 79]. In this chapter, the linear version of RADP is extended for stochastic systems by taking into account signal-dependent noise [50], and the proposed method is applied to study the sensorimotor control problem with both static and dynamic uncertainties.

There are two main advantages of the proposed modeling strategy. First of all, as a non-model-based approach, RADP shares some essential features with reinforcement learning (RL) [155], which is originally inspired by learning mechanisms observed in biological systems. RL concerns how an agent should modify its actions to interact with the unknown environment and to achieve a long-term goal. In addition, certain brain areas that can realize the steps of RL have been discussed by [36]. Like in many other ADP-based methods, the proposed RADP theory solves the Bellman equation [8] iteratively using real-time sensory information and can avoid the so-called "curse of dimensionality" in conventional dynamic programming. Also, rigorous convergence

analysis can be performed. Second, instead of identifying the dynamics of the overall dynamic system, we decompose the system into an interconnection of a simplified model (nominal system) with measurable state variables and the dynamic uncertainty (or, unmodeled dynamics) with unmeasurable state variables and unknown system order (see Figure 7.1). Then, we design the robust optimal control policy for the overall system using partial-state feedback. In this way, we can preserve optimality for the nominal reduced model as well as guarantee robust stability for the overall system. Compared with identification-based models, this modeling strategy is more realistic for sensorimotor systems for at least two reasons. First, identifying the exact model of both the motor system and the uncertain environment is not an easy task. Second, it is time-consuming and would yield slow response if the sensorimotor system first estimates all system variables before taking actions.

Detailed learning algorithms are presented in this chapter, and numerical studies are also provided. Interestingly, our computational results match well with experiments reported from the past literature [22, 41, 142]. The proposed theory also provides a unified theoretical framework that connects optimality and robustness. In addition, it links the stiffness geometry to the selection of the weighting matrices in the cost function. Therefore, we argue that the CNS may use RADP-like learning strategy to coordinate movements and to achieve successful adaptation in the presence of static and/or dynamic uncertainties. In the absence of the dynamic uncertainties, the learning strategy reduces to an ADP-like mechanism.

7.1 ADP for continuous-time stochastic systems

7.1.1 Problem formulation

To study sensorimotor control, we consider the following system governed by stochastic differential equations:

$$dx = Axdt + Budt + B\Sigma_{i=1}^{q} C_i u d\eta_i \tag{7.1}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are constant matrices describing the system dynamics with the pair (A, B) assumed to be stabilizable (i.e., there exists some constant matrix $K_0 \in \mathbb{R}^{m\times n}$ such that $A-BK_0$ is a Hurwitz matrix in the sense that all its eigenvalues are in the open left-half plane), $u \in \mathbb{R}^m$ is the control signal, η_i are independent scalar Brownian motions and $C_i \in \mathbb{R}^{m\times m}$ are constant matrices, for $i = 1, 2, \cdots, q$.

The control objective is to determine a linear control policy

$$u = -Kx \tag{7.2}$$

which minimizes the following cost function

$$J = \int_0^\infty (x^T Q x + u^T R u)dt \tag{7.3}$$

for the nominal system of (7.1) (i.e., system (7.1) with $\eta_i = 0$, $\forall i = 1, 2, \cdots, m$), where $Q = Q^T \geq 0$, $R = R^T > 0$, with $(A, Q^{1/2})$ observable. By observability, we mean that the solution $x(t)$ of the system $dx = Axdt$ is identically zero when the output $y = Q^{1/2}x$ is identically zero [99].

According to linear optimal control theory [99], when both A and B are accurately known, solution to this problem can be found by solving the following well-known

algebraic Riccati equation (ARE)

$$A^T P + PA + Q - PBR^{-1}B^T P = 0. \tag{7.4}$$

By the assumptions mentioned above, (7.4) has a unique symmetric positive definite solution $P^* \in \mathbb{R}^{n\times n}$. The optimal feedback gain matrix K^* in (7.2) can thus be determined by

$$K^* = R^{-1}B^T P^*. \tag{7.5}$$

In the presence of the signal-dependent noise η_i, the closed-loop system is mean-square stable [90], if

$$\begin{aligned} 0 \;>\; & (A - BK^*) \otimes I_n + I_n \otimes (A - BK^*) \\ & + \Sigma_{i=1}^{q} \left(BC_i K^* \otimes BC_i K^*\right). \end{aligned} \tag{7.6}$$

If the constant matrices C_i, $i = 1, 2, \cdots, q$, are so small that (7.6) holds, the control policy $u = -K^* x$ is called *robust optimal*, i.e., it is optimal in the absence of the noise η_i, and is stabilizing in the presence of η_i.

7.1.2 Policy iteration

To solve (7.4) which is nonlinear in P, the policy iteration algorithm from reinforcement learning can be applied [91], and in [91], it has been proved that the sequences $\{P_k\}$ and $\{K_k\}$ iteratively determined from policy iteration (7.7) and (7.8) have the following properties:

1) $A - BK_k$ is Hurwitz,

2) $P^* \leq P_{k+1} \leq P_k$, and

3) $\lim_{k\to\infty} K_k = K^*$, $\lim_{k\to\infty} P_k = P^*$.

Algorithm 7.1.1 Policy iteration with control-dependent noise

1: Find an initial stabilizing feedback gain matrix K_0, such that $A - BK_0$ is Hurwitz.
2: Solve P_k from

$$0 = (A - BK_k)^T P_k + P_k(A - BK_k) + Q + K_k^T R K_k \tag{7.7}$$

3: Improve the control policy by

$$K_{k+1} = R^{-1}B^T P_k \tag{7.8}$$

4: Go to Step 2) and solve for P_{k+1} with K_k replaced by K_{k+1}.

7.1.3 ADP for linear stochastic systems with signal-dependent noise

The policy iteration algorithm relies on the perfect knowledge of the system dynamics, because the system matrices A and B are involved in the equations (7.7) and (7.8). In [68], it has been shown that in the deterministic case, when A and B are unknown, equivalent iterations can be achieved using online measurements. Here we extend the methodology by [68] to deal with stochastic linear systems with signal-dependent noise, and to find online the optimal control policy without assuming the a priori knowledge of A and B.

To begin with, let us rewrite the original system (7.1) as

$$dx = (A - BK_k)xdt + B(dw + K_k x dt) \tag{7.9}$$

where

$$dw = udt + \Sigma_{i=1}^{q} C_i u d\eta_i \tag{7.10}$$

represents the combined signal received by the motor plant from the input channel.

Now, let us define $A_k = A - BK_k$, $Q_k = Q + K_k^T R K_k$, and $M_k = B^T P_k B$. Then, by Itô's lemma [62], along the solutions of (7.53), it follows that

$$\begin{aligned} & d(x^T P_k x) \\ = \; & dx^T P_k x + x^T P_k dx + dx^T P_k dx \\ = \; & x^T (A_k^T P_k + P_k A_k) x dt + 2(K_k x dt + dw)^T B^T P_k x \\ & + dw^T B^T P_k B dw \\ = \; & -x^T Q_k x dt + 2(K_k x dt + dw)^T B^T P_k x + dw^T M_k dw \end{aligned} \tag{7.11}$$

Notice that $dw^T M_k dw \neq 0$ because

$$\begin{aligned} & dw^T M_k dw \\ = \; & (u dt + \Sigma_{i=1}^q C_i u d\eta_i)^T M_k (u dt + \Sigma_{i=1}^q C_i u d\eta_i) \\ = \; & u^T M_k u (dt)^2 + \Sigma_{i=1}^q u^T C_i^T M_k C_i u (d\eta_i)^2 \\ & + \Sigma_{1 \leq i \neq j \leq q} u^T C_i^T M_k C_i u d\eta_i d\eta_j \\ = \; & \Sigma_{i=1}^q u^T C_i^T M_k C_i u dt \end{aligned}$$

Next, integrating both sides of (7.55) from t to $t + \delta t$, we obtain

$$\begin{aligned} & x(t+\delta t)^T P_k x(t+\delta t) - x(t)^T P_k x(t) \\ = \; & -\int_t^{t+\delta t} \left(x^T Q x + u_k^T R u_k \right) dt + \int_t^{t+\delta t} dw^T M_k dw \\ & + 2 \int_t^{t+\delta t} (K_k x + dw)^T R K_{k+1} x \end{aligned} \tag{7.12}$$

where $u_k = -K_k x$.

Notice that (7.59) plays an important role in separating accurately the system dynamics from the iterative process. As a result, the requirement of the system matrices in (7.7) and (7.8) can now be replaced by the state and input information

measured in real-time.

We now show that given a matrix K_k such that $A - BK_k$ is Hurwitz, a pair of matrices (P_k, K_{k+1}), with $P_k = P_k^T > 0$, satisfying (7.7) and (7.8) can be uniquely determined without knowing A or B . To this end, recall that we have defined the following two operators:

$$\begin{aligned} P \in \mathbb{R}^{n\times n} &\rightarrow \hat{P} \in \mathbb{R}^{\frac{1}{2}n(n+1)} \\ x \in \mathbb{R}^{n} &\rightarrow \bar{x} \in \mathbb{R}^{\frac{1}{2}n(n+1)} \end{aligned}$$

where

$$\begin{aligned} \hat{P} &= [p_{11}, 2p_{12}, \cdots, 2p_{1n}, p_{22}, 2p_{23}, \cdots, 2p_{n-1,n}, p_{nn}]^T, \\ \bar{x} &= [x_1^2, x_1x_2, \cdots, x_1x_n, x_2^2, x_2x_3, \cdots, x_{n-1}x_n, x_n^2]^T. \end{aligned}$$

Therefore, by Kronecker product representation [56], we have

$$(K_k x dt + dw)^T R K_{k+1} x = [x \otimes R(K_k x dt + dw)]^T \operatorname{vec}(K_{k+1}).$$

Further, for a sufficiently large positive integer $l_k > 0$, we define matrices $\delta_{x,k} \in \mathbb{R}^{l_k \times \frac{1}{2}n(n+1)}$, $I_{q,k} \in \mathbb{R}^{l_k}$, $I_{xv,k} \in \mathbb{R}^{l_k \times mn}$, and $I_{u,k} \in \mathbb{R}^{l_k \times \frac{1}{2}m(m+1)}$, such that

$$\delta_{x,k} = \begin{bmatrix} \bar{x}^T(t_{1,k}) - \bar{x}^T(t_{0,k}) \\ \bar{x}^T(t_{2,k}) - \bar{x}^T(t_{1,k}) \\ \vdots \\ \bar{x}^T(t_{l_k,k}) - \bar{x}^T(t_{l_k-1,k}) \end{bmatrix},$$

$$I_{q,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} \left(x^T Q x + u_k^T R u_k\right) dt \\ \int_{t_{1,k}}^{t_{2,k}} \left(x^T Q x + u_k^T R u_k\right) dt \\ \cdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} \left(x^T Q x + u_k^T R u_k\right) dt \end{bmatrix},$$

$$I_{xv,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} x^T \otimes (K_k x dt + dw)^T R \\ \int_{t_{1,k}}^{t_{2,k}} x^T \otimes (K_k x dt + dw)^T R \\ \vdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} x^T \otimes (K_k x dt + dw)^T R \end{bmatrix},$$

$$I_{u,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} \bar{w}^T d\tau \\ \int_{t_{1,k}}^{t_{2,k}} \bar{w}^T d\tau \\ \cdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} \bar{w}^T d\tau \end{bmatrix}$$

where $0 \leq t_{l_{k-1},k-1} \leq t_{0,k} < t_{1,k} < \cdots < t_{l_k,k} < t_{0,k+1}$.

Therefore, (7.59) implies the following compact form of linear equations

$$\Theta_k \begin{bmatrix} \hat{P}_k \\ \hat{M}_k \\ \mathrm{vec}\,(K_{k+1}) \end{bmatrix} = -I_{q,k} \tag{7.13}$$

where $\Theta_k \in \mathbb{R}^{l_k \times \left[\frac{1}{2}n(n+1)+mn\right]}$ is defined as:

$$\Theta_k = \left[\delta_{x,k}, -I_{u,k}, -2I_{xv,k}\right],$$

To guarantee the existence and uniqueness of solution to (7.62), we assume Θ_k has

full column rank for all $k \in \mathbb{Z}_+$. As a result, (7.62) can be directly solved as follows:

$$\begin{bmatrix} \hat{P}_k \\ \hat{M}_k \\ \mathrm{vec}\,(K_{k+1}) \end{bmatrix} = -(\Theta_k^T \Theta_k)^{-1} \Theta_k^T I_{q,k}. \tag{7.14}$$

It is worth noticing that when the sensory noise is taken into account, numerical errors may occur when computing the matrices $I_{q,k}$, $I_{xv,k}$, and $I_{u,k}$. Consequently, the solution of (7.14) can be viewed as the least squares solution of (7.62). Alternatively, an approximation of the unique solution of (7.62) can be obtained using a recursive least-squares method [111].

7.1.4 The ADP algorithm

Now, we are ready to give the following ADP algorithm for practical online implementation.

Algorithm 7.1.2 Adaptive dynamic programming algorithm

1: Find an initial stabilizing control policy $u_0 = -K_0 x$, and set $k = 0$.
2: Apply $u_k = -K_k x$ as the control input on the time interval $[t_{0,k},\ t_{l_k,k}]$. Compute $\delta_{x,k}$, $I_{q,k}$, $I_{xv,k}$, and $I_{u,k}$.
3: Solve P_k, M_k, and K_{k+1} from (7.14).
4: Let $k \leftarrow k + 1$, and go to Step 2.

Compared with most of the existing models for motor adaptation, the proposed ADP algorithm can be used to study both the online learning during one single trial and the learning among different trials. In the latter case, the interval $[t_{0,k},\ t_{l_k,k}]$ should be taken from the time duration of a single trial.

7.1.5 Convergence analysis

The convergence property of the proposed algorithm can be summarized in the following Theorem.

Theorem 7.1.1. *Suppose Θ_k has full column rank for all $k \in \mathbb{Z}_+$ and $A - BK_0$ is Hurwitz. Then, the sequences $\{K_k\}$, $\{P_k\}$, and $\{M_k\}$ obtained from (7.14) satisfy $\lim_{k\to\infty} P_k = P^*$, $\lim_{k\to\infty} K_k = K^*$, and $\lim_{k\to\infty} M_k = B^T P^* B$.*

Proof. Given a stabilizing feedback gain matrix K_k, if $P_k = P_k^T$ is the solution of (7.7), K_{k+1} and M_k are uniquely determined by $K_{k+1} = R^{-1}B^T P_k$ and $M_k = B^T P_k B$, respectively. By (7.59), we know that P_k, K_{k+1}, and M_k satisfy (7.14).

On the other hand, let $P = P^T \in \mathbb{R}^{n\times n}$, $M \in \mathbb{R}^{m\times m}$, and $K \in \mathbb{R}^{m\times n}$, such that

$$\Theta_k \begin{bmatrix} \hat{P} \\ \hat{M} \\ \mathrm{vec}(K) \end{bmatrix} = \Xi_k. \tag{7.15}$$

Then, we immediately have $\hat{P} = \hat{P}_k$, $\hat{M} = \hat{M}_k$, and $\mathrm{vec}(K) = \mathrm{vec}(K_{k+1})$. Since Θ_k has full column rank, $P = P^T$, $M = M^T$, and K are unique. In addition, by the definitions of $\hat{P}$, $\hat{M}$, and $\mathrm{vec}(K)$, $P_k = P$, $M_k = M$ and $K_{k+1} = K$ are uniquely determined. Therefore, the policy iteration (7.14) is equivalent to (7.7) and (7.8). By [91], the convergence is thus proved. □

7.2 RADP for continuous-time stochastic systems

7.2.1 Problem formulation

In this paper, we generalize the commonly used linear models for sensorimotor control ([50, 40, 63, 163]) by taking into account the dynamic uncertainty, or unmodeled

dynamics. To be more specific, consider the stochastic differential equations

$$
\begin{aligned}
dw &= Fwdt + Gxdt && (7.16)\\
dx &= Axdt + B_1\left[zdt + \Delta_1(w,x)\,dt + \sum_{i=1}^{q_1} E_{1i} z d\eta_{1i}\right] && (7.17)\\
dz &= B_2\left[udt + \Delta_2(w,x,z)\,dt + \sum_{i=1}^{q_2} E_{2i} u d\eta_{2i}\right] && (7.18)\\
\Delta_1 &= D_{11}w + D_{12}x && (7.19)\\
\Delta_2 &= D_{21}w + D_{22}x + D_{23}z && (7.20)
\end{aligned}
$$

where $[x^T, z^T]^T \in \mathbb{R}^{n+m}$ is the measurable state which will be used to represent the states of the sensorimotor system, $w \in \mathbb{R}^{n_w}$ is the unmeasurable state of the dynamic uncertainty, representing the unknown dynamics in the interactive environment, Δ_1 and Δ_2 are the outputs of the dynamic uncertainty, A, B_1, B_2, F, G, E_{1i} with $i = 1, 2, \cdots, q_1$, and E_{2i} with $i = 1, 2, \cdots, q_2$ are unknown constant matrices with suitable dimensions and $B_2 \in \mathbb{R}^{m\times m}$ is assumed to be invertiable, η_{1i} with $i = 1, 2, \cdots, q_1$ and η_{2i} with $i = 1, 2, \cdots, q_2$ are independent scalar Brownian motions, $u \in \mathbb{R}^m$ denotes the input of the motor control command.

Design objective: Find a robust optimal feedback control policy which

1. robustly stabilizes the overall system (7.16)-(7.20), and
2. is optimal for the nominal system, i.e., the system comprised of (7.17) and (7.18) with $\Delta_1 \equiv 0$, $\Delta_2 \equiv 0$, $E_{1i} = 0$, and $E_{2i} = 0$.

For this purpose, let us introduce an assumption on the dynamic uncertainty, which is modeled by the w-subsystem.

Assumption 7.2.1. *There exist $S = S^T > 0$ and a constant $\gamma_w > 0$, such that*

$$SF + F^T S + I + \gamma_w^{-1} SGG^T S < 0. \qquad (7.21)$$

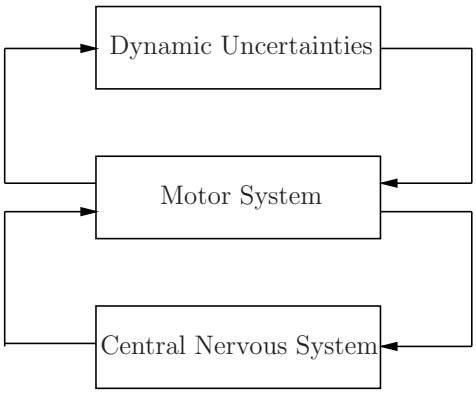

Figure 7.1: RADP framework for sensorimotor control.

Remark 7.2.1. *Assumption 7.2.1 implies that the dynamic uncertainty, described by the w-subsystem, is finite-gain L_2 stable with a linear gain smaller than $\sqrt{\gamma_w}$, when x is considered as the input and w is considered as the output [93].*

7.2.2 Reduced-order system design

Consider the reduced-order system comprised of (7.16), (7.17), and (7.19) with z regarded as the input. For convenience, the system is rewritten as follows:

$$dw = Fwdt + Gxdt \tag{7.22}$$

$$dx = Axdt + B_1 \left[zdt + \Delta_1 (w, x)\, dt + \sum_{i=1}^{q_1} E_{1i} z d\eta_{1i} \right] \tag{7.23}$$

$$\Delta_1 = D_{11} w + D_{12} x \tag{7.24}$$

and the related nominal deterministic system is defined as

$$dx = Axdt + B_1 zdt \tag{7.25}$$

$$\Delta_1 = D_{11} w + D_{12} x \tag{7.26}$$

The cost associated with the deterministic system comprised of (7.25) and (7.26)

is selected to be

$$J_1 = \int_0^\infty \left(x^T Q_1 x + z^T R_1 z\right) dt \tag{7.27}$$

where $Q_1 = Q_1^T \geq 0$, $R_1 = R_1^T > 0$, and the pair $(A, Q_1^{1/2})$ is observable.

By linear optimal control theory [99], the optimal control policy takes the following form

$$z = -R_1^{-1} B_1^T P_1 x \tag{7.28}$$

where $P_1 = P_1^T > 0$ is the solution of the following algebraic Riccati Equation (ARE):

$$A^T P_1 + P_1 A + Q_1 - P_1 B_1 R_1^{-1} B_1^T P_1 = 0. \tag{7.29}$$

The following concept on mean-square stability [194] will be used in the remainder of the paper.

Definition 7.2.1. *Consider the system*

$$dx = Axdt + \sum_{i=1}^{q} B_i x d\eta_i \tag{7.30}$$

where η_i with $i = 1, 2, \cdots, q$ are standard scalar Brownian motions. Then, the system is said to be stable in the mean-square sense if

$$\lim_{t\to\infty} E\left[x(t)x(t)^T\right] = 0. \tag{7.31}$$

Now, the following theorem gives stability criteria of the closed-loop system comprised of (7.22), (7.23), (7.24), and (7.28).

Theorem 7.2.1. *The closed-loop system comprised of (7.22), (7.23),* (7.24)*, and*

(7.28) is mean-square stable if

1. *the weighting matrices Q_1 and R_1 are selected such that*

$$Q_1 > (\kappa_{12} + \kappa_{11}\gamma_w)\, I_n \quad \text{and} \quad R_1 < I_m \tag{7.32}$$

   where $\kappa_{11} = 2|D_{11}|^2$ and $\kappa_{12} = 2|D_{12}|^2$.

2. *the constant matrices E_{1i} with $i = 1, 2, \cdots, q_1$ satisfy*

$$\sum_{i=1}^{q_1} E_{1i}^T B_1^T P_1 B_1 E_{1i} \leq R_1(I_m - R_1) \tag{7.33}$$

Proof. First, we define $\mathcal{L}(\cdot)$ as the infinitesimal generator [98]. Then, along the trajectories of the x-subsystem, (7.23), we have

$$\begin{aligned}
\mathcal{L}(x^T P_1 x) &= -x^T \left(Q_1 + P_1 B_1 R^{-1} B_1^T P_1\right) x + 2x^T P_1 B_1 \Delta_1 \\
&\quad + x^T P_1 B_1 R_1^{-1} \sum_{i=1}^{q_1} E_{1i}^T B_1^T P_1 B_1 E_{1i} R_1^{-1} B_1^T P_1 x \\
&= -x^T Q_1 x - |\Delta_1 - B_1^T P_1^T x|^2 dt + |\Delta_1|^2 \\
&\quad - x^T P_1 B_1 R_1^{-1} \left[R_1 - R_1^2 - \sum_{i=1}^{q_1} E_{1i}^T B_1^T P_1 B_1 E_{1i} \right] \\
&\quad \times R_1^{-1} B_1^T P_1 x \\
&\leq -x^T Q_1 x + |\Delta_1|^2
\end{aligned}$$

On the other hand, under Assumption 7.2.1, along the solutions of the w-subsystem (7.22), we have

$$\begin{aligned}
\mathcal{L}\left(w^T S w\right) &= w^T (SF + F^T S) w + w^T S G x + x^T G^T S w \\
&< -|w|^2 - {\gamma_w}^{-1} w^T S G G^T S w + w^T S G x + x^T G^T S w \\
&< -|w|^2 + \gamma_w |x|^2 \qquad\qquad (\forall w \neq 0)
\end{aligned}$$

By definition, we know that

$$\begin{aligned} |\Delta_1|^2 &= |D_{11}w + D_{12}x|^2 \\ &\leq 2|D_{11}|^2|w|^2 + 2|D_{12}|^2|x|^2 \\ &\leq \kappa_{11}|w|^2 + \kappa_{12}|x|^2 \end{aligned}$$

Therefore, for all $(x, w) \neq 0$, the following holds

$$\begin{aligned} & \mathcal{L}(x^T P_1 x + \kappa_{11} w^T S w) \\ < \; & -\gamma_x |x|^2 + |\Delta_1|^2 - \kappa_{11}|w|^2 + \kappa_{11}\gamma_w |x|^2 \\ < \; & -\gamma_x |x|^2 + \kappa_{11}|w|^2 + 2x^T D_{12}^T D_{12} x - \kappa_{11}|w|^2 + \kappa_{11}\gamma_w |x|^2 \\ < \; & -x^T \left(Q_1 - \kappa_{12} I_n - \kappa_{11}\gamma_w I_n\right) x \\ < \; & 0 \end{aligned}$$

Notice that $V(w, x) = x^T P_1 x + \kappa_{11} w^T S w$ can be regarded as a stochastic Lyapunov function [98], and the proof is thus complete. □

7.2.3 Optimal control design for the full system

Now we proceed ahead to study the full system. Define the transformation

$$\xi = z + K_1 x \tag{7.34}$$

where $K_1 = R_1^{-1} B_1^T P_1$. Then, we have

$$\begin{aligned} d\xi &= dz + K_1 dx \\ &= B_2 \left[u dt + \Delta_2 dt + \sum_{i=1}^{q_2} E_{2i} u d\eta_{2i} \right] \\ &\quad + K_1 (A_c x dt + B_1 \Delta_1 dt + B_1 \xi dt + B_1 \sum_{i=1}^{q_1} E_{1i} z d\eta_{1i}) \\ &= K_1 A_c x dt + B_2 u dt + B_2 \bar{\Delta}_2 dt + K_1 B_1 \xi dt \\ &\quad + \sum_{i=1}^{q_2} B_2 E_{2i} u d\eta_{2i} + K_1 \sum_{i=1}^{q_1} B_1 E_{1i} (\xi - K_1 x) d\eta_{1i} \end{aligned}$$

where $\bar{\Delta}_2 = B_2^{-1} K_1 B_1 \Delta_1 + \Delta_2$ and $A_c = A - B_1 K_1$.

Consequently, the system (7.16)-(7.20) is converted to

$$dw = Fw dt + Gx dt \tag{7.35}$$

$$dx = A_c x dt + B_1 \left[\xi dt + \Delta_1 dt + \sum_{i=1}^{q_1} E_{1i} (\xi - K_1 x) d\eta_{1i} \right] \tag{7.36}$$

$$\begin{aligned} d\xi &= K_1 A_c x dt + B_2 u dt + B_2 \bar{\Delta}_2 dt + K_1 B_1 \xi dt \\ &\quad + \sum_{i=1}^{q_2} B_2 E_{2i} u d\eta_{2i} + K_1 \sum_{i=1}^{q_1} B_1 E_{1i} (\xi - K_1 x) d\eta_{1i} \end{aligned} \tag{7.37}$$

Now, let us consider the control policy

$$u = -R_2^{-1} B_2^T P_2 \xi = -R_2^{-1} B_2^T P_2 \left(z + R_1^{-1} B_1 P_1 x \right) \tag{7.38}$$

where $P_2 = P_2^T > 0$ is the solution of the following equation

$$Q_2 - P_2 B_2 R_2^{-1} B_2^T P_2 = 0 \tag{7.39}$$

with Q_2 and R_2 two positive definite and symmetric matrices.

Remark 7.2.2. *Notice that* (7.39) *is an ARE associated with the following optimal*

control problem

$$\min_{u} \quad J_2 = \int_0^{\infty} \left(z^T Q_2 z + u^T R_2 u \right) dt, \tag{7.40}$$

$$\text{s.t.} \quad \dot{z} = B_2 u, \tag{7.41}$$

where (7.41) *is the nominal system of* (7.18).

The stability criteria are given in the following theorem.

Theorem 7.2.2. *The closed-loop system comprised of (7.35)-(7.37) and (7.38) is mean-square stable if*

1. *the weighting matrices Q_1 and R_1 as defined in* (7.29)*, and Q_2 and R_2 as defined in* (7.39) *are selected such that $R_1 < I_m$, $R_2 < I_m$ and*

$$\begin{bmatrix} \left(\kappa_{12} + \kappa_{22} + \gamma_w \kappa_{11} + \gamma_w \kappa_{21}\right) I_n & -A_c^T K_1^T P_2 - P_1 B_1 \\ -P_2 K_1 A_c - B_1^T P_1 & \left(\kappa_{23} + \kappa_3\right) I_m \end{bmatrix} < \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \tag{7.42}$$

2. *the constant matrices E_{1i} and E_{2i} satisfy the following inequalities*

$$\sum_{i=1}^{q_1} E_{1i}^T B_1^T \left(P_1 + K_1^T P_2 K_1 \right) B_1 E_{1i} \leq \frac{1}{2} R_1 \left(I_m - R_1 \right) \tag{7.43}$$

$$\sum_{i=1}^{q_2} E_{2i}^T B_2^T P_2 B_2 E_{2i} \leq R_2 \left(I_m - R_2 \right) \tag{7.44}$$

where

$$\begin{aligned}
\kappa_{21} &= 3|B_2^{-1}K_1B_1D_{11} + D_{21}|^2, \\
\kappa_{22} &= 3|B_2^{-1}K_1B_1D_{12} + D_{22} - D_{23}K_1^*|^2, \\
\kappa_{23} &= 3|D_{23}|^2, \\
\kappa_3 &= |2\sum_{i=1}^{q_1} E_{1i}^T B_1^T K_1^T P_2 B_1 K_1 E_{1i} - P_2K_1B_1 - B_1^T K_1^T P_2|
\end{aligned}$$

Proof. Along the trajectories of the x-subsystem (7.17), we have

$$\begin{aligned}
\mathcal{L}(x^TP_1x) &\leq -x^TQ_1x - x^TP_1B_1\left(R_1^{-1} - I_m\right)B_1^TP_1x \\
&\quad -x^TP_1B_1B_1^TP_1x + 2x^TP_1B_1(\Delta_1 + \xi) \\
&\quad + (\xi - K_1x)^T\sum_{i=1}^{q_1} E_{1i}^TB_1^TP_1B_1E_{1i}\left(\xi - K_1x\right) \\
&\leq -x^TQ_1x + |\Delta_1|^2 + 2x^TP_1B_1\xi \\
&\quad -x^TP_1B_1\left(R^{-1} - I_m\right)B_1^TP_1x \\
&\quad +2x^TK_1^T\sum_{i=1}^{q_1} E_{1i}^TB_1^TP_1B_1E_{1i}K_1x \\
&\quad +2\xi^T\sum_{i=1}^{q_1} E_{1i}^TB_1^TPB_1E_{1i}\xi \\
&\leq -x^TQ_1x + |\Delta_1|^2 + 2x^TP_1B_1\xi \\
&\quad -x^TK_1^T\left[R_1 - R_1^2 - 2\sum_{i=1}^{q_1} E_{1i}^TB_1^TP_1B_1E_{1i}\right]K_1x \\
&\quad +2\xi^T\sum_{i=1}^{q_1} E_{1i}^TB_1^TP_1B_1E_{1i}\xi
\end{aligned}$$

Then, along the trajectories of the ξ-subsystem,

$$\begin{aligned}
\mathcal{L}(\xi^T P_2 \xi) &= 2\xi^T P_2 \left(K_1 A_c x + B_2 u + B_2 \bar{\Delta}_2 + K_1 B_1 \xi \right) \\
&\quad + \sum_{i=1}^{q_1} \left(\xi - K_1 x \right)^T E_{1i}^T B_1^T K_1^T P_2 K_1 B_1 E_{1i} \left(\xi - K_1 x \right) \\
&\quad + \sum_{i=1}^{q_2} \xi^T K_2^T E_{2i}^T B_2^T P_2 B_2 E_{2i} K_2 \xi \\
&= -\xi^T \left(Q_2 - 2 \sum_{i=1}^{q_1} E_{1i}^T B_1^T K_1^T P_2 K_1 B_1 E_{1i} \right) \xi \\
&\quad + \xi^T \left(P_2 K_1 B_1 + B_1^T K_1^T P_2 \right) \xi \\
&\quad + 2\xi^T P_2 K_1 A_c x + |\bar{\Delta}_2|^2 \\
&\quad + 2 \sum_{i=1}^{q_1} x^T K_1^T E_{1i}^T B_1^T K_1^T P_2 K_1 B_1 E_{1i} K_1 x \\
&\quad - \sum_{i=1}^{q_2} \xi^T K_2^T \left(R_2 - R_2^2 - E_{2i}^T B_2^T P_2 B_2 E_{2i} \right) K_2 \xi
\end{aligned}$$

Also, by definition

$$\begin{aligned}
|\bar{\Delta}_2|^2 &= | \left(B_2^{-1} K_1 B_1 D_{11} + D_{21} \right) w \\
&\quad + \left(B_2^{-1} K_1 B_1 D_{12} + D_{22} \right) x \\
&\quad + D_{23} \left(\xi - K_1 x \right) |^2 \\
&= | \left(B_2^{-1} K_1 B_1 D_{11} + D_{21} \right) w \\
&\quad + \left(B_2^{-1} K_1 B_1 D_{12} + D_{22} - D_{23} K_1 \right) x \\
&\quad + D_{23} \xi |^2 \\
&\leq \kappa_{21} |w|^2 + \kappa_{22} |x|^2 + \kappa_{23} |\xi|^2
\end{aligned}$$

Finally, along solutions of the closed-loop system (7.16)-(7.20) and (7.38), the

following holds for all $[w^T, x^T, z^T] \neq 0$

$$
\begin{aligned}
& \mathcal{L}\left[\left(\kappa_{11}+\kappa_{21}\right) w^T S w + x^T P_1 x + \xi^T P_2 \xi\right]) \\
\leq & -\begin{bmatrix} x \\ \xi \end{bmatrix}^T \begin{bmatrix} Q_1 - \left(\kappa_{12}+\kappa_{22}+\kappa_{11}\gamma+\kappa_{21}\gamma\right) I_n & A_c^T K_1^T P_2 + P_1 B_1 \\ P_2 K_1 A_c + B_1^T P_1 & Q_2 - \left(\kappa_{23}+\kappa_3\right) I_m \end{bmatrix} \\
& \times \begin{bmatrix} x \\ \xi \end{bmatrix} - x^T K_1^T \left(R_1 - R_1^2\right) K_1 x \\
& + 2 x^T K_1^T \sum_{i=1}^{q_1} E_{1i}^T B_1^T \left(P_1 + K_1^T P_2 K_1\right) B_1 E_{1i} K_1 x \\
& - \xi^T K_2^T \left(R_2 - R_2^2 - \sum_{i=1}^{q_2} E_{2i}^T B_2^T P_2 B_2 E_{2i}\right) K_2 \xi \\
< & \ 0
\end{aligned}
$$

The proof is complete. □

Remark 7.2.3. *Consider the nominal system (7.35)-(7.37). In the absence of dynamic uncertainties (i.e., the w-subsystem is absent, $\Delta_1 \equiv 0$, $\Delta_2 \equiv 0$, $E_{1i} = 0$ for $i = 1, 2, \cdots, q_1$, and $E_{2i} = 0$ for $i = 1, 2, \cdots, q_2$), the control policy (7.38) is optimal in the sense that it minimizes the cost*

$$
J_2 = \int_0^\infty \left(\begin{bmatrix} x \\ \xi \end{bmatrix}^T \bar{Q}_2 \begin{bmatrix} x \\ \xi \end{bmatrix} + u^T R_2 u \right) dt \tag{7.45}
$$

where

$$
\bar{Q}_2 = \begin{bmatrix} Q_1 + K_1^T R_1 K_1 & A_c^T K_1^T P_2 + P_1 B_1 \\ P_2 K_1 A_c + B_1^T P_1 & Q_2 - P_2 K_1 B_1 - B_1^T K_1^T P_2 \end{bmatrix} > 0
$$

Notice that the design methodology is different from inverse optimal control [96]. Indeed, although the weighing matrix $\bar{Q}_2$ cannot be arbitrary specified, it can be indi-

rectly modified by tuning the matrices Q_1, Q_2, R_1, and R_2. In motor control systems, Q_1, Q_2, and R_1 are related with the weights assigned on the movement accuracy, while R_2 represents the weights assigned on the control effort.

7.2.4 Off-line policy iteration technique

In order to obtain the robust optimal control policy (7.38), we need to first solve (7.29) and (7.39) which are nonlinear in P_1 and P_2, respectively. This can be done using the following off-line policy iteration algorithm.

Algorithm 7.2.1 Offline policy iteration technique

1: Find feedback gain matrices $K_{1,0}$ and $K_{2,0}$, such that $A - B_1K_{1,0}$ and $-B_2K_{2,0}$ are both Hurwitz (i.e., the real parts of their eigenvalues are all negative). Let $k = 0$ and

$$u_0 = -K_{2,0}z - K_{2,0}K_{1,0}x \tag{7.46}$$

2: Solve $P_{1,k}$ from

$$\begin{aligned} 0 &= (A - B_1K_{1,k})^T P_{1,k} + P_{1,k}(A - B_1K_{1,k}) \\ &\quad +Q_1 + K_{1,k}R_1K_{1,k} \end{aligned} \tag{7.47}$$

3: Solve $P_{2,k}$ from

$$0 = -B_2K_{2,k}^T P_{2,k} - P_{2,k}B_2K_{2,k} + Q_2 + K_{2,k}^T R_2 K_{2,k} \tag{7.48}$$

4: Let $k \leftarrow k + 1$, and improve the control policy by

$$u_k = -K_{2,k}z - K_{2,k}K_{1,k}x \tag{7.49}$$

where

$$K_{i,k} = R_i^{-1}B_i^T P_{i,k-1}, \quad \forall i = 1, 2 \tag{7.50}$$

5: Go to Step 2.

Remark 7.2.4. *The proposed RADP-based online learning method requires an initial stabilizing control policy. To be more specific, we need to know feedback gain matrices $K_{1,0}$ and $K_{2,0}$, such that $A - B_1K_{1,0}$ and $-B_2K_{2,0}$ are both Hurwitz. Even if A, B_1,*

and B_2 are uncertain, it is still possible to find such $K_{1,0}$ and $K_{2,0}$ when some upper and lower bounds of the elements in A, B_1, and B_2 are available. In practice, these bounds can be estimated by the CNS during the first several trials. Take the model in Section 4.2 as an example. In the absence of disturbances (i.e., $f \equiv 0$), we have

$$A - B_1 K_{1,0} = \begin{bmatrix} 0 & I_2 \\ 0 & -\frac{b}{m} I_2 \end{bmatrix} - \begin{bmatrix} 0 \\ \frac{1}{m} I_2 \end{bmatrix} K_{1,0} \tag{7.51}$$

and

$$B_2 K_{2,0} = \frac{1}{\tau} K_{2,0} \tag{7.52}$$

Since we know that $b > 0$, $m > 0$, and $\tau > 0$, we can choose, for example, $K_{1,0} = [I_2, 0]$ and $K_{2,0} = I_2$. Then, Algorithm 7.2.1 can proceed, with the resulted initial stabilizing control policy.

Convergence of this off-line policy iteration method can be concluded in the following theorem. The proof is omitted here because it is a trivial extension of the main theorem in [90].

Theorem 7.2.3. *The sequences $\{P_{i,k}\}$, $\{K_{i,k}\}$ with $i = 1, 2$ and $k = 0, 1, \cdots$ iteratively determined from Algorithm 7.2.1 have the following properties $\forall\ k = 0, 1, \cdots$.*

1) $A - B_1 K_{1,k}$ and $-B_2 K_{2,k}$ are both Hurwitz,

2) $0 < P_i \leq P_{i,k+1} \leq P_{i,k}$, and

3) $\lim_{k\to\infty} K_{i,k} = K_i$, $\lim_{k\to\infty} P_{i,k} = P_i$, $\forall i = 1, 2$.

7.2.5 Online implementation

Here, we will show how these iteration steps can be made using online sensory data without the need to identify the system dynamics.

To begin with, let us consider the reduced-order system and rewrite the x-subsystem (7.17) as

$$dx = (A - B_1 K_{1,k})xdt + B_1(dw_1 + K_{1,k}xdt) \tag{7.53}$$

where

$$dw_1 = zdt + \Delta_1 dt + \Sigma_{i=1}^{q} E_{1i} z d\eta_{1i} \tag{7.54}$$

represents the combined signal received by the motor plant from the input channel.

By Itô's lemma [62], along the solutions of (7.53), it follows that

$$\begin{aligned}
& d(x^T P_{1,k} x) \\
= \; & dx^T P_{1,k} x + x^T P_{1,k} dx + dx^T P_{1,k} dx \\
= \; & x^T (A_k^T P_{1,k} + P_{1,k} A_k) x dt \\
& + 2(K_{1,k} x dt + dw_1)^T B_1^T P_{1,k} x \\
& + dw_1^T B_1^T P_{1,k} B_1 dw_1 \\
= \; & -x^T Q_{1,k} x dt \\
& + 2(K_{1,k} x dt + dw_1)^T B_1^T P_{1,k} x \\
& + dw_1^T M_{1,k} dw_1
\end{aligned} \tag{7.55}$$

where

$$A_k = A - B_1 K_{1,k}, \tag{7.56}$$

$$Q_{1,k} = Q_1 + K_{1,k}^T R_1 K_{1,k}, \tag{7.57}$$

$$M_{1,k} = B_1^T P_{1,k} B_1. \tag{7.58}$$

Next, integrating both sides of (7.55) from t to $t+\delta t$, we obtain

$$\begin{aligned} & x(t+\delta t)^T P_{1,k} x(t+\delta t) - x(t)^T P_{1,k} x(t) \\ = & -\int_t^{t+\delta t} x^T Q_{1,k} x dt + \int_t^{t+\delta t} dw_1^T M_{1,k} dw_1 \\ & +2\int_t^{t+\delta t} (K_{1,k} x dt + dw_1)^T R_1 K_{1,k+1} x \end{aligned} \tag{7.59}$$

We now show that given a matrix $K_{1,k}$ such that $A - B_1 K_{1,k}$ is Hurwitz, a pair of matrices $(P_{1,k}, K_{1,k+1})$, with $P_{1,k} = P_{1,k}^T > 0$, satisfying (7.47) and (7.50) can be uniquely determined without knowing A or B_1 . To this end, we define the following two operators:

$$\begin{aligned} P \in \mathbb{R}^{n\times n} & \rightarrow \nu(P) \in \mathbb{R}^{\frac{1}{2}n(n+1)} \\ x \in \mathbb{R}^n & \rightarrow \mu(x) \in \mathbb{R}^{\frac{1}{2}n(n+1)} \end{aligned}$$

where

$$\begin{aligned} \nu(P) &= [p_{11}, 2p_{12}, \cdots, 2p_{1n}, p_{22}, 2p_{23}, \cdots, 2p_{n-1,n}, p_{nn}]^T, \\ \mu(x) &= [x_1^2, x_1x_2, \cdots, x_1x_n, x_2^2, x_2x_3, \cdots, x_{n-1}x_n, x_n^2]^T. \end{aligned}$$

In addition, by Kronecker product representation [56], we have

$$\begin{aligned} & (K_{1,k} x dt + dw_1)^T R_1 K_{1,k+1} x \\ = & [x \otimes R_1 (K_{1,k} x dt + dw_1)]^T \operatorname{vec}(K_{1,k+1}). \end{aligned}$$

Further, for a sufficiently large positive integer $l_k > 0$, we define the following two

matrices.

$$\Xi_{1,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} x^T Q_{1,k} x dt \\ \int_{t_{1,k}}^{t_{2,k}} x^T Q_{1,k} x dt \\ \vdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} x^T Q_{1,k} x dt \end{bmatrix} \in \mathbb{R}^{l_k} \tag{7.60}$$

$$\Theta_{1,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} d\theta_{1,k}^{(1)} \\ \int_{t_{1,k}}^{t_{2,k}} d\theta_{1,k}^{(2)} \\ \vdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} d\theta_{1,k}^{(l_k)} \end{bmatrix} \in \mathbb{R}^{l_k \times \left[\frac{n(n+1)+m(m+1)}{2}+mn\right]}$$

where $\theta_{1,k}^{(i)} \in \mathbb{R}^{1\times\left[\frac{n(n+1)+m(m+1)}{2}+mn\right]}$ is defined as

$$d\theta_{1,k}^{(i)} = \begin{bmatrix} \mu\left(x(t_{i+1,k})\right) - \mu\left(x(t_{i,k})\right) \\ -\mu\left(dw_1\right) \\ -2x \otimes R_1(K_{1,k}xdt + dw_1) \end{bmatrix}^T \tag{7.61}$$

and $t_{0,k} < t_{1,k} < \cdots < t_{l_k,k}$ are nonnegative constants denoting the time points during the movements when u_k is applied as the control policy.

Now, (7.59) implies the following compact form of linear equations

$$\Theta_{1,k} \begin{bmatrix} \nu(P_{1,k}) \\ \nu(M_{1,k}) \\ \mathrm{vec}\left(K_{1,k+1}\right) \end{bmatrix} = -\Xi_{1,k}. \tag{7.62}$$

To guarantee the existence and uniqueness of the solution to (7.62), we assume $\Theta_{1,k}$ has full column rank for all $k \in \mathbb{Z}_+$. As a result, (7.62) can be directly solved as

follows:

$$\begin{bmatrix} \nu(P_{1,k}) \\ \nu(M_{1,k}) \\ \mathrm{vec}\,(K_{1,k+1}) \end{bmatrix} = -(\Theta_{1,k}^T\Theta_{1,k})^{-1}\Theta_{1,k}^T\Xi_{1,k}. \tag{7.63}$$

Remark 7.2.5. *The rank condition is in the spirit of persistency of excitation in adaptive control and is a necessary condition for parameter convergence [60, 158]. In practice, if $\Theta_{1,k}$ does not satisfy the rank condition, the CNS needs to keep using the current control policy $z = -K_{1,k}x$ such that more online data can be collected and more rows can be appended into $\Theta_{1,k}$ until the rank condition of $\Theta_{1,k}$ is satisfied. In addition, exploration noise can be added along with the control signal, such that the persistency of excitation can be better achieved.*

Remark 7.2.6. *It is worth noticing that when the sensory noise is taken into account, numerical errors may occur when computing the matrices $I_{q,k}$, $I_{xv,k}$, and $I_{u,k}$. Consequently, the solution of* (7.63) *can be viewed as the least-squares solution of* (7.62)*. Alternatively, an approximation of the unique solution of* (7.62) *can be obtained using a recursive least-squares method [111].*

Similarly, for the z-subsystem, we have

$$\begin{aligned} & z(t+\delta t)^T P_{2,k} z(t+\delta t) - z(t)^T P_{2,k} z(t) \\ = & -\int_t^{t+\delta t} z^T Q_{2,k} z dt + \int_t^{t+\delta t} dw_2^T M_{2,k} dw_2 \\ & +2\int_t^{t+\delta t} (K_{2,k} z dt + dw_2)^T R_2 K_{2,k+1} z \end{aligned} \tag{7.64}$$

where

$$
\begin{aligned}
Q_{2,k} &= Q_2 + K_{2,k}^T R_2 K_{2,k}, && (7.65)\\
M_{2,k} &= B_2^T P_{2,k} B_2, && (7.66)\\
dw_2 &= udt + \Delta_2 dt + \sum_{i=1}^{q_2} E_{2i} u d\eta_{2i}. && (7.67)
\end{aligned}
$$

Further, we define matrices

$$
\Xi_{2,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} z^T Q_{2,k} z dt \\ \int_{t_{1,k}}^{t_{2,k}} z^T Q_{2,k} z dt \\ \vdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} z^T Q_{2,k} z dt \end{bmatrix} \in \mathbb{R}^{l_k},
$$

$$
\Theta_{2,k} = \begin{bmatrix} \int_{t_{0,k}}^{t_{1,k}} d\theta_{2,k}^{(1)} \\ \int_{t_{1,k}}^{t_{2,k}} d\theta_{2,k}^{(2)} \\ \vdots \\ \int_{t_{l_k-1,k}}^{t_{l_k,k}} d\theta_{2,k}^{(l_k)} \end{bmatrix} \in \mathbb{R}^{l_k \times \left(2m^2+m\right)}
$$

where

$$
d\theta_{2,k}^{(i)} = \begin{bmatrix} \mu(z(t_{i+1,k})) - \mu(z(t_{i,k})) \\ -\mu(dw_2) \\ -2x \otimes R_2(K_{2,k} z dt + dw_2) \end{bmatrix}^T \in \mathbb{R}^{1 \times \left(2m^2+m\right)}. \qquad (7.68)
$$

Then, the unknown matrices in each iteration step can be directly solved, that is,

$$\begin{bmatrix} \nu(P_{2,k}) \\ \nu(M_{2,k}) \\ \mathrm{vec}\,(K_{2,k+1}) \end{bmatrix} = -(\Theta_{2,k}^T\Theta_{2,k})^{-1}\Theta_{2,k}^T\Xi_{2,k}. \tag{7.69}$$

The motor learning algorithm based on the proposed RADP theory can thus be summarized as follows:

Algorithm 7.2.2 RADP-based motor learning algorithm

1: Apply an initial control policy in the form of (7.46), and let $k = 0$.
2: Collect online sensory data to compute the matrices $\Theta_{i,k}$ and $\Xi_{i,k}$ with $i = 1, 2$.
3: Let $k \leftarrow k + 1$, and update the control policy using (7.63) and (7.69). Then, apply the new control policy (7.49) to the motor system.
4: Go to Step 2.

Corollary 7.2.1. *Suppose $\Theta_{i,k}$ is of full-column rank for all $i = 1, 2$ and $k = 0, 1, \cdots$. Then, the control policies obtained from Algorithm 7.2.2 converge to the robust optimal control policy* (7.38).

Proof. This corollary is a direct result from Theorem 7.2.3 by noticing that the matrices $P_{i,k}$, $M_{i,k}$ and $K_{i,k+1}$ obtained from Algorithm 7.2.1 and Algorithm 7.2.2 are equivalent for $i = 1, 2$, and $k = 0, 1, \cdots$. □

Remark 7.2.7. *It is worth noticing that the past measurements on dw_1 and dw_2 are assumed available for online learning purpose. In the following sections, we will see that they correspond to the combined control signals received by the muscles. These signals can be measured by the muscle spindle and the Golgi tendon organs, and can be transmitted to the brain via the peripheral nervous systems (PNS).*

7.3 Numerical results: ADP-based sensorimotor control

7.3.1 Open-loop model of the motor system

We adopted the proposed ADP algorithm to model arm movements in force fields, and to reproduce similar results observed from experiments [22, 41]. For simulation purpose, we used the mathematical model describing two-joint arm movements [108], as shown below.

$$dp = vdt \tag{7.70}$$

$$mdv = (a - bv + f)dt \tag{7.71}$$

$$\tau da = (u - a)dt + d\xi \tag{7.72}$$

where $p = [p_x, p_y]^T$, $v = [v_x, v_y]^T$, $a = [a_x, a_y]^T$, $u = [u_x, u_y]^T$, $f = [f_x, f_y]$ are two-dimensional hand position, velocity, acceleration state, control signal, and external force generated from the field, respectively, m denotes the mass of the hand, b is the viscosity constant, τ is the time constant, $d\xi$ denotes the signal-dependent noise, and is given by

$$d\xi = \begin{bmatrix} c_1 & 0 \\ c_2 & 0 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} d\eta_1 + \begin{bmatrix} 0 & c_2 \\ 0 & c_1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} d\eta_2 \tag{7.73}$$

where η_1 and η_2 are two standard and independent Brownian motions, $c_1 > 0$ and $c_2 > 0$ are constants describing the magnitude of the signal-dependent noise. The values of the parameters are specified in Table 7.1.

It is worth mentioning that this model is similar to many other linear models for describing arm movements; see, for example, [50, 40, 157, 63, 163, 182, 208], to which

Table 7.1: Parameters of the linear model

Parameters	Description	Value	Dimension
m	Hand mass	1.3	kg
b	Viscosity constant	10	N$\cdot$ s/m
τ	Time constant	0.05	s
c_1	Noise magnitude	0.075	
c_2	Noise magnitude	0.025	

our ADP theory is also applicable.

7.3.2 Determining the initial stabilizing control policy

The proposed ADP-based online learning methodology requires an initial stabilizing control policy. To be more specific, we need to find an initial stabilizing feedback gain matrix $K_0 \in \mathbb{R}^{2\times 6}$, such that the closed-loop matrix $A - BK_0$ is Hurwitz. By robust control theory [207], it is possible to find such a matrix K_0 if upper and lower bounds of the elements in both A and B are available and the pair (A, B) is stabilizable. Indeed, these bounds can be estimated by the CNS during the first several trials.

For example, in the absence of disturbances (i.e., $f \equiv 0$), we have

$$A - BK_0 = \begin{bmatrix} 0 & I_2 & 0 \\ 0 & -\frac{b}{m} I_2 & \frac{1}{m} I_2 \\ 0 & 0 & \frac{1}{\tau} I_2 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\tau} I_2 \end{bmatrix} K_0 \tag{7.74}$$

Then, the first several trials in the NF can be interpreted as the exploration of an initial stabilizing feedback gain matrix K_0, by estimating the bounds on the parameters b, m, and τ, and solving a robust control problem. Indeed, if the CNS finds out that $b \in [-8, 12]$, $m \in [1, 1.5]$, and $\tau \in [0.03, 0.07]$ through the first several

trials, the feedback gain matrix K_0 can thus be selected as

$$K_0 = \begin{bmatrix} 100 & 0 & 10 & 0 & 10 & 0 \\ 0 & 100 & 0 & 10 & 0 & 10 \end{bmatrix}. \tag{7.75}$$

Once K_0 is obtained, the CNS can use the proposed ADP method to approximate the optimal control policy.

7.3.3 Selection of the weighting matrices

Here we explain how the weighting matrices are selected in the numerical simulations. First of all, we hypothesize that the selection of Q and R is task-dependent, i.e., the CNS can select different weighting matrices to perform different tasks. For example, if the subject realizes there are disturbance forces along the x-axis, the CNS can assign more weights along that direction to increase the stiffness. This assumption is consistent with the experimental observations by [22].

Therefore, according to the experimental data obtained in a given task, we are able to apply data-fitting-like techniques to find appropriate weighting matrices. However, notice that the search for appropriate symmetric weighting matrices $Q \in \mathbb{R}^{6\times 6}$ and $R \in \mathbb{R}^{2\times 2}$ could be difficult, because they contain as many as 24 independent parameters. To reduce redundancy, we consider three task-dependent parameters $q_x > 0$, $q_y > 0$, and $\theta \in (0, 2\pi]$, which are graphically illustrated in Figure 7.2.

Now, we assume the weighting matrices take the following forms:

$$Q = \begin{bmatrix} T^T Q_0 T & 0 & 0 \\ 0 & 10^{-2} T^T Q_0 T & 0 \\ 0 & 0 & 10^{-4} T^T Q_0 T \end{bmatrix} \tag{7.76}$$

$$R = T^T R_0 T \tag{7.77}$$

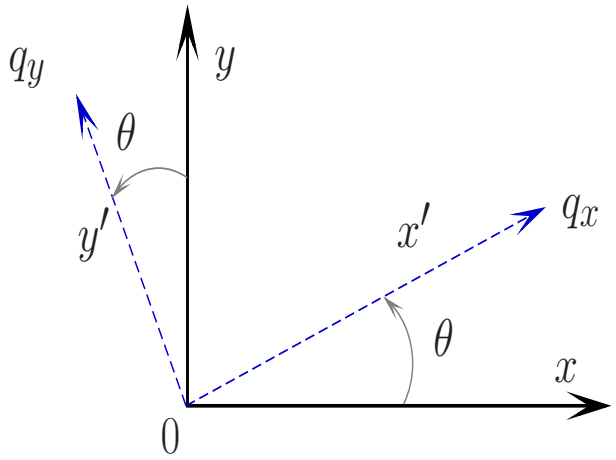

Figure 7.2: Illustration of three weighting factors. The constants $q_x > 0$ and $q_y > 0$ are the weights assigned by the CNS along the x'-axis and the y'-axis, respectively. $\theta \in (0, 2\pi]$ denotes the angular difference between the (x, y) and the (x', y') coordinates.

where

$$T = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \quad Q_0 = \begin{bmatrix} q_x & 0 \\ 0 & q_y \end{bmatrix}, \text{ and } R_0 = I_2. \tag{7.78}$$

with $q_x > 0$, $q_y > 0$, and $\theta \in (0, 2\pi]$ to be determined according to different tasks.

Notice that the ratio between the position weights and the velocity weights is 100, so as the ratio between the velocity weights and the acceleration weights. This ratio is qualitatively consistent with the one used in [162].

7.3.4 Sensorimotor control in a velocity-dependent force field

We used the proposed ADP method to simulate the experiment conducted by [41]. In that experiment, human subjects were seated and asked to move a parallel-link direct drive airmagnet floating manipulandum (PFM) to perform a series of forward arm reaching movements in the horizontal plane. All the subjects performed reaching movements from a start point located $0.25m$ away from the target. The experiment tested human muscle stiffness and motor adaptation in a velocity-dependent force

field (VF). The VF produced a stable interaction with the arm. The force exerted on the hand by the robotic interface in the VF was set to be

$$\begin{bmatrix} f_x \\ f_y \end{bmatrix} = \chi \begin{bmatrix} 13 & -18 \\ 18 & 13 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{7.79}$$

where $\chi \in [2/3, 1]$ is a constant that can be adjusted to the subject's strength. In our simulation, we set $\chi = 0.7$.

Subjects in the experiment [41] first practiced in the null field (NF). Trials were considered successful if they ended inside the target within the prescribed time $0.6 \pm 0.1s$. After enough successful trials were completed, the force field was activated without notifying the subjects. Then, the subjects practiced in the VF until enough successful trials were achieved. After a short break, the subjects then performed several movements in the NF. These trials were called after-effects and were recorded to confirm that adaptation to the force field did occur. More details of the experimental setting can be found in [41, 47].

We first applied the proposed ADP algorithm to simulate the movements in the NF. The simulation started with an initial stabilizing control policy that can be found by the CNS as explained in Section 7.3.2. During each trial, we collected the online data to update the control policy once. After enough trials, an approximate optimal control policy in the NF can be obtained. Also, the stiffness, which is defined as graphical depiction of the elastic restoring force corresponding to the unit displacement of the hand for the subject in the force fields [22], can be numerically computed. In addition, it can be represented in terms of an ellipse by plotting the elastic force produced by a unit displacement [123]. We ran the simulation multiple times with different values of q_x, q_y, and θ. Then, we found that, by setting $q_x = 5 \times 10^4$, $q_y = 10^5$, and $\theta = 15^\circ$, the resultant stiffness ellipse has good consistency with experimental observations [41]. Notice that θ is a positive number in the NF. One possible expla-

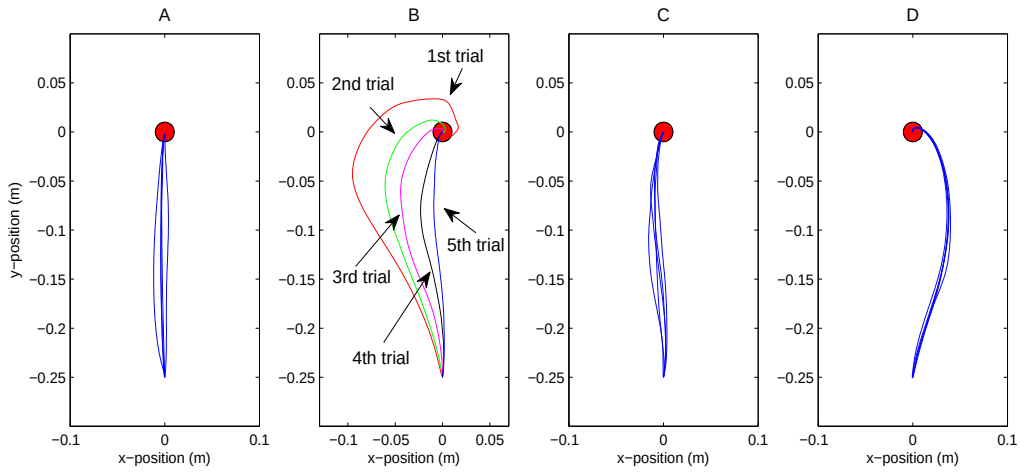

Figure 7.3: Simulated movement trajectories using the proposed learning scheme. **A**, Five successful movement trajectories of one subject in the NF. **B**, The first five consecutive movement trajectories of the subject when exposed to the VF. **C**, Five consecutive movement trajectories of the subject in the VF after 30 trials. **D**, Five independent after-effect trials.

nation is that the subject used the right hand to complete the task, and gave higher weights on the left-hand side.

Then, we proceed with the simulations in the VF. The initial control policy here is the approximate optimal control policy learned in the NF. Once the subject started to realize the existence of the external force after the first trial, it is reasonable for the CNS to modify the weights because larger penalty should be given to the displacement along the direction of the force. It also explains why the selection of the weighting matrices is task-dependent. Indeed, by setting $q_x = 7.5 \times 10^4$, $q_y = 2 \times 10^5$, and $\theta = 60^\circ$ for the VF, we found good consistency with experimental results [41]. The stiffness ellipses are shown in Figure 7.5. One can compare it with the experimental observations in [41].

After 30 trials, the feedback control gain was updated to

$$K_{30} = \begin{bmatrix} 355.52 & 30.31 & 89.83 & -24.25 & 1.67 & -0.24 \\ -198.07 & 322.00 & -5.27 & 95.26 & -0.24 & 1.60 \end{bmatrix}$$

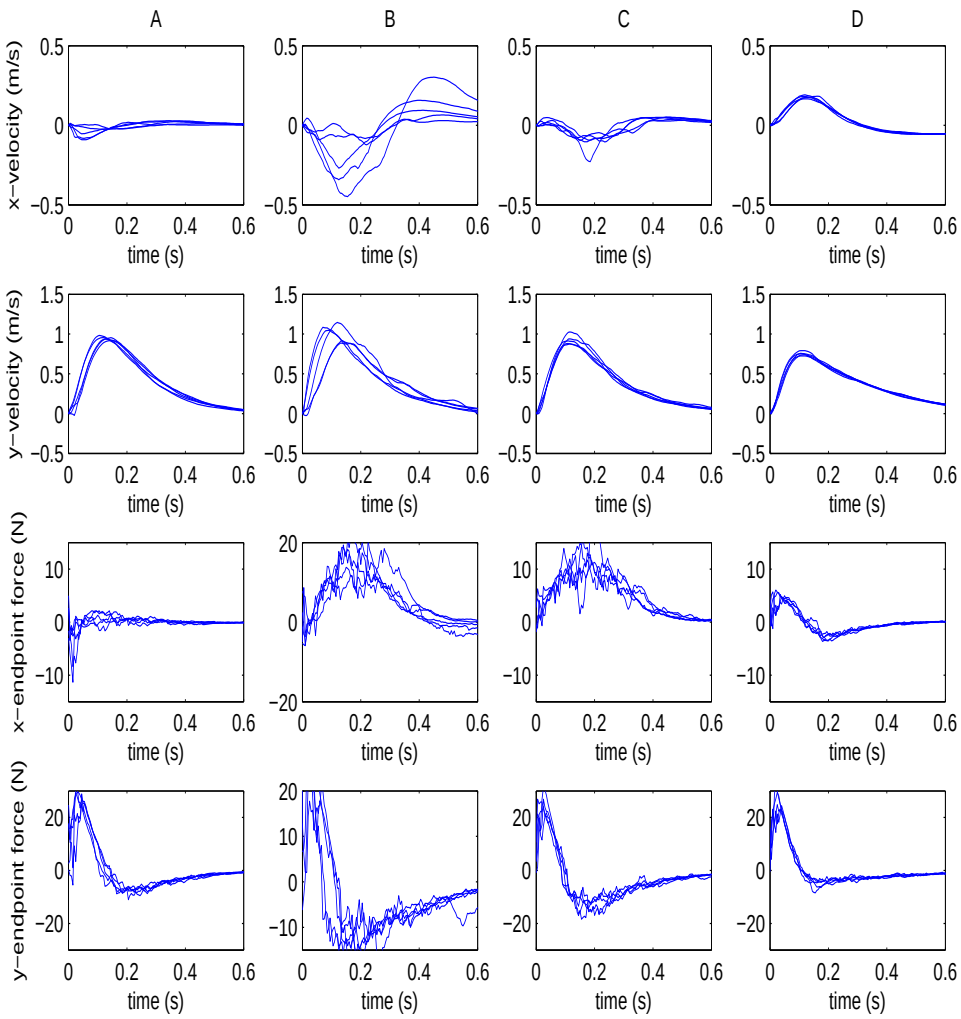

Figure 7.4: Simulated velocity and endpoint force curves show strong consistency with the experimental observations by [41]. **A**, Simulated trajectories of one subject in the NF. **B**, Simulated trajectories of the subject when first exposed into the VF. **C**, Simulated trajectories of the subject in the VF after 30 trials. **D**, After-effect trials. Velocity curves are shown in the first and second rows, in which bell-shaped velocity curves along the y-axis (i.e., the movement direction) are clearly observed. Endpoint force curves are shown in the third and fourth rows. By comparing the first and third figures in the third row, we see subjects adapted to the VF by generating compensation force to counteract the force produced by the field. The shapes of the after-learning endpoint force curves are nearly identical to the experimentally measured endpoint forces reported by [41].

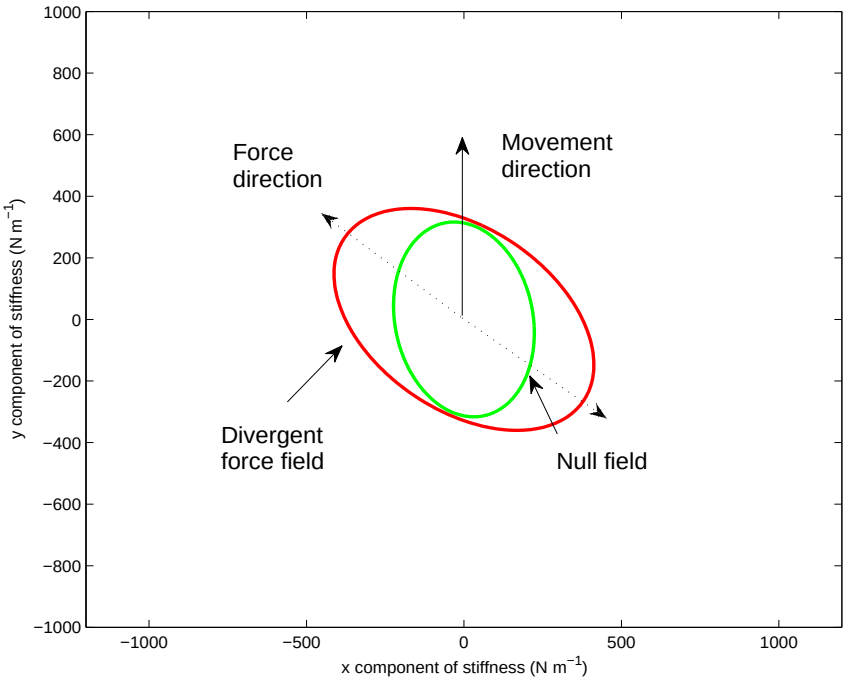

Figure 7.5: Illustration of the stiffness geometry to the VF. The stiffness in the VF (red) increased significantly in the direction of the external force, compared with the stiffness in the NF (green).

For comparison, the optimal feedback gain matrix is provided as follows:

$$K^* = \begin{bmatrix} 358.32 & 30.89 & 90.17 & -24.15 & 1.67 & -0.24 \\ -200.92 & 324.49 & -5.65 & 95.63 & -0.24 & 1.60 \end{bmatrix}.$$

The simulated movement trajectories, the velocity curves, and the endpoint force curves are shown in Figures 7.3 and 7.4. It can be seen that the simulated movement trajectories in the NF are approximately straight lines, and the velocity curves along the y-axis are bell-shaped curves. These simulation results are consistent with experimental observations as well as the curves produced by the previous models [120, 40, 162]. After the subject was exposed to the VF, the first trial was simulated with the same feedback control policy as in the NF. This is because subjects in the experiment were not notified when the external force was activated. Apparently, this control policy was not optimal because the system dynamics was changed and the cost function was also different. Then, the ADP algorithm proceeded. In Figure 7.3, we see the first trial gives a movement trajectory which deviated far away from the straight path but eventually reached the target. Motor adaptation can be observed by comparing the first five consecutive trials. After 30 trials, the movement trajectories return to be straight lines, and the velocity curves become bell-shaped again. It implies that after 30 trials in the VF, the CNS can learn well the optimal control policy using real-time data, without knowing or using the precise system parameters. Finally, our numerical study shows clearly the after-effects of the subject behavior when the VF was suddenly de-activated.

To better illustrate the learning behavior in the VF, we define the movement time t_f of each trial as the time duration from the beginning of the trial until the hand-path enters and remains in the target area. Then, the movement times and distance were calculated and are shown in Figure 7.6.

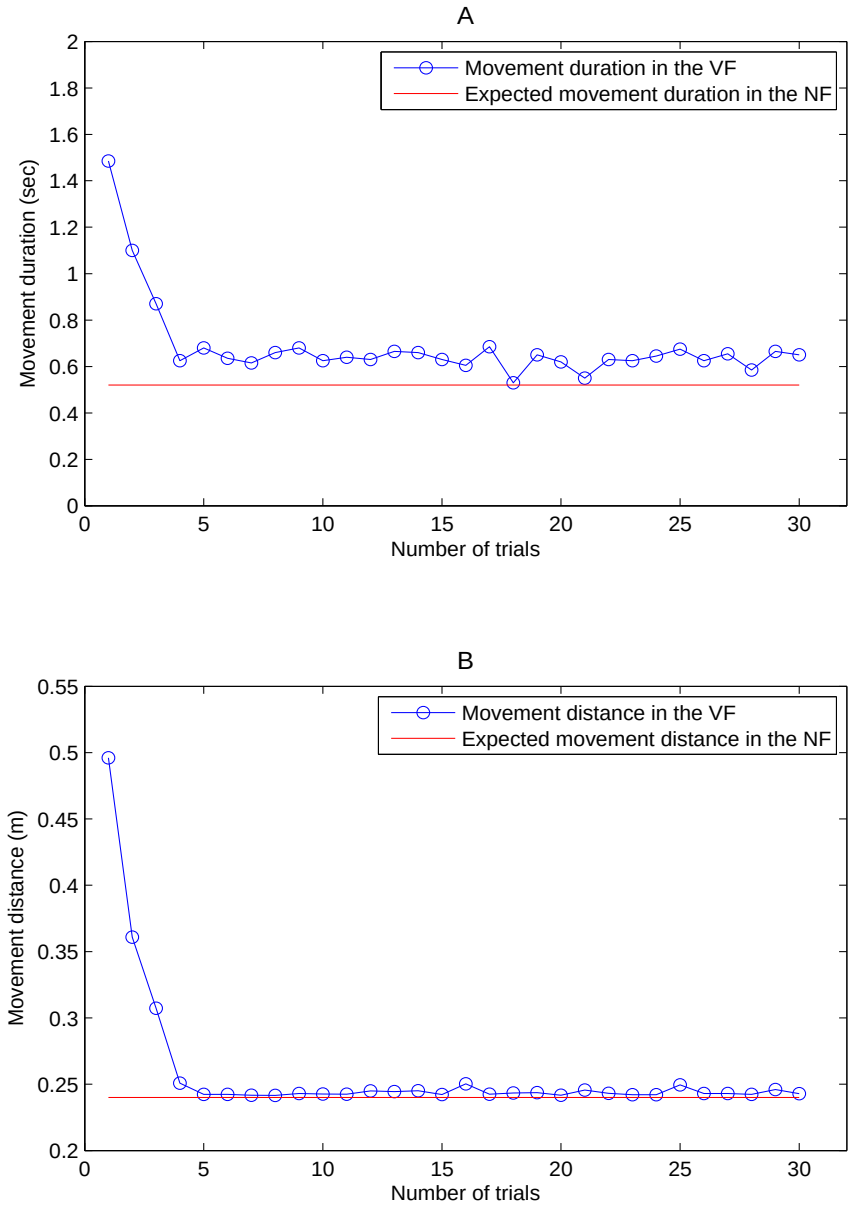

Figure 7.6: **A**, Movement duration as a function of the number of learning trials in the VF. **B**, Movement distance as a function of the number of learning trials in the VF.

7.3.5 Sensorimotor control in a divergent field

Now, let us describe how we simulated the sensorimotor control system in a divergent field (DF) using the proposed ADP theory. In the experiment conducted by [22], the DF produced a negative elastic force perpendicular to the target direction, and was computed as

$$f = \begin{bmatrix} \beta & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ 0 \end{bmatrix} \tag{7.80}$$

where $\beta > 0$ is a sufficiently large constant such that the overall system is unstable. In our simulations, we set $\beta = 150$.

The simulation of the movements before the DF was applied is identical to the one described in the previous subsection, and an approximate optimal control policy in the NF has been obtained. However, this control policy is not stabilizing in the DF, and therefore an initial stabilizing control policy in the DF is needed. To be more specific, we need a matrix $K_0 \in \mathbb{R}^{2\times 6}$ such that

$$A - BK_0 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{\beta}{m} & 0 & -\frac{b}{m} & 0 & \frac{1}{m} & 0 \\ 0 & 0 & 0 & -\frac{b}{m} & 0 & \frac{1}{m} \\ 0 & 0 & 0 & 0 & \frac{1}{\tau} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\tau} \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{\tau} & 0 \\ 0 & \frac{1}{\tau} \end{bmatrix} K_0 \tag{7.81}$$

is Hurwitz.

Therefore, we applied the same control policy learned in the NF to control the movements for the first five trials in the DF. As a result, unstable behaviors were observed in the first several trials (see Figure 7.8 B). Then, a stabilizing feedback

control gain matrix K_0 was assumed available to the CNS, since the CNS has estimated the range of the unknown parameters β, b, m, and τ, and should be able to find K_0 by solving a robust control problem. Here, we increased the first entry in the first row of the matrix $\hat{K}_{nf}$ by 300 and set the resultant matrix to be K_0, which is stabilizing. Then, we applied the proposed ADP algorithm with this K_0 as the initial stabilizing feedback gain matrix. Of course, K_0 can be selected in different ways. Some alternative models describing the learning process from instability to stability can also be found in [42, 160, 199, 208].

To obtain appropriate weighting matrices in the DF, we set $q_x = 1.5 \times 10^5$, $q_y = 10^5$, and $\theta = 15^\circ$. This set of values can give good consistency between our simulation results and the experimental results [22]. Intuitively, we conjecture that the CNS increased the stiffness along the y-axis by assigning more weights to deal with the divergent force. Then, the stiffness ellipses can be numerically shown as in Figure 7.7. One can compare them with the experimental results reported by [22].

It can be seen in Figures 7.8 and 7.9 that the simulated movement trajectories in the NF are approximately straight lines and their velocity curves are bell-shaped. It is easy to notice that the movement trajectories differ slightly from trial to trial. This is due to the motor output variability caused by the signal-dependent noise. When the subject was first exposed to the DF, these variations were further amplified by the DF. As a result, unstable behaviors were observed in the first several trials.

In Figures 7.8 and 7.9, it is clear that, a stabilizing control policy is obtained, the proposed ADP scheme can be applied to generate an approximate optimal control policy. After 30 trials, the hand-path trajectories became approximately straight as in the NF. It implies that the subject has learned to adapt to the dynamics of the

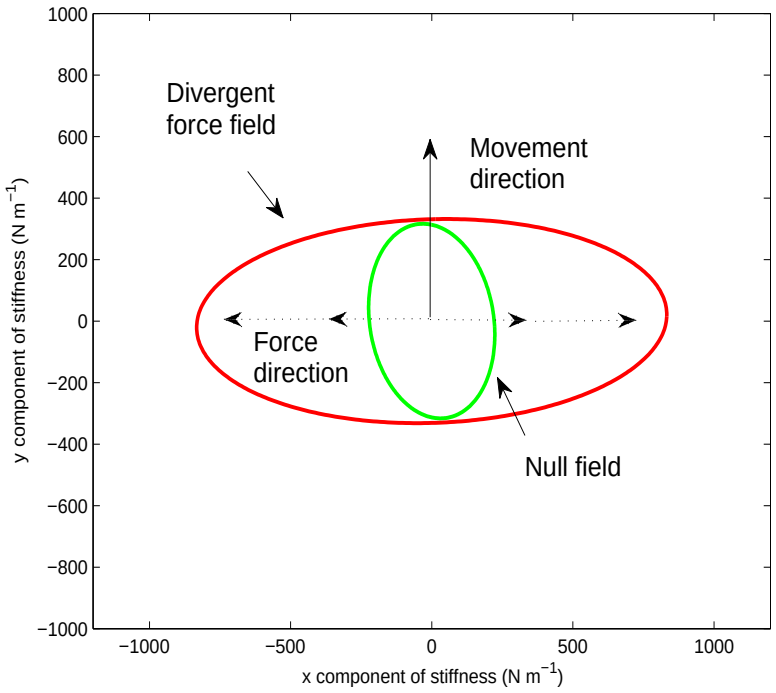

Figure 7.7: Illustration of stiffness geometry to the DF. The stiffness in the DF (red) increased significantly in the direction of the external force, compared with the stiffness in the NF (green).

DF. Indeed, after 30 trials in the DF, the feedback gain matrix has been updated to

$$K_{30} = \begin{bmatrix} 848.95 & 15.73 & 95.60 & 2.67 & 1.69 & 0.05 \\ 24.12 & 319.04 & 2.60 & 62.65 & 0.05 & 1.27 \end{bmatrix}.$$

For comparison purpose, the optimal feedback gain matrix for the ideal case with no noise is shown below:

$$K_{df}^{*} = \begin{bmatrix} 853.67 & 15.96 & 96.07 & 2.70 & 1.70 & 0.05 \\ 24.39 & 321.08 & 2.63 & 62.86 & 0.05 & 1.27 \end{bmatrix}.$$

Finally, we simulated behavior of the subject when the force field is unexpectedly removed. From our simulation results, it is clear to see that the movement trajectories are even straighter than the trajectories in the NF. This is because the CNS has modified the weighting matrices and put more weights on the displacement along the x-axis. As a result, the stiffness ellipses in the NF and DF are apparently different, because the stiffness increased significantly in the direction of the divergence force. The change of stiffness along the movement direction is not significant, as shown in our simulations. These characteristics match well the experimental observations [22, 41].

Again, our simulation results show that the CNS can learn and find an approximate optimal control policy using real-time data, without knowing the precise system parameters.

7.3.6 Fitts's Law

According to [39], the movement duration t_f required to rapidly move to a target area is a function of the distance d to the target and the size of the target s, and a logarithmic law is formulated to represent to the relationship among the three

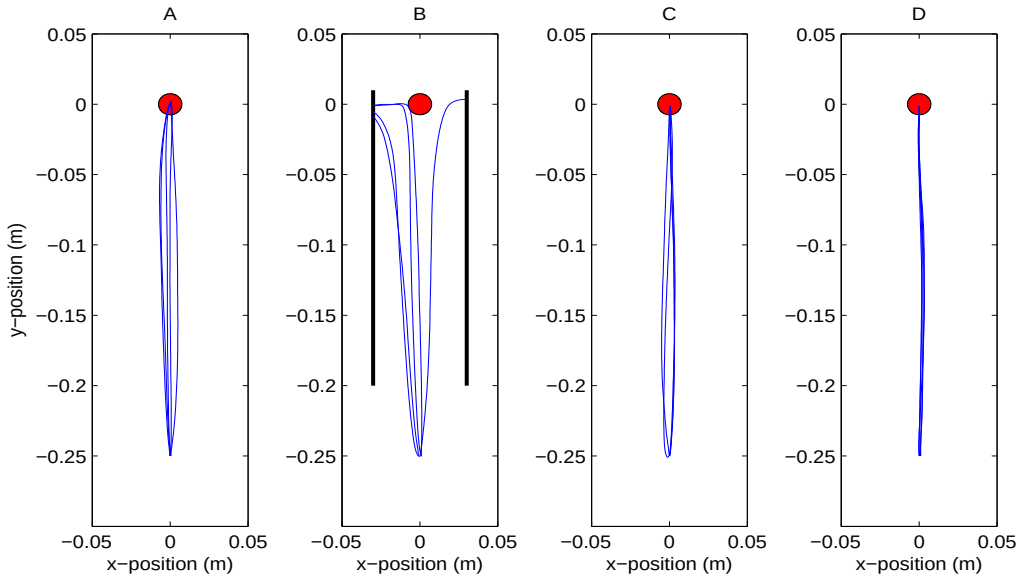

Figure 7.8: Simulated movement trajectories using the proposed learning scheme. **A**, Five successful movement trajectories of one subject in the NF. **B**, Five independent movement trajectories of the subject when exposed to the DF. The black lines on either side of trials in the DF indicate the safety zone, outside of which the force field was turned off. **C**, Five consecutive movement trajectories of the subject in the divergent field after 30 trials. **D**, Five consecutive after-effect trials.

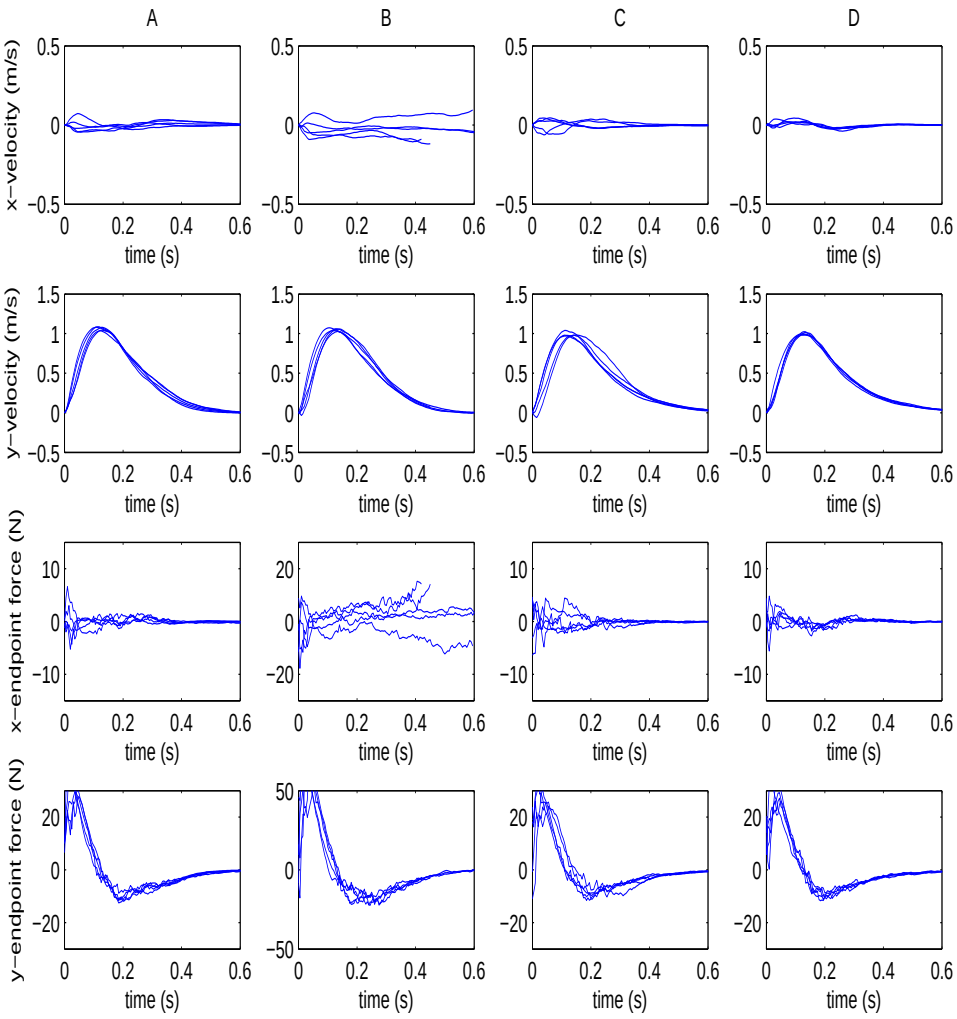

Figure 7.9: Simulated velocity and endpoint force curves using the proposed learning scheme. **A**, Simulated trajectories of one subject in the NF. **B**, Simulated trajectories of the subject when first exposed into the DF. Some trials were terminated earlier than $0.6s$ because they went out of the save zone. **C**, Simulated trajectories of the subject in the divergent force field after 30 trials. **D**, After-effect trials. Velocity curves are shown in the first and second rows, in which bell-shaped velocity curves along the y-axis (i.e., the movement direction) are clearly observed. Endpoint force curves are shown in the third and fourth rows, in which we see subjects adapted to the DF by generating compensation force in the x-direction to counteract the force produced by the DF. In addition, the endpoint force curves are nearly identical to the experimentally measured data [22].

Table 7.2: Data fitting for the log law and power law

Parameters	NF	VF	DF
a (Log law)	0.0840	0.1137	0.0829
b (Log law)	0.0254	-0.0376	0.0197
a (Power law)	0.3401	0.4101	0.3468
b (Power law)	-1.7618	-1.7796	-1.8048

variables t_f, d, and s as follows

$$t_f = a + b \log_2 \left(\frac{d}{s} \right) \tag{7.82}$$

where a and b are two constants. In 1998, [138] proposed the following power law:

$$t_f = a \left(\frac{d}{s} \right)^b. \tag{7.83}$$

Here we validated our model by using both the log law and the power law. The target size s is defined as its diameter, and the distance is fixed as $d = 0.24m$. We simulated the movement times from the trials in the NF, the after-learning trials in the VF and DF. The data fitting results are shown in Figure 7.10 and Table 7.2. It can be seen that our simulation results are consistent with Fitts's law predictions.

7.4 Numerical results: RADP-based sensorimotor control

In this section, we apply the proposed RADP algorithm to model arm movements in a divergent force field, and arm movements in a velocity-dependent force field. However, different from the previous section, we assume the mechanical device generating the forces was subject to certain-time delay. Therefore, the dynamics of the mechanical device is treated as the dynamic uncertainty. We will also compare our simulation

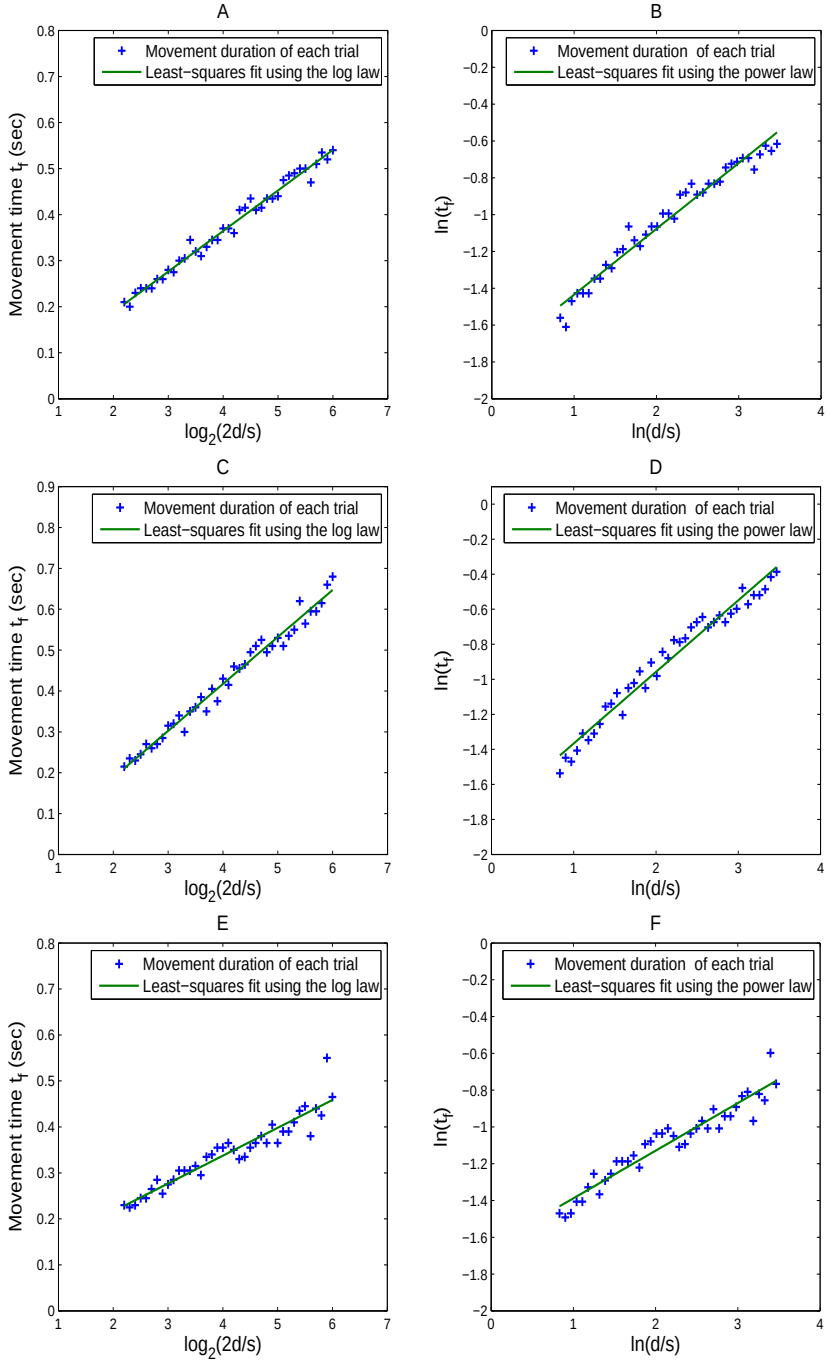

Figure 7.10: Log and power forms of Fitts's law. Crosses in the first row, the second row, and the third row represent movement times simulated in the NF, the VF, and the DF, respectively. Solid lines in **A, C, E** are least-squares fits using the log Fitts's law, and solid lines in **B, D, F** are the least-squares fits using the power Fitts's law.

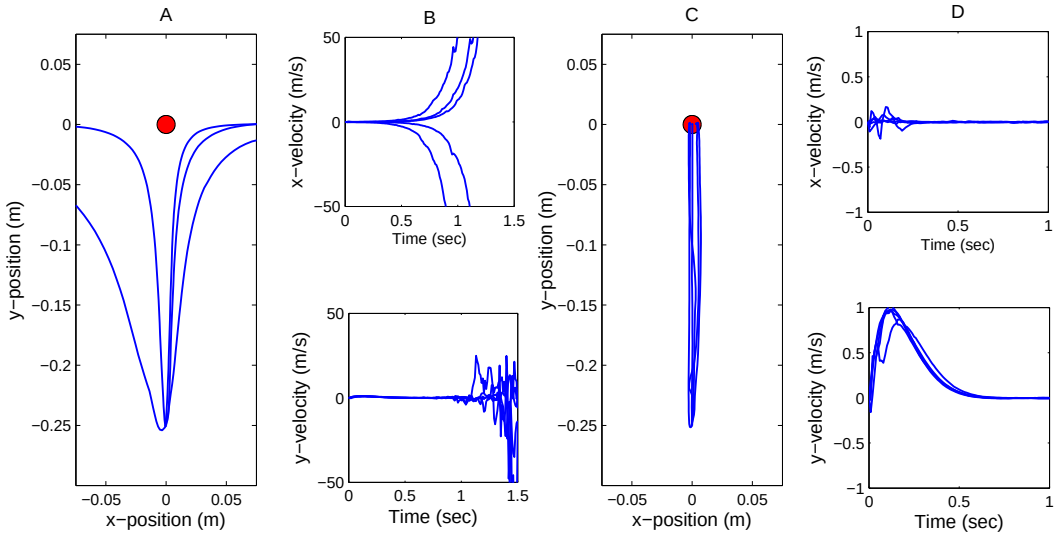

Figure 7.11: Simulations of hand trajectories in the divergent force field using the proposed RADP based learning scheme. Movements originate at $(0, -0.25m)$ and the target is located at $(0, 0)$. **A.** Simulated hand trajectories during initial exposure to the force field. **B.** Simulated hand trajectories after 30 trials. **C.** Simulated speed curves before learning. **D.** Simulated speed curves after learning.

results with experimental observations [22, 41, 142].

We adopt the proposed RADP algorithm to model arm movements in force fields, and to reproduce similar results observed from experiments [22, 41]. The mathematical model for the motor system is the same as (7.70)-(7.72), with the parameters given in Table 7.1.

7.4.1 Divergent force field with time-delay

Let us describe how we simulated the sensorimotor control system in a divergent field (DF) using the proposed RADP theory. To generate such a force field, we consider the following system.

$$d\begin{bmatrix} f_x \\ f_y \end{bmatrix} = -\begin{bmatrix} \tau_f & 0 \\ 0 & \tau_f \end{bmatrix}^{-1} \left(\begin{bmatrix} f_x \\ f_y \end{bmatrix} - \begin{bmatrix} \beta & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} \right) dt \tag{7.84}$$

where $\beta > 0$ is a sufficiently large constant such that the closed system is unstable with the initial control policy. System (7.89) is suitable to model the dynamics of divergent force field used by [22, 41] with the time constant $\tau_f > 0$ describing the time-delay of the physical device which generated the force field.

In our simulation, the subject first practiced in the null field (NF) with an initial control policy obtained as described in Remark 7.2.4. Then, we simulated the scenario that the CNS implement the proposed RADP method with respect to the following weighting matrices

$$\begin{aligned} Q_1 &= \begin{bmatrix} 500 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}, \quad Q_2 = I_2, \\ R_1 &= R_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}. \end{aligned} \tag{7.85}$$

The force field was simulated with $\beta = 230$. After enough trials, we turned on the divergent force. Then, the overall system became unstable.

After the CNS became aware of the divergent force, the RADP algorithm was applied to update the control policy using the online sensory data to find a new robust optimal control policy with respect to the following cost

$$\begin{aligned} Q_1 &= \begin{bmatrix} 10^4 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_2 = I_2, \\ R_1 &= R_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}. \end{aligned} \tag{7.86}$$

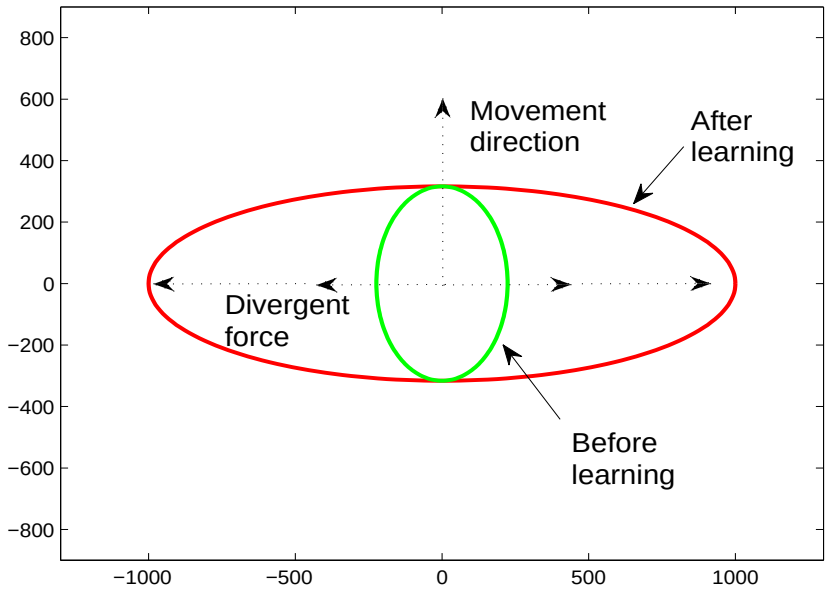

Figure 7.12: Adaptation of stiffness geometry to the force field. The after-learning stiffness (red) increased significantly in the direction of the external force, compared with the initial stiffness (green).

The before-learning and after-learning movement trajectories are shown in Figure 7.11. It can be seen that, stability can be recovered after applying the RADP-based learning schemes, and the hand-path trajectories became approximately straight. It implies that the subject has learned adapting to the dynamics of the DF. It is also worth pointing out that the learning algorithm does not require the precise dynamics of the sensorimotor system or the dynamic uncertainty.

Our model suggests a time-invariant control policy. Consequently, it is straightforward to study the change of muscle stiffness before and after learning. Indeed, the compensation force caused by displacement after-learning can be calculated as:

$$\Delta f_{df} = -\begin{bmatrix} 1000 & 0 \\ 0 & 316.2278 \end{bmatrix} \begin{bmatrix} \Delta p_x \\ \Delta p_y \end{bmatrix} \tag{7.87}$$

In contrast, before the learning starts, the compensation force was

$$\Delta f_{nf} = -\begin{bmatrix} 223.6068 & 0 \\ 0 & 316.2278 \end{bmatrix} \begin{bmatrix} \Delta p_x \\ \Delta p_y \end{bmatrix} \tag{7.88}$$

Now, the stiffness, defined as a graphical depiction of the elastic restoring force corresponding to the unit displacement of the hand for the subject in the force fields [22, 41], can be numerically computed. In addition, it can be presented in terms of an ellipse by plotting the elastic force produced by a unit displacement [123]. In Figure 7.12, we plotted the stiffness ellipses. It can be seen that the difference between the stiffness ellipses before and after learning is a direct result from the changes in both the weighting matrices and the system dynamics.

7.4.2 Velocity-dependent force field with time delay

Now, we use the proposed RADP method to simulate the experiment conducted by [142]. We model the velocity-dependent force field using the following dynamic system

$$d\begin{bmatrix} f_x \\ f_y \end{bmatrix} = -\begin{bmatrix} \tau_f & 0 \\ 0 & \tau_f \end{bmatrix}^{-1} \times \left(\begin{bmatrix} f_x \\ f_y \end{bmatrix} - \begin{bmatrix} -10.1 & -11.2 \\ -11.2 & 11.1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) dt \tag{7.89}$$

In the experiment [142], each subject was asked to move a cursor from the center of a workspace to a target at an angle randomly chosen from the set $\{0°, 45°, \cdots, 315°\}$, and at a distance of $0.1m$. After one target was reached, the next target, randomly selected, was presented.

There were four different stages in the experiment. First, the subject made arm movements without the force field. Second, the force field was applied without notifying the subject. Third, the subject adapted to the force field. Finally, the force field was suddenly removed and the after-learning effects were observed.

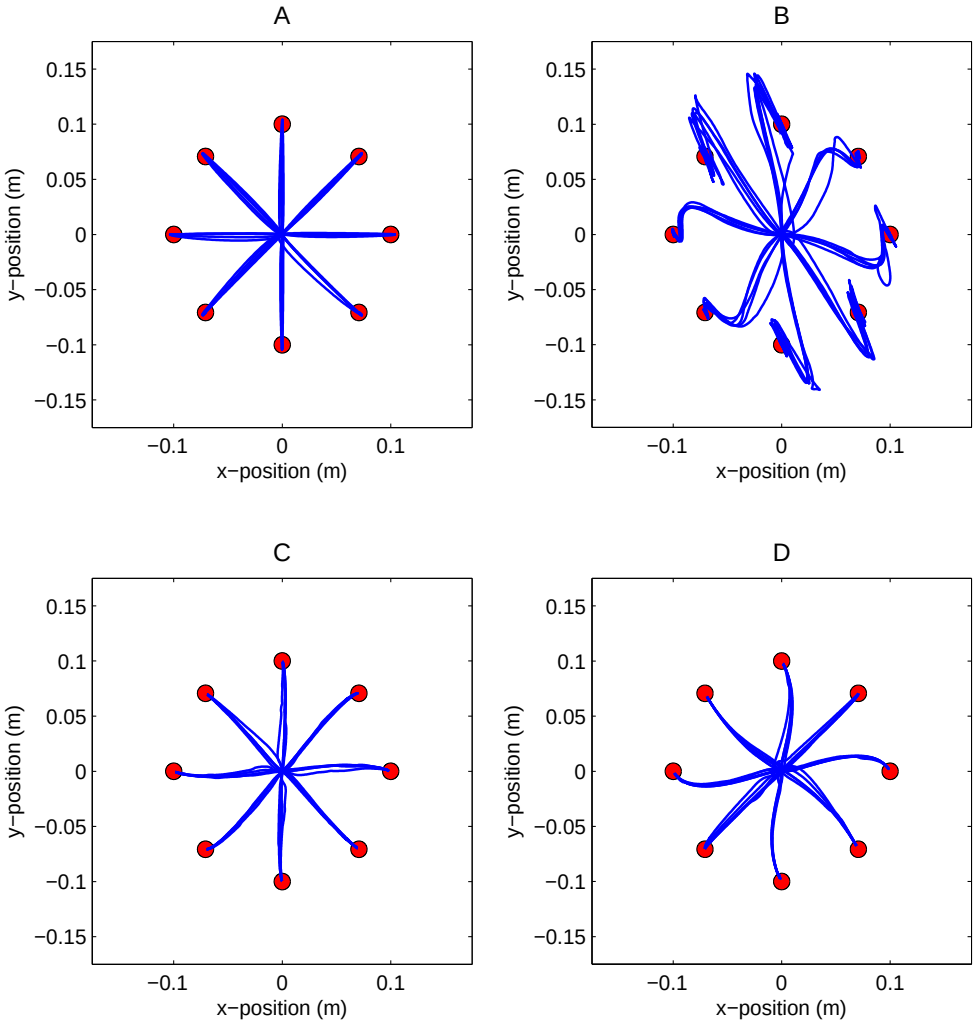

Figure 7.13: Simulation of hand trajectories in the velocity-dependent force field using the proposed RADP based learning scheme. Movements originate at the center. **A.** Simulation of hand trajectories in the null force field. **B.** Simulated Performance during initial exposure to the force field. **C.** Simulated hand trajectories in the force field after 30 trials. **D.** Simulated aftereffects of adaptation to the force field.

Before the force field was activated, the initial control policy we assumed was the same as the one in the previous simulation. Once the CNS noticed the velocity-dependent field, the new weighting matrices was replaced by

$$\begin{aligned} Q_1 &= \begin{bmatrix} 10^4 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_2 = I_2, \\ R_1 &= R_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}. \end{aligned} \tag{7.90}$$

The movement performance in the four different stages were simulated using the proposed RADP algorithm and the results are given in Figure 7.13. In addition, we plotted the velocity curves of the arm movement during the first three stages as shown in Figure. Interestingly, consistency can be found by comparing Figure 7.14 in this paper with Figure 10 in [142].

7.5 Discussion

7.5.1 Non-model-based learning

Most of the previous models for sensorimotor control have concluded that the CNS knows precisely the knowledge of the motor system and its interacting environment [33, 40, 50, 77, 131, 140, 163, 162, 165]. The computation of optimal control laws is based on this assumption. By contrast, the proposed ADP methodology is a non-model-based approach and informs that the optimal control policy is derived using the real-time sensory data and is robust to dynamic uncertainties such as signal-dependent noise. In the absence of external disturbances, our model can generate

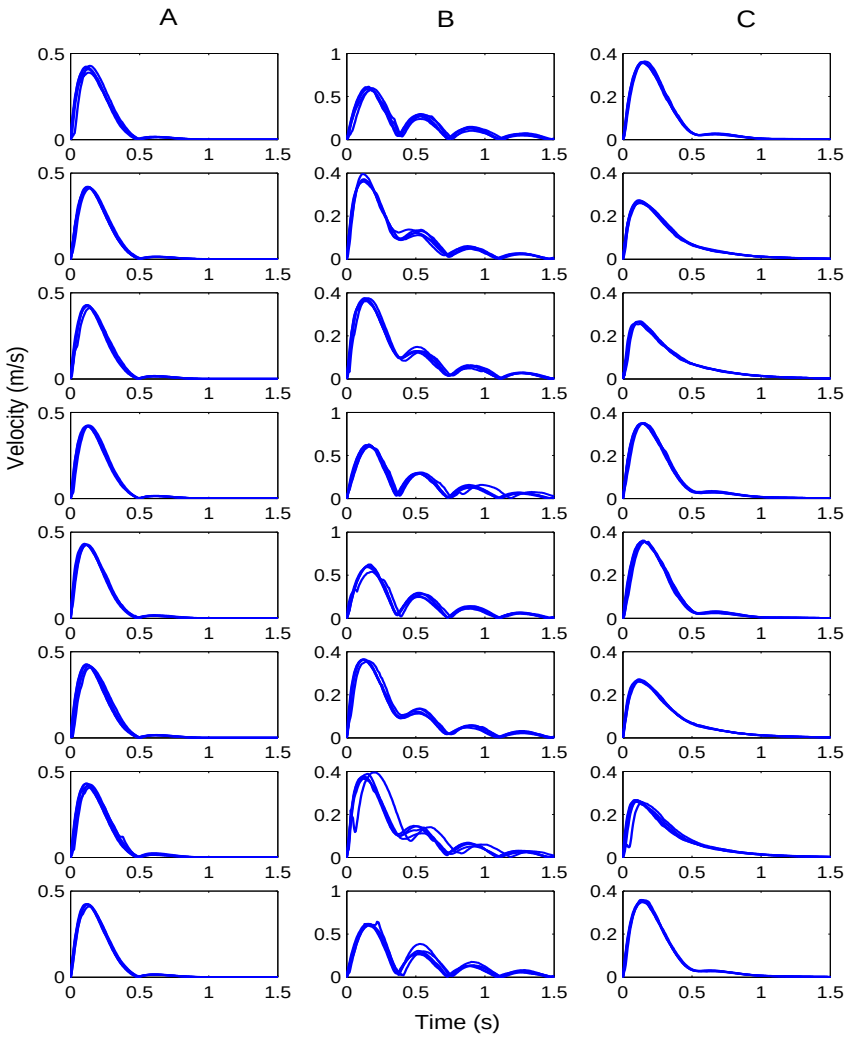

Figure 7.14: Hand velocities before and after adaptation to the force field. The curves, from the first row to the eight row, are for targets at 0°, 45°, $\cdots$, 315°. **A,** hand velocities in a null field before exposure to the force field. **B**, hand velocities upon initial exposure to the force field. **C**, hand velocities after adaptation to the force field.

typically observed position, velocity, and endpoint force curves as produced by the previous models. As one of the key differences with existing sensorimotor models, our proposed computational mechanism suggests that, when confronted with unknown environments and imprecise dynamics, the CNS may update and improve its command signals for movement through learning and repeated trials.

In the presence of perturbations, most of the previous models have relied on sensory prediction errors to form an estimate of the perturbation [9, 92, 195]. However, this viewpoint is difficult to be justified theoretically and has not been convincingly validated by experiments. Indeed, evidence against this source-identified adaptation is reported by [59], where a self-generated perturbation was created but it was not identified or used in formulating the control policy. This is consistent with the learning scheme we proposed in this paper. Indeed, our ADP-based learning scheme does not identify the dynamics of the force fields. Instead, optimal control policies in the presence of force fields are directly obtained through successive approximations. By simulating the experiments conducted by [22] and [41], we have found that our computational results match well with the experimental results. In particular, our simulation results show gradual adaptation to the unknown force fields, with nearly identical movement trajectories in the first several consecutive trials reported in the experiment [41]. The simulated post-learning velocity and endpoint force curves fit well with the experimental observations [22, 41]. Our simulations clearly demonstrated the after-effects phenomenon.

7.5.2 Stability and convergence properties

Several reinforcement-learning-based models for motor adaptation have been developed in the past literature [35, 64]. However, it is not easy to analyze the convergence and the stability properties of the learning schemes. In this paper, we have extended the ADP theory to continuous-time linear systems with signal-dependent noise, and

applied this theory to model sensorimotor control. An added value of the proposed ADP methodology is that rigorous convergence and stability analysis is given and is surprisingly simple by means of linear optimal control theory.

7.5.3 Muscle stiffness

Several practical methods for experimentally measuring stiffness have been proposed in the past literature [21, 47, 54], and the changes of stiffness in force fields were reported by [22] and [41]. However, how the CNS modifies the stiffness geometry and achieves optimal motor behavior remains a largely open question. [22] suggested that the CNS minimizes the hand-path error relative to a straight line joining the start position and the target center. This optimization problem does not involve any system dynamics, and cannot link the modification of stiffness to optimal feedback control theory. On the other hand, the stiffness may not be well analyzed using other models based on finite-horizon optimal control theory (see, for example, [162, 50]). This is because those models use time-varying control policies, leading to continuous change of the stiffness during the movement.

In the ADP-based model, time-invariant control policies are computed, and it is comparably easy to analyze the muscle stiffness by studying the position feedback gains. Our modeling methodology implies that the change of stiffness results from the modification of the weighing matrices by the CNS and the change of the system dynamics. Indeed, our simulation results provide similar stiffness ellipses as those measured in experiments [22, 41]. In addition, our model suggests that different stiffness geometries of different individuals may be a consequence of different weighting matrices they selected. Therefore, compared with other models of motor control and motor adaptation, our modeling strategy can explain naturally the change of stiffness observed in experiments from the viewpoint of optimal feedback control [99].

7.5.4 Data fitting for the weighting matrices

The weighting matrices we used in the numerical simulation are selected such that our resultant computational results can have qualitative consistency with experimental results [22, 41]. If accurate human motor data become available, better fits for the weights can be obtained using a two-loop optimization approach [65]. The inner-loop uses the proposed ADP method to approximate an optimal control policy and generate the stiffness ellipses. The outer-loop compares the error between the simulated stiffness ellipses with experimental observations and adjusts the parameters q_x, q_y, and θ to minimize the error.

7.5.5 Infinite-horizon optimal control

During each trial, a time-invariant control policy is suggested in our methodology. The time-invariant control policy has a main advantage that movement duration does not need to be pre-fixed by the CNS. This seems more realistic because the duration of each movement is different from each other due to the signal-dependent noise and is difficult to be pre-fixed. [163] suggested that if the target is not reached at the predicted reaching time, the CNS can similarly plan an independent new trajectory between the actual position of the hand and the final target, and the final trajectory will be the superposition of all the trajectories. By contrast, our model matches the intuitive notion that the motor system keeps moving the hand toward the target until it is reached, and much less computational burden is required. Our simulation results match well with Fitts's law predictions. In addition, this type of control policies can also be used to analytically derive the Fitts's law as illustrated by [131].

7.5.6 Comparison with iterative learning control

Iterative learning control (ILC) is an open-loop control scheme that iteratively learns a feedforward signal based on the tracking error of the previous trials [20]. It has also been used to model motor learning [208]. Compared with ILC, the ADP-based learning method has at least three advantages. First, conventional ILC method uses open-loop control policy for each individual trial. Hence, it may not explain the feedback control mechanism involved in each individual movement, which is essential in generating bell-shaped velocity-curves. Second, ILC assumes the external disturbance is iteration invariant. However, motor uncertainties do not satisfy this assumption, since no two movements are exactly the same. Third, in the ILC model, learning only happens among different trials. Therefore, it cannot model the online learning during a single trial. On the other hand, this online learning process can be modeled by our ADP method, since the ADP scheme gives the flexibility to specify the amount of time that is needed for collecting online data and updating the control policy. Indeed, if the time duration is set to be less than the time needed for one trial, then online learning during one movement can be simulated.

7.5.7 Connection between optimality and robustness

Although optimal control theory is the dominant paradigm for understanding motor behavior, and optimization based models can explain many aspects of sensor motor control, it is not clear if the CNS keeps using the optimal control policy when dynamic uncertainty occurs. Experimental results obtained by [59] show that the control scheme employed by the CNS in the presence of external disturbances may only be sub-optimal because the control scheme they observed experimentally is not energy efficient. Indeed, in the presence of dynamic uncertainties, guaranteeing optimality and stability becomes a nontrivial task.

The RADP studies the stability of interconnected systems by analyzing the gain

conditions (7.32), (7.33), and (7.42)-(7.44), which are inspired by a simplified version of the nonlinear small-gain theorem [83]. As shown previously, these conditions can be satisfied by choosing suitable cost functions with appropriately designed weighting matrices [69]. It should be mentioned that the control policy generated from our RADP method is optimal for the nominal/reduced system, and remains stable and retains suboptimality in the presence of dynamic uncertainty. The change of stiffness in the divergent force field was reported by [22]. However, they assumed that the stiffness was modified to minimize some cost function associated with the minimum-jerk model [40] which does not involve optimal feedback control theory. Alternatively, in the RADP theory, the change of stiffness can be interpreted as a direct result of the change of weighting matrices.

Hence, the proposed RADP theory is compatible with the optimal control theory and the experimental results observed in the past literature. More importantly, it provides a unified framework that naturally connects optimality and robustness to explain the motor behavior with/without uncertainties.

7.6 Conclusions

We have developed ADP and RADP methods for linear stochastic systems with signal-dependent noise with an objective to model goal-oriented sensorimotor control systems. An appealing feature of this new computational mechanism of sensorimotor control is that the CNS does not rely upon the a priori knowledge of systems dynamics and the environment to generate a command signal for hand movement. Our theory can explain the change of stiffness geometry from a perspective of adaptive optimal feedback control versus nonadaptive optimal control theory [131, 163, 162]. In particular, the RADP theory not only gives computational results which are compatible with experimental data [22, 41, 142], but also provides a unified framework to study

robustness and optimality simultaneously. We therefore argue that the human motor system may use ADP and RADP-like mechanisms to control movements.

Chapter 8

Conclusions and future work

8.1 Conclusions

This dissertation has proposed our recent contributions to the new framework of RADP and illustrated its potential applications. The major contributions of this dissertation can be summarized from two aspects.

On one hand, developing ADP-based methodology for the continuous-time (CT) setting with completely unknown dynamics is a challenging topic. Indeed, although the action-dependent heuristic dynamic programming [189] (or Q-learning [181]) does not depend on the discrete-time (DT) system, it cannot be directly applied for solving CT problems, such as the problems addressed in this dissertation. One major reason is that the structures of CT algebraic Riccati equations (AREs) or the Hamilton-Jacobi-Bellman (HJB) equations are significantly different from their DT counterparts. This dissertation has introduced a novel computational policy iteration approach we developed recently [68]. It finds online adaptive optimal controllers for CT linear systems with completely unknown system dynamics, and solves the ARE iteratively using system state and input information collected online, without knowing the system matrices. This objective is achieved by taking advantages of the exploration noise. This

methodology has been further extended for affine nonlinear systems. An immediate result has been obtained by using neural networks (NNs) to approximate online the cost function and the new control policy [75]. We have provided rigorous proofs on the convergence property of this method. However, NNs-based approximation can only be effective on some compact set, and it is generally not trivial to determine the type of basis functions to achieve good approximation performance. Therefore, we have proposed for the first time the idea of global adaptive dynamic programming (GADP), which finds a suboptimal control policy but provides global stability and gives rise to computational efficiency.

On the other hand, in the past literature of ADP, it is commonly assumed that both the system order is known and the state variables are perfectly measurable. These two conditions are generally restrictive and are needed to be relaxed in further research. In addition, it is widely recognized that, besides maximizing the reward from the environment, biological systems learn to achieve enhanced robustness (or greater chance of survival) through interacting with the unknown environment, and they may only be able to make decisions based on *partial-state* information due to the complexity of the real-world environment. This dissertation bridges the gap in the past literature of ADP where dynamic uncertainties or unmodeled dynamics were not addressed. In this dissertation, two strategies have been introduced to achieve robust stabilization in the presence of dynamic uncertainties. First, for partially linear systems [75] and weakly nonlinear large-scale systems [70], we have derived conditions on the weighting matrices for the performance indices for each nominal system. These conditions are in spirit of the small-gain theorem [83]. Second, for affine nonlinear systems which interact with nonlinear dynamic uncertainties, we have employed the robust redesign technique [83, 129] and the Lyapunov-based small-gain theorem [81]. This methodology has been used to redesign the approximate optimal control policy obtained using the NN-based approximation to achieve robust optimal

stabilization [75]. Also, it has been employed to redesign the suboptimal control policy obtained using the SOS-based policy iteration, such that global robust and suboptimal stabilization can be achieved [71].

We have applied the proposed online learning methodologies for engineering applications. In particular, the RADP method finds good applications in power systems related control problems. Simulations for a two-machine and a ten-machine power systems have been been provided to show the efficiency and effectiveness of the proposed algorithms. Further, since RADP shares many essential features with reinforcement learning, we found it is reasonable to use RADP as a reverse-engineering approach to model human motor control and learning. To this end, we have developed RADP-based learning methods to study stochastic systems with control-dependent noise, and use these methods to numerically reproduce experimental results obtained in [22, 41, 142]. Observing the strong consistency of numerical results with experimental data, we argue that the central nervous system (CNS) may use RADP-like mechanisms to coordinate movements in the presence of static and/or dynamic uncertainties.

8.2 Future work

A gap has been bridged by the development of the RADP framework, in which dynamic uncertainty is taken into account. However, compared with a true brain-like intelligent learning control system, results presented in this dissertation are only the tips of an iceberg. Quite a few interesting and exciting related topics deserve further research. For example, the following are some of the directions for continuation of this work.

1. Extending the current results on GADP for more generalized nonlinear systems, such as nonaffine nonlinear systems [14].

2. Gaining more insights into understanding the rank condition and its relations with the exploration noise.

3. Developing output-feedback-based RADP theory (see [46] for some preliminary results).

4. Extending the current ADP and RADP methodologies for nonlinear stochastic systems, and apply them to more realistic models to study human motor learning (see [14, 15] for some preliminary results).

5. Enriching the features of ADP and RADP theories by incorporating more biologically-inspired learning behaviors.

6. Developing RADP-based tracking methods to achieve online tracking with both static and dynamic uncertainties.

7. Quantifying the sub-optimality in the presence of dynamic uncertainties.

Finally, the author believes that the RADP theory developed in this thesis has numerous potential applications in practical engineering systems and reverse-engineering problems. In addition to further developing and refining the theory of RADP, it is also important to study the practical implementation issues for real-world applications.

Chapter 9

Appendices

9.1 Review of optimal control theory

9.1.1 Linear quadratic regulator (LQR) for CT linear systems

Consider a CT linear system described by

$$\dot{x} = Ax + Bu \tag{9.1}$$

where $x \in \mathbb{R}^n$ is the system state fully available for feedback control design; $u \in \mathbb{R}^m$ is the control input; $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$ are unknown constant matrices. In addition, the system is assumed to be stabilizable.

The design objective is to find a linear optimal control law in the form of

$$u = -Kx \tag{9.2}$$

which minimizes the following performance index

$$J = \int_0^\infty (x^T Q x + u^T R u) dt, \quad x(0) = x_0 \in \mathbb{R}^n, \tag{9.3}$$

where $Q = Q^T \geq 0$, $R = R^T > 0$, with $(A, Q^{1/2})$ observable.

By linear optimal control theory [103], when both A and B are accurately known, solution to this problem can be found by solving the following well-known algebraic Riccati equation (ARE)

$$A^T P + PA + Q - PBR^{-1}B^T P = 0. \tag{9.4}$$

If the pair (A, B) is stabilizable and the pair $(A, Q^{1/2})$ is observable, (9.4) has a unique symmetric positive definite solution P^*. The optimal feedback gain matrix K^* in (9.2) can thus be determined by

$$K^* = R^{-1}B^T P^*. \tag{9.5}$$

Notice that the optimal feedback gain matrix K^* does not depend on the initial condition x_0.

9.1.2 Nonlinear optimal control for CT affine nonlinear systems

Consider the nonlinear system

$$\dot{x} = f(x) + g(x)u \tag{9.6}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the control input, $f(x)$ and $g(x)$ are locally Lipschitz functions with $f(0) = 0$.

In classical nonlinear optimal control theory [103], the common objective is to find a control policy u that minimizes certain performance, which takes, for example, the following form.

$$J(x_0, u) = \int_0^\infty r(x(t), u(t))dt, \quad x(0) = x_0 \tag{9.7}$$

where $r(x, u) = Q(x) + u^T Ru$, with $Q(x)$ a positive definite function, and R is a symmetric positive definite matrix. Notice that, the purpose of specifying $r(x, u)$ in this form is to guarantee that an optimal control policy can be explicitly determined, if it exists.

Now, suppose there exists $V^o \in \mathcal{P}$, such that the Hamilton-Jacobi-Bellman (HJB) equation holds

$$\mathcal{H}(V^o) = 0 \tag{9.8}$$

where

$$\mathcal{H}(V) \;=\; \nabla V^T(x) f(x) + Q(x) - \frac{1}{4}\nabla V^T(x) g(x) R^{-1} g^T(x) \nabla V(x).$$

Then, it is easy to see that V^o is a well-defined Lyapunov function for the closed-loop system comprised of (9.6) and

$$u^o(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla V^o(x). \tag{9.9}$$

Hence, this closed-loop system is globally asymptotically stable at $x = 0$ [86]. Then, according to [141, Theorem 3.19], u^o is the optimal control policy, and the value function $V^o(x_0)$ gives the optimal cost at the initial condition $x(0) = x_0$, i.e.,

$$V^o(x_0) = \min_u J(x_0, u) = J(x_0, u^o), \quad \forall x_0 \in \mathbb{R}^n. \tag{9.10}$$

It can also be shown that V^o is the unique solution to the HJB equation (9.8) with $V^o \in \mathcal{P}$. Indeed, let $\hat{V} \in \mathcal{P}$ be another solution to (9.8). Then, by Theorem 3.19 in [141], along the solutions of the closed-loop system composed of (9.6) and $u = \hat{u} = -\frac{1}{2}R^{-1}g^T\nabla\hat{V}$, it follows that

$$\hat{V}(x_0) \;=\; V^o(x_0) - \int_0^\infty |u^o - \hat{u}|_R^2 dt, \qquad \forall x_0 \in \mathbb{R}^n. \tag{9.11}$$

Finally, comparing (9.10) and (9.11), we conclude that $V^o = \hat{V}$.

9.2 Review of ISS and the nonlinear small-gain theorem

Here we review some important tools from modern nonlinear control; see, for instance, [61, 83, 80, 86, 114, 150], and references therein for the details. See [85] for more recent developments in nonlinear systems and control.

Consider the system

$$\dot{x} = f(x, u) \tag{9.12}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the input, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is locally Lipschitz.

Definition 9.2.1 ([148, 149])**.** *The system* (9.12) *is said to be input-to-state stable (ISS) with gain γ if, for any measurable essentially bounded input u and any initial condition $x(0)$, the solution $x(t)$ exists for every $t \geq 0$ and satisfies*

$$|x(t)| \leq \beta(|x(0)|, t) + \gamma(\|u\|) \tag{9.13}$$

where β is of class $\mathcal{KL}$ and γ is of class $\mathcal{K}$.

Definition 9.2.2 ([151]). *A continuously differentiable function V is said to be an ISS-Lyapunov function for the system* (9.12) *if V is positive definite and proper, and satisfies the following implication:*

$$|x| \geq \chi(|u|) \Rightarrow \nabla V(x)^T f(x,u) \leq -\kappa(|x|) \tag{9.14}$$

where κ is positive definite and χ is of class $\mathcal{K}$.

Next, consider an interconnected system described by

$$\dot{x}_1 = f_1(x_1, x_2, v), \tag{9.15}$$

$$\dot{x}_2 = f_2(x_1, x_2, v) \tag{9.16}$$

where, for $i = 1, 2$, $x_i \in \mathbb{R}^{n_i}$, $v \in \mathbb{R}^{n_v}$, $f_i : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_i}$ is locally Lipschitz.

Assumption 9.2.1. *For each $i = 1, 2$, there exists an ISS-Lyapunov function V_i for the x_i subsystem such that the following hold:*

1. *there exist functions $\underline{\alpha}_i, \bar{\alpha}_i \in \mathcal{K}_\infty$, such that*

$$\underline{\alpha}_i(|x_i|) \leq V_i(x_i) \leq \bar{\alpha}_i(|x_i|), \ \forall x_i \in \mathbb{R}^{n_i}; \tag{9.17}$$

2. *there exist class $\mathcal{K}$ functions χ_i, γ_i and a class $\mathcal{K}_\infty$ function α_i, such that*

$$\nabla V_1(x_1)^T f_1(x_1, x_2, v) \leq -\alpha_1(V_1(x_1)), \tag{9.18}$$

if $V_1(x_1) \geq \max\{\chi_1(V_2(x_2)), \gamma_1(|v|)\}$, and

$$\nabla V_2(x_2)^T f_2(x_1, x_2, v) \leq -\alpha_2(V_2(x_2)), \tag{9.19}$$

if $V_2(x_2) \geq \max\{\chi_2(V_1(x_1)), \gamma_2(|v|)\}$.

Based on the ISS-Lyapunov functions, the following theorem gives the small-gain condition, under which the ISS property of the interconnected system can be achieved.

Theorem 9.2.1 ([81]). *Under Assumption 9.2.1, if the following small-gain condition holds:*

$$\chi_1 \circ \chi_2(s) < s, \quad \forall s > 0, \tag{9.20}$$

then, the interconnected system (9.15), (9.16) is ISS with respect to v as the input.

Under Assumption 9.2.1 and the small-gain condition (9.20), Let $\hat{\chi}_1$ be a function of class $\mathcal{K}_\infty$ such that

1. $\hat{\chi}_1(s) \leq \chi_1^{-1}(s)$, $\forall s \in [0, \lim_{s\to\infty} \chi_1(s))$,
2. $\chi_2(s) \leq \hat{\chi}_1(s)$, $\forall s \geq 0$.

Then, as shown in [81], there exists a class $\mathcal{K}_\infty$ function $\sigma(s)$ which is continuously differentiable over $(0, \infty)$ and satisfies $\frac{d\sigma}{ds}(s) > 0$ and $\chi_2(s) < \sigma(s) < \hat{\chi}_1(s)$, $\forall s > 0$.

In [81], it is also shown that the function

$$V_{12}(x_1, x_2) = \max\{\sigma(V_1(x_1)), V_2(x_2)\} \tag{9.21}$$

is positive definite and proper. In addition, we have

$$\dot{V}_{12}(x_1, x_2) < 0 \tag{9.22}$$

holds almost everywhere in the state space, whenever

$$V_{12}(x_1, x_2) \geq \eta(|v|) > 0 \tag{9.23}$$

for some class $\mathcal{K}_\infty$ function η.

9.3 Matlab code for the simulation in Chapter 2

For the readers' convenience, here I put the MATLAB code for the very first simulation in the dissertation. All the other simulations can be made using the same techniques. A few other available MATLAB programs can be found on the author's personal website at http://files.nyu.edu/yj348/public/index.html. They are also available upon request.

```
% Code for the paper "Computational adaptive optimal control with an
% application to a car engine control problem",  Yu Jiang and Zhong-Ping
% Jiang,vol. 48, no. 10, pp. 2699-2704, Oct. 2012.
% \copyright Copyright 2011-2014 Yu Jiang, New York University.

function []=engine_main()
clc;
x_save=[];
t_save=[];
flag=1;  % 1: learning is on. 0: learning is off.

% System matrices used for simulation purpose
A=[-0.4125      -0.0248        0.0741     0.0089   0          0;
    101.5873    -7.2651        2.7608     2.8068   0          0;
    0.0704       0.0085       -0.0741    -0.0089   0          0.0200;
    0.0878       0.2672        0         -0.3674   0.0044     0.3962;
    -1.8414      0.0990        0          0       -0.0343    -0.0330;
    0            0             0       -359      187.5364   -87.0316];

B=[-0.0042  0.0064
    -1.0360  1.5849
    0.0042  0;
    0.1261  0;
    0      -0.0168;
    0       0];

[xn,un]=size(B);%size of B. un-column #, xn row #

% Set the weighting matrices for the cost function
Q=diag([1 1 0.1 0.1 0.1 0.1]);
R=eye(2);
```

```
% Initialize the feedback gain matrix
K=zeros(un,xn);  % Only if A is Hurwitz, K can be set as zero.
N=200;  %Length of the window, should be at least greater than xn^2
NN=10;  %Max iteration times
T=.01;  %Duration of time for each integration

%x0=[10;2;100;2;-1;-2]; %Initial condition
x0=[10;2;10;2;-1;-2];
i1=(rand(1,100)-.5)*1000;
i2=(rand(1,100)-.5)*1000;

Dxx=[];XX=[];XU=[];  % Data matrices

X=[x0;kron(x0',x0')';kron(x0,zeros(un,1))]';

% Run the simulation and obtain the data matrices \delta_{xx}, I_{xx},
% and I_{xu}

for i=1:N
    % Simulation the system and at the same time collect online info.
    [t,X]=ode45(@mysys, [(i-1)*T,i*T],X(end,:));

    %Append new data to the data matrices
    Dxx=[Dxx;kron(X(end,1:xn),X(end,1:xn))-kron(X(1,1:xn),X(1,1:xn))];
    XX=[XX;X(end,xn+1:xn+xn^2)-X(1,xn+1:xn+xn^2)];
    XU=[XU;X(end,xn+xn^2+1:end)-X(1,xn+xn^2+1:end)];

    % Keep track of the system trajectories
    x_save=[x_save;X];
    t_save=[t_save;t];
end

Dxx=processing_Dxx(Dxx); % Only the distinct columns left

% K=zeros(un,xn);  % Initial stabilizing feedback gain matrix
P_old=zeros(xn);P=eye(xn)*10; % Initialize the previous cost matrix
it=0;            % Counter for iterations
p_save=[];       % Track the cost matrices in all the iterations
k_save=[];       % Track the feedback gain matrix in each iterations

[K0,P0]=lqr(A,B,Q,R) % Calculate the ideal solution for comparion purpose
k_save=[norm(K-K0)];

while norm(P-P_old)>1e-10 & it<16   % Stopping criterion for learning
    it=it+1                         % Update and display the # of iters
```

```
    P_old=P;                          % Update the previous cost matrix
    QK=Q+K'*R*K;                      % Update the Qk matrix
    X2=XX*kron(eye(xn),K');           %
    X1=[Dxx,-X2-XU];                  % Left-hand side of the key equation
    Y=-XX*QK(:);                      % Right-hand side of the key equation
    pp=X1\Y;                          % Solve the equations in the LS sense
    P=reshape_p(pp);                  % Reconstruct the symmetric matrix
    p_save=[p_save,norm(P-P0)];       % Keep track of the cost matrix
    BPv=pp(end-(xn*un-1):end);
    K=inv(R)*reshape(BPv,un,xn)/2     % Get the improved gain matrix
    k_save=[k_save,norm(K-K0)];       % Keep track of the control gains
end

% Plot the trajectories
figure(1)
plot([0:length(p_save)-1],p_save,'o',[0:length(p_save)-1],p_save)
axis([-0.5,it-.5,-5,15])
legend('||P_k-P^*||')
xlabel('Number of iterations')

figure(2)
plot([0:length(k_save)-1],k_save,'^',[0:length(k_save)-1],k_save)
axis([-0.5,it+0.5,-.5,2])
legend('||K_k-K^*||')
xlabel('Number of iterations')


% Post-learning simulation
[tt,xx]=ode23(@mysys,[t(end) 200],X(end,:)');

% Keep track of the post-learning trajectories
t_final=[t_save;tt];
x_final=[x_save;xx];


figure(3)
plot(t_final,x_final(:,1:6),'Linewidth',2)
axis([0,10,-100,200])
legend('x_1','x_2','x_3','x_4','x_5','x_6')
xlabel('Time (sec)')


figure(4)
plot(t_final,sqrt(sum(x_final(:,1:6).^2,2)),'Linewidth',2)
axis([0,200,-50,200])
```

```
legend('||x||')
xlabel('Time (sec)')

figure(5)
plot(t_final,3.6*x_final(:,1),'k-.', ...
    t_final, x_final(:,6),'-','Linewidth',2)
axis([0,10,-80,50])
legend('y_1 (MAF)','y_2 (MAP)')
xlabel('Time (sec)')

% The following nested function gives the dynamics of the sytem. Also,
% integraters are included for the purpose of data collection.
    function dX=mysys(t,X)
        %global A B xn un i1 i2 K flag
        x=X(1:xn);

        if t>=2;   % See if learning is stopped
            flag=0;
        end

        if flag==1
            u=zeros(un,1);
            for i=i1
                u(1)=u(1)+sin(i*t)/length(i1); % constructing the
                % exploration noise
            end
            for i=i2
                u(2)=u(2)+sin(i*t)/length(i2);
            end
            u=10000*u;
        else
            u=-K*x;
        end

        dx=A*x+B*u;
        dxx=kron(x',x')';
        dux=kron(x',u')';
        dX=[dx;dxx;dux];
    end

% This nested function reconstruct the P matrix from its distinct elements
    function P=reshape_p(p)
        P=zeros(xn);
        ij=0;
        for i=1:xn
```

```
            for j=1:i
                ij=ij+1;
                P(i,j)=p(ij);
                P(j,i)=P(i,j);
            end
        end
    end

% The following nested function removes the repeated columns from Dxx
    function Dxx=processing_Dxx(Dxx)
        ij=[]; ii=[];

        for i=1:xn
            ii=[ii (i-1)*xn+i];
        end

        for i=1:xn-1
            for j=i+1:xn
                ij=[ij (i-1)*xn+j];
            end
        end

        Dxx(:,ii)=Dxx(:,ii)/2;
        Dxx(:,ij)=[];
        Dxx=Dxx*2;
    end
end
```

Bibliography

[1] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf. Model-free Q-learning designs for linear discrete-time zero-sum games with application to h-infinity control. *Automatica*, 43(3):473–481, 2007.

[2] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf. Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 38(4):943–949, 2008.

[3] D. Angeli and E. D. Sontag. Forward completeness, unboundedness observability, and their Lyapunov characterizations. *Systems & Control Letters*, 38(4):209–217, 1999.

[4] L. C. Baird. Reinforcement learning in continuous time: Advantage updating. In *Proceedings of the IEEE World Congress on Computational Intelligence*, volume 4, pages 2448–2453, 1994.

[5] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man and Cybernetics*, 13(5):834–846, 1983.

[6] R. W. Beard, G. N. Saridis, and J. T. Wen. Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica*, 33(12):2159–2177, 1997.

[7] R. Bellman and S. Dreyfus. Functional approximations and dynamic programming. *Mathematical Tables and Other Aids to Computation*, 13(68):247–251, 1959.

[8] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[9] M. Berniker and K. Kording. Estimating the sources of motor errors for adaptation and generalization. *Nature Neuroscience*, 11(12):1454–1461, 2008.

[10] D. P. Bertsekas. *Dynamic Programming and Optimal Control, 4th ed.* Athena Scientific Belmont, Belmonth, MA, 2007.

[11] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Nashua, NH, 1996.

[12] S. Bhasin, N. Sharma, P. Patre, and W. Dixon. Asymptotic tracking by a reinforcement learning-based adaptive critic controller. *Journal of Control Theory and Applications*, 9(3):400–409, 2011.

[13] N. Bhushan and R. Shadmehr. Computational nature of human adaptive control during learning of reaching movements in force fields. *Biological Cybernetics*, 81(1):39–60, 1999.

[14] T. Bian, Y. Jiang, and Z. P. Jiang. Adaptive dynamic programming and optimal control of nonlinear nonaffine systems. *Automatica, provisionally accepted*, 2014.

[15] T. Bian, Y. Jiang, and Z. P. Jiang. Adaptive dynamic programming for stochastic systems with state and control dependent noise. *IEEE Transactions on Automatic Control, submitted*, 2014.

[16] T. Bian, Y. Jiang, and Z. P. Jiang. Decentralized and adaptive optimal control of large-scale systems with application to power systems. *IEEE Transactions on Industrial Electronics, revised*, Apr 2014.

[17] G. Blekherman, P. A. Parrilo, and R. R. Thomas, editors. *Semidefinite Optimization and Convex Algebraic Geometry*. SIAM, Philadelphia, PA, 2013.

[18] V. S. Borkar. *Stochastic approximation: a dynamical systems viewpoint*. Cambridge University Press Cambridge, 2008.

[19] S. J. Bradtke, B. E. Ydstie, and A. G. Barto. Adaptive linear quadratic control using policy iteration. In *Proceedings of the American Control Conference*, volume 3, pages 3475–3479. IEEE, 1994.

[20] D. A. Bristow, M. Tharayil, and A. G. Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006.

[21] E. Burdet, R. Osu, D. Franklin, T. Yoshioka, T. Milner, and M. Kawato. A method for measuring endpoint stiffness during multi-joint arm movements. *Journal of Biomechanics*, 33(12):1705–1709, 2000.

[22] E. Burdet, R. Osu, D. W. Franklin, T. E. Milner, and M. Kawato. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414(6862):446–449, 2001.

[23] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst. *Reinforcement learning and dynamic programming using function approximators*. CRC Press, 2010.

[24] M. Chen. Some simple synchronization criteria for complex dynamical networks. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 53(11):1185–1189, 2006.

[25] P.-C. Chen and M. Kezunovic. Analysis of the impact of distributed generation placement on voltage profile in distribution systems. In *Proceedings of the 2013 IEEE Power and Energy Society General Meeting (PES)*, pages 1–5, Vancouver, BC, Jul 2013.

[26] P.-C. Chen, V. Malbasa, and M. Kezunovic. Analysis of voltage stability issues with distributed generation penetration in distribution networks. In *Proceedings of the 2013 IEEE North American Power Symposium (NAPS)*, pages 1–6, Manhattan, KS, 2013.

[27] P.-C. Chen, R. Salcedo, Q. Zhu, F. de León, D. Czarkowski, Z. P. Jiang, V. Spitsa, Z. Zabar, and R. E. Uosef. Analysis of voltage profile problems due to the penetration of distributed generation in low-voltage secondary distribution networks. *IEEE Transactions on Power Delivery*, 27(4):2020–2028, 2012.

[28] Z. Chen and J. Huang. Global robust stabilization of cascaded polynomial systems. *Systems & Control Letters*, 47(5):445–453, 2002.

[29] Z. Chen, L. Wu, and Y. Fu. Real-time price-based demand response management for residential appliances via stochastic optimization and robust optimization. *IEEE Transactions on Smart Grid*, 3(4):1822–1831, 2012.

[30] D. A. Cox. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2007.

[31] P. R. Davidson and D. M. Wolpert. Motor learning and prediction in a variable environment. *Current Opinion in Neurobiology*, 13(2):232–237, 2003.

[32] D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[33] J. Diedrichsen, R. Shadmehr, and R. B. Ivry. The coordination of movement: optimal feedback control and beyond. *Trends in Cognitive Sciences*, 14(1):31–39, 2010.

[34] T. Dierks and S. Jagannathan. Output feedback control of a quadrotor uav using neural networks. *IEEE Transactions on Neural Networks*, 21(1):50–66, 2010.

[35] K. Doya. Reinforcement learning in continuous time and space. *Neural Computation*, 12(1):219–245, 2000.

[36] K. Doya, H. Kimura, and M. Kawato. Neural mechanisms of learning and control. *IEEE Control Systems Magazine*, 21(4):42–54, 2001.

[37] P. D. Feldkamp LA. Recurrent neural networks for state estimation. In *Proceedings of the Twelve Yale Workshop on Adaptive and Learning Systems*, pages 17–22, New Haven, CT, 2003.

[38] S. Ferrari, J. E. Steck, and R. Chandramohan. Adaptive feedback control by constrained approximate dynamic programming. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(4):982–987, 2008.

[39] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

[40] T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5(7):1688–1703, 1985.

[41] D. W. Franklin, E. Burdet, R. Osu, M. Kawato, and T. E. Milner. Functional significance of stiffness in adaptation of multijoint arm movements to stable and unstable dynamics. *Experimental Brain Research*, 151(2):145–157, 2003.

[42] D. W. Franklin, E. Burdet, K. P. Tee, R. Osu, C.-M. Chew, T. E. Milner, and M. Kawato. CNS learns stable, accurate, and efficient movements using a simple algorithm. *The Journal of Neuroscience*, 28(44):11165–11173, 2008.

[43] D. W. Franklin and D. M. Wolpert. Computational mechanisms of sensorimotor control. *Neuron*, 72(3):425–442, 2011.

[44] G. Franze, D. Famularo, and A. Casavola. Constrained nonlinear polynomial time-delay systems: A sum-of-squares approach to estimate the domain of attraction. *IEEE Trans. Auto. Contr.*, 57(10):2673–2679, 2012.

[45] P. Gahinet, A. Nemirovskii, A. J. Laub, and M. Chilali. The LMI control toolbox. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, volume 3, pages 2038–2041, 1994.

[46] W. Gao, Y. Jiang, Z. P. Jiang, and T. Chai. Adaptive and optimal output feedback control of linear systems: an adaptive dynamic programming approach. In *The 11th World Congress on Intelligent Control and Automation, accepted*, Shenyang, China, June 2014.

[47] H. Gomi and M. Kawato. Equilibrium-point control hypothesis examined by measured arm stiffness during multijoint movement. *Science*, 272:117–120, 1996.

[48] M. Grant and S. Boyd. Cvx: Matlab software for disciplined convex programming, version 2.0 beta. *http://cvxr.com/cvx*, 2013.

[49] G. Guo, Y. Wang, and D. J. Hill. Nonlinear output stabilization control for multimachine power systems. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 47(1):46–53, 2000.

[50] C. M. Harris and D. M. Wolpert. Signal-dependent noise determines motor planning. *Nature*, 394:780–784, 1998.

[51] H. He, Z. Ni, and J. Fu. A three-network architecture for on-line learning and optimization based on adaptive dynamic programming. *Neurocomputing*, 78(1):3–13, 2012.

[52] J. W. Helton and M. R. James. *Extending H_∞ Control to Nonlinear Systems: Control of Nonlinear Systems to Achieve Performance Objectives*. SIAM, 1999.

[53] D. Henrion and J.-B. Lasserre. Gloptipoly: Global optimization over polynomials with Matlab and SeDuMi. *ACM Transactions on Mathematical Software*, 29(2):165–194, 2003.

[54] N. Hogan. The mechanics of multi-joint posture and movement control. *Biological Cybernetics*, 52(5):315–331, 1985.

[55] N. Hogan and T. Flash. Moving gracefully: quantitative theories of motor coordination. *Trends in Neurosciences*, 10(4):170–174, 1987.

[56] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, 1990.

[57] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

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

[59] T. E. Hudson and M. S. Landy. Adaptation to sensory–motor reflex perturbations is blind to the source of errors. *Journal of Vision*, 12(1):1–10, 2012.

[60] P. A. Ioannou and J. Sun. *Robust Adaptive Control*. Prentice-Hall, Upper Saddle River, NJ, 1996.

[61] A. Isidori. *Nonlinear control systems Vol. 2*. Springer, 1999.

[62] K. Itô. Stochastic integral. *Proceedings of the Japan Academy, Series A, Mathematical Sciences*, 20(8):519–524, 1944.

[63] J. Izawa, T. Rane, O. Donchin, and R. Shadmehr. Motor adaptation as a process of reoptimization. *The Journal of Neuroscience*, 28(11):2883–2891, 2008.

[64] J. Izawa and R. Shadmehr. Learning from sensory and reward prediction errors during motor adaptation. *PLoS Computational Biology*, 7(3):e1002012, 2011.

[65] Y. Jiang, S. Chemudupati, J. M. Jorgensen, Z. P. Jiang, and C. S. Peskin. Optimal control mechanism involving the human kidney. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 3688–3693, Orlando, FL, 2011.

[66] Y. Jiang and Z. P. Jiang. Approximate dynamic programming for optimal stationary control with control-dependent noise. *Neural Networks, IEEE Transactions on*, 22(12):2392–2398, 2011.

[67] Y. Jiang and Z. P. Jiang. Robust approximate dynamic programming and global stabilization with nonlinear dynamic uncertainties. In *Proceedings of the Joint Decision and Control Conference and European Control Conference (CDC-ECC)*, pages 115–120. IEEE, 2011.

[68] Y. Jiang and Z. P. Jiang. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*, 48(10):2699–2704, 2012.

[69] Y. Jiang and Z. P. Jiang. Robust adaptive dynamic programming. In D. Liu and F. Lewis, editors, *Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control*, chapter 13, pages 281–302. John Wiley & Sons, 2012.

[70] Y. Jiang and Z. P. Jiang. Robust adaptive dynamic programming for large-scale systems with an application to multimachine power systems. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 59(10):693–697, 2012.

[71] Y. Jiang and Z. P. Jiang. Global adaptive dynamic programming for continuous-time nonlinear systems. *arXiv preprint arXiv:1401.0020*, 2013.

[72] Y. Jiang and Z. P. Jiang. Robust adaptive dynamic programming for optimal nonlinear control design. *arXiv preprint arXiv:1303.2247v1 [math.DS]*, 2013.

[73] Y. Jiang and Z. P. Jiang. Robust adaptive dynamic programming with an application to power systems. *IEEE Transactions on Neural Networks and Learning Systems*, 24(7):1150–1156, 2013.

[74] Y. Jiang and Z. P. Jiang. Adaptive dynamic programming as a theory of sensorimotor control. *Biological Cybernetics*, May 2014.

[75] Y. Jiang and Z. P. Jiang. Robust adaptive dynamic programming and feedback stabilization of nonlinear systems. *IEEE Transactions on Neural Netorks and Learning Systems*, 25(5):882–893, 2014.

[76] Y. Jiang and Z. P. Jiang. A robust adaptive dynamic programming principle for sensorimotor control with signal-dependent noise. *Journal of Systems Science and Complexity*, Mar 2014.

[77] Y. Jiang, Z. P. Jiang, and N. Qian. Optimal control mechanisms in human arm reaching movements. In *Proceedings of the 30th Chinese Control Conference*, pages 1377–1382, Yantai, China, 2011. IEEE.

[78] Z. P. Jiang. Decentralized control for large-scale nonlinear systems: a review of recent results. *Dynamics of Continuous, Discrete and Impulsive Systems*, 11:537–552, 2004.

[79] Z. P. Jiang and Y. Jiang. Robust adaptive dynamic programming for linear and nonlinear systems: An overview. *European Journal of Control*, 19(5):417–425, 2013.

[80] Z. P. Jiang and I. Mareels. A small-gain control method for nonlinear cascaded systems with dynamic uncertainties. *IEEE Transactions on Automatic Control*, 42(3):292–308, 1997.

[81] Z. P. Jiang, I. M. Mareels, and Y. Wang. A Lyapunov formulation of the nonlinear small-gain theorem for interconnected ISS systems. *Automatica*, 32(8):1211–1215, 1996.

[82] Z. P. Jiang and L. Praly. Design of robust adaptive controllers for nonlinear systems with dynamic uncertainties. *Automatica*, 34(7):825–840, 1998.

[83] Z. P. Jiang, A. R. Teel, and L. Praly. Small-gain theorem for ISS systems and applications. *Mathematics of Control, Signals and Systems*, 7(2):95–120, 1994.

[84] M. Jung, K. Glover, and U. Christen. Comparison of uncertainty parameterisations for h_∞ robust control of turbocharged diesel engines. *Control Engineering Practice*, 13(1):15–25, 2005.

[85] I. Karafyllis and Z. P. Jiang. *Stability and stabilization of nonlinear systems*. Springer, 2011.

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

[87] Y. H. Kim and F. L. Lewis. *High-level feedback control with neural networks*. World Scientific, 1998.

[88] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M.-B. Naghibi-Sistani. Reinforcement q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica*, 50(4):1167–1175, 2014.

[89] D. Kleinman. On an iterative technique for riccati equation computations. *IEEE Transactions on Automatic Control*, 13(1):114–115, 1968.

[90] D. Kleinman. On the stability of linear stochastic systems. *IEEE Transactions on Automatic Control*, 14(4):429–430, 1969.

[91] D. Kleinman. Optimal stationary control of linear systems with control-dependent noise. *IEEE Transactions on Automatic Control*, 14(6):673–677, 1969.

[92] K. P. Kording, J. B. Tenenbaum, and R. Shadmehr. The dynamics of memory as a consequence of optimal adaptation to a changing body. *Nature Neuroscience*, 10(6):779–786, 2007.

[93] M. Krstic and H. Deng. *Stabilization of Nonlinear Uncertain Systems*. Springer, 1998.

[94] M. Krstic, D. Fontaine, P. V. Kokotovic, and J. D. Paduano. Useful nonlinearities and global stabilization of bifurcations in a model of jet engine surge and stall. *IEEE Transactions on Automatic Control*, 43(12):1739–1745, 1998.

[95] M. Krstic, I. Kanellakopoulos, and P. V. Kokotovic. *Nonlinear and Adaptive Control Design*. John Wiley & Sons, New York, 1995.

[96] M. Krstic and Z.-H. Li. Inverse optimal design of input-to-state stabilizing nonlinear controllers. *IEEE Transactions on Automatic Control*, 43(3):336–350, 1998.

[97] P. Kundur, N. J. Balu, and M. G. Lauby. *Power system stability and control*, volume 7. McGraw-hill, New York, 1994.

[98] H. J. Kushner. *Stochastic stability*. Springer, Berlin Heidelberg, 1972.

[99] F. Lewis and V. Syrmos. *Optimal Control*. Wiley, 1995.

[100] F. L. Lewis and D. Liu. *Reinforcement learning and approximate dynamic programming for feedback control*. John Wiley & Sons, 2012.

[101] F. L. Lewis and K. G. Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(1):14–25, 2011.

[102] F. L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[103] F. L. Lewis, D. Vrabie, and V. L. Syrmos. *Optimal Control, 3rd ed.* Wiley, New York, 2012.

[104] Z. Li and G. Chen. Global synchronization and asymptotic stability of complex dynamical networks. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 53(1):28–33, 2006.

[105] Z. Li, Z. Duan, G. Chen, and L. Huang. Consensus of multiagent systems and synchronization of complex networks: a unified viewpoint. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 57(1):213–224, 2010.

[106] B. Lincoln and A. Rantzer. Relaxing dynamic programming. *IEEE Transactions on Automatic Control*, 51(8):1249–1260, 2006.

[107] D. Liu and W. D. Optimal control of unknown nonlinear discrete-time systems using iterative globalized dual heuristic programming algorithm. In F. Lewis and L. D, editors, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*, pages 52–77. John Wiley & Sons, 2012.

[108] D. Liu and E. Todorov. Evidence for the flexible sensorimotor strategies predicted by optimal feedback control. *The Journal of Neuroscience*, 27(35):9354–9368, 2007.

[109] T. Liu, D. J. Hill, and Z. P. Jiang. Lyapunov formulation of ISS cyclic-small-gain in continuous-time dynamical networks. *Automatica*, 47(9):2088–2093, 2011.

[110] Y. Liu, T. Chen, C. Li, Y. Wang, and B. Chu. Energy-based disturbance L_2 attenuation excitation control of differential algebraic power systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 55(10):1081–1085, 2008.

[111] L. Ljung. *System Identification*. Wiley, 1999.

[112] J. Lofberg. Yalmip: A toolbox for modeling and optimization in Matlab. In *Proceedings of 2004 IEEE International Symposium on Computer Aided Control Systems Design*, pages 284–289, 2004.

[113] I. Mareels and J. W. Polderman. *Adaptive systems*. Springer, 1996.

[114] R. Marino and P. Tomei. *Nonlinear control design: geometric, adaptive and robust*. Prentice Hall International, Ltd., 1996.

[115] J. Mendel and R. McLaren. Reinforcement-learning control and pattern recognition systems. In *A prelude to neural networks*, pages 287–318. Prentice Hall Press, 1994.

[116] A. N. Michel. On the status of stability of interconnected systems. *IEEE Transactions on Automatic Control*, 28(6):639–653, 1983.

[117] M. Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

[118] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani. Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 24(10):1513–1525, 2013.

[119] F. Moore and E. Greitzer. A theory of post-stall transients in axial compression systems: Part idevelopment of equations. *Journal of engineering for gas turbines and power*, 108(1):68–76, 1986.

[120] P. Morasso. Spatial control of arm movements. *Experimental Brain Research*, 42(2):223–227, 1981.

[121] E. Moulay and W. Perruquetti. Stabilization of nonaffine systems: A constructive method for polynomial systems. *IEEE Transactions on Automatic Control*, 50(4):520–526, 2005.

[122] J. J. Murray, C. J. Cox, G. G. Lendaris, and R. Saeks. Adaptive dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 32(2):140–153, 2002.

[123] F. A. Mussa-Ivaldi, N. Hogan, and E. Bizzi. Neural, mechanical, and geometric factors subserving arm posture in humans. *The Journal of Neuroscience*, 5(10):2732–2743, 1985.

[124] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.

[125] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, Pasadena, California, 2000.

[126] M. J. D. Powell. *Approximation theory and methods*. Cambridge university press, 1981.

[127] W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, New York, 2007.

[128] S. Prajna, A. Papachristodoulou, and P. A. Parrilo. Introducing sostools: A general purpose sum of squares programming solver. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 741–746, 2002.

[129] L. Praly and Y. Wang. Stabilization in spite of matched unmodeled dynamics and an equivalent definition of input-to-state stability. *Mathematics of Control, Signals and Systems*, 9(1):1–33, 1996.

[130] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*, volume 414. John Wiley & Sons, 2009.

[131] N. Qian, Y. Jiang, Z. P. Jiang, and P. Mazzoni. Movement duration, Fitts's law, and an infinite-horizon optimal feedback control model for biological motor systems. *Neural Computation*, 25(3):697–724, 2013.

[132] G. Revel, A. E. Leon, D. M. Alonso, and J. L. Moiola. Bifurcation analysis on a multimachine power system model. *IEEE Transactions on Robust nonlinear coordinated control of power systemsCircuits and Systems I: Regular Papers*, 57(4):937–949, 2010.

[133] A. Saberi, P. Kokotovic, and S. Summers. Global stabilization of partially linear composite systems. *SIAM*, 2(6):1491–1503, 1990.

[134] R. Salcedo, X. Ran, F. de León, D. Czarkowski, and V. Spitsa. Long duration overvoltages due to current backfeeding in secondary networks. *IEEE Transactions on Power Delivery*, 28(4), Oct 2013.

[135] N. R. Sandell, P. Varaiya, M. Athans, and M. G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, 23(2):108–128, 1978.

[136] G. N. Saridis and C.-S. G. Lee. An approximation theory of optimal control for trainable manipulators. *IEEE Transactions on Systems, Man and Cybernetics*, 9(3):152–159, 1979.

[137] C. Savorgnan, J. B. Lasserre, and M. Diehl. Discrete-time stochastic optimal control via occupation measures and moment relaxations. In *Proceedings of the Joint 48th IEEE Conference on Decison and Control and the 28th Chinese Control Conference, Shanghai, P. R. China*, pages 519–524, 2009.

[138] R. A. Schmidt and T. D. Lee. *Motor Control and Learning: A Behavioral Emphasis*. Human Kinetics, 5 edition, 2011.

[139] P. J. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110(2):568–582, 1985.

[140] S. H. Scott. Optimal feedback control and the neural basis of volitional motor control. *Nature Reviews Neuroscience*, 5(7):532–546, 2004.

[141] R. Sepulchre, M. Jankovic, and P. Kokotovic. *Constructive Nonlinear Control*. Springer Verlag, New York, 1997.

[142] R. Shadmehr and F. A. Mussa-Ivaldi. Adaptive representation of dynamics during learning of a motor task. *The Journal of Neuroscience*, 14(5):3208–3224, 1994.

[143] Y. She, X. She, and M. E. Baran. Universal tracking control of wind conversion system for purpose of maximum power acquisition under hierarchical control structure. *IEEE Transactions on Energy Conversion*, 26(3):766–775, 2011.

[144] J. Si, A. G. Barto, W. B. Powell, D. C. Wunsch, et al., editors. *Handbook of learning and approximate dynamic programming*. Wiley, Inc., Hoboken, NJ, 2004.

[145] D. D. Siljak. *Large-scale dynamic systems: stability and structure*, volume 310. North-Holland New York, 1978.

[146] Q. Song and J. Cao. On pinning synchronization of directed and undirected complex dynamical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 57(3):672–680, 2010.

[147] E. D. Sontag. On the observability of polynomial systems, I: Finite-time problems. *SIAM Journal on Control and Optimization*, 17(1):139–151, 1979.

[148] E. D. Sontag. Smooth stabilization implies coprime factorization. *IEEE Transactions on Automatic Control*, 34(4):435–443, 1989.

[149] E. D. Sontag. Further facts about input to state stabilization. *IEEE Transactions on Automatic Control*, 35(4):473–476, 1990.

[150] E. D. Sontag. Input to state stability: Basic concepts and results. In *Nonlinear and optimal control theory*, pages 163–220. Springer, 2008.

[151] E. D. Sontag and Y. Wang. On characterizations of the input-to-state stability property. *Systems & Control Letters*, 24(5):351–359, 1995.

[152] V. Spitsa, X. Ran, R. Salcedo, J. F. Martinez, R. E. Uosef, F. de León, D. Czarkowski, and Z. Zabar. On the transient behavior of large-scale distribution networks during automatic feeder reconfiguration. *IEEE Transactions on Smart Grid*, 3(2):887–896, 2012.

[153] T. H. Summers, K. Kunz, N. Kariotoglou, M. Kamgarpour, S. Summers, and J. Lygeros. Approximate dynamic programming via sum of squares programming. *arXiv preprint arXiv:1212.1269*, 2012.

[154] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[155] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge Univ Press, 1998.

[156] C. Szepesvari. Reinforcement learning algorithms for mdps. Technical Report Report TR09-13, Department of Computing Science, University of Alberta, Edmonton, CA, 2009.

[157] H. Tanaka, J. W. Krakauer, and N. Qian. An optimization principle for determining movement duration. *Journal of neurophysiology*, 95(6):3875–3886, 2006.

[158] G. Tao. *Adaptive Control Design and Analysis*. Wiley, 2003.

[159] A. Taware and G. Tao. *Control of Sandwich Nonlinear Systems*. Springer, 2003.

[160] K. P. Tee, D. W. Franklin, M. Kawato, T. E. Milner, and E. Burdet. Concurrent adaptation of force and impedance in the redundant muscle system. *Biological Cybernetics*, 102(1):31–44, 2010.

[161] A. Teel and L. Praly. Tools for semiglobal stabilization by partial state and output feedback. *SIAM Journal on Control and Optimization*, 33(5):1443–1488, 1995.

[162] E. Todorov. Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system. *Neural Computation*, 17(5):1084–1108, 2005.

[163] E. Todorov and M. I. Jordan. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, 5(11):1226–1235, 2002.

[164] J. Tsinias. Partial-state global stabilization for general triangular systems. *Systems & control letters*, 24(2):139–145, 1995.

[165] Y. Uno, M. Kawato, and R. Suzuki. Formation and control of optimal trajectory in human multijoint arm movement: Minimum torque-change model. *Biological cybernetics*, 61(2):89–101, 1989.

[166] K. G. Vamvoudakis and F. L. Lewis. Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica*, 46(5):878–888, 2010.

[167] K. G. Vamvoudakis and F. L. Lewis. Multi-player non-zero-sum games: online adaptive learning solution of coupled hamilton–jacobi equations. *Automatica*, 47(8):1556–1569, 2011.

[168] K. G. Vamvoudakis and F. L. Lewis. Online solution of nonlinear two-player zero-sum games using synchronous policy iteration. *International Journal of Robust and Nonlinear Control*, 22(13):1460–1483, 2012.

[169] A. J. van der Schaft. *L_2-Gain and Passivity in Nonlinear Control*. Springer, Berlin, 1999.

[170] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[171] D. Vrabie and F. Lewis. Adaptive dynamic programming algorithm for finding online the equilibrium solution of the two-player zero-sum differential game. In *Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2010.

[172] D. Vrabie and F. L. Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Networks*, 22(3):237–246, 2009.

[173] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. Lewis. Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica*, 45(2):477–484, 2009.

[174] D. Vrabie, K. G. Vamvoudakis, and F. L. Lewis. *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. IET, London, 2013.

[175] D. D. Šiljak. *Decentralized Control of Complex Systems*. Academic Press, 1991.

[176] M. Waltz and K. Fu. A heuristic approach to reinforcement learning control systems. *IEEE Transactions on Automatic Control*, 10(4):390–398, 1965.

[177] F.-Y. Wang, H. Zhang, and D. Liu. Adaptive dynamic programming: an introduction. *IEEE Computational Intelligence Magazine*, 4(2):39–47, 2009.

[178] Y. Wang and S. Boyd. Approximate dynamic programming via iterated Bellman inequalities. *Manuscript preprint*, 2010.

[179] Y. Wang and S. Boyd. Performance bounds and suboptimal policies for linear stochastic control via lmis. *International Journal of Robust and Nonlinear Control*, 21(14):1710–1728, 2011.

[180] Y. Wang and D. J. Hill. Robust nonlinear coordinated control of power systems. *Automatica*, 32(4):611–618, 1996.

[181] C. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, 1989.

[182] K. Wei and K. Körding. Uncertainty of feedback and state estimation determines the speed of motor adaptation. *Frontiers in computational neuroscience*, 4, 2010.

[183] Q. Wei and D. Liu. Data-driven neuro-optimal temperature control of water gas shift reaction using stable iterative adaptive dynamic programming. *IEEE Transactions on Industrial Electronics, in press*, 2014.

[184] Q. Wei, H. Zhang, and J. Dai. Model-free multiobjective approximate dynamic programming for discrete-time nonlinear systems with general performance index functions. *Neurocomputing*, 72(7):1839–1848, 2009.

[185] P. Werbos. The elements of intelligence. *Cybernetica (Namur)*, (3), 1968.

[186] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[187] P. Werbos. Advanced forecasting methods for global crisis warning and models of intelligence. *General Systems Yearbook*, 22:25–38, 1977.

[188] P. Werbos. Reinforcement learning and approximate dynamic programming (RLADP) – Foundations, common misceonceptsions and the challenges ahead. In F. L. Lewis and D. Liu, editors, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*, pages 3–30. Wiley, Hoboken, NJ, 2013.

[189] P. J. Werbos. Neural networks for control and system identification. In *Proceedings of the 28th IEEE Conference on Decision and Control*, pages 260–265, 1989.

[190] P. J. Werbos. A menu of designs for reinforcement learning over time. In W. Miller, R. Sutton, and P. Werbos, editors, *Neural Networks for Control*, pages 67–95. MIT Press, Cambridge, MA, 1990.

[191] P. J. Werbos. Approximate dynamic programming for real-time control and neural modeling. In D. White and D. Sofge, editors, *Handbook of intelligent control: Neural, fuzzy, and adaptive approaches*, pages 493–525. Van Nostrand Reinhold, New York, 1992.

[192] P. J. Werbos. Intelligence in the brain: A theory of how it works and how to build it. *Neural Networks*, 22(3):200–212, 2009.

[193] P. J. Werbos. From adp to the brain: Foundations, roadmap, challenges and research priorities. *arXiv preprint arXiv:1404.0554*, 2014.

[194] J. L. Willems and J. C. Willems. Feedback stabilizability for stochastic systems with state and control dependent noise. *Automatica*, 12(3):277–283, 1976.

[195] D. M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *Nature Neuroscience*, 3:1212–1217, 2000.

[196] H. Xu, S. Jagannathan, and F. L. Lewis. Stochastic optimal control of unknown linear networked control system in the presence of random delays and packet losses. *Automatica*, 48(6):1017–1030, 2012.

[197] J. Xu, L. Xie, and Y. Wang. Simultaneous stabilization and robust control of polynomial nonlinear systems using SOS techniques. *IEEE Transactions on Automatic Control*, 54(8):1892–1897, 2009.

[198] X. Xu, C. Wang, and F. L. Lewis. Some recent advances in learning and adaptation for uncertain feedback control systems. *International Journal of Adaptive Control and Signal Processing*, 28(3-5):201–204, 2014.

[199] C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schaeffer, and E. Burdet. Human-like adaptation of force and impedance in stable and unstable interactions. *IEEE Transactions on Robotics*, 27(5):918–930, 2011.

[200] X. Yang, J. Cao, and J. Lu. Stochastic synchronization of complex networks with nonidentical nodes via hybrid adaptive and impulsive control. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 59(2):371–384, 2012.

[201] L. Yu, D. Czarkowski, and F. de León. Optimal distributed voltage regulation for secondary networks with dgs. *IEEE Transactions on Smart Grid*, 3(2):959–967, 2012.

[202] H. Zhang, D. Liu, Y. Luo, and D. Wang. *Adaptive Dynamic Programming for Control*. Springer, London, 2013.

[203] H. Zhang, Q. Wei, and D. Liu. An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. *Automatica*, 47(1):207–214, 2011.

[204] X. Zhang, H. He, H. Zhang, and Z. Wang. Optimal control for unknown discrete-time nonlinear markov jump systems using adaptive dynamic programming. *IEEE Transactions on Neural Networks and Learning Systems, in press*, 2014.

[205] Y. Zhang, P.-Y. Peng, and Z. P. Jiang. Stable neural controller design for unknown nonlinear systems using backstepping. *IEEE Transactions on Neural Networks*, 11(6):1347–1360, 2000.

[206] J. Zhou and Y. Ohsawa. Improved swing equation and its properties in synchronous generators. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56(1):200–209, 2009.

[207] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control*. New Jersey, Prentice Hall, 1996.

[208] S.-H. Zhou, D. Oetomo, Y. Tan, E. Burdet, and I. Mareels. Modeling individual human motor behavior through model reference iterative learning control. *IEEE Transactions on Biomedical Engineering*, 59(7):1892–1901, 2012.