DISTRIBUTED ESTIMATION AND INVERSE REINFORCEMENT LEARNING FOR MULTI-AGENT SYSTEMS

by

BOSEN LIAN, M.S.

DISSERTATION
Presented to the Graduate Faculty of
The University of Texas at Arlington
In Partial Fulfillment
Of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

COMMITTEE MEMBERS:
Frank L. Lewis, Ph.D., Chair
Ali Davoudi, Ph.D.
Yan Wan, Ph.D.
Ramtin Madani, Ph.D.
Michael A. Niestroy, Ph.D.

THE UNIVERSITY OF TEXAS AT ARLINGTON
College of Engineering
Department of Electrical Engineering
December 2021

ACKNOWLEDGEMENTS

I wish to express my deep sense of thanks to all the people that have had positive influences on me during my doctoral studies at UTA.

I deeply thank my Ph.D. supervisor, Dr. Frank L. Lewis, for his teaching, his guidance and his dedication on the research during all these years. He taught me how to write, how to communicate, and how to think as a researcher. It is my great honor to work with him. He is also a friend who has a great sense of humor and makes me feel relaxed all the times.

I also wish to thank Dr. Yan Wan and Dr. Ali Davoudi, as co-authors of my papers, for their meaningful inputs and devotion of time to improve the works. It is nice to work with them. Moreover, many thanks to Dr. Ramtin Madani and Dr. Michael A Niestroy, as members of my dissertation committee, for their insightful comments to improve the dissertation. Moreover, I was teaching assistant of Dr. Niestroy. It's nice to work with him. His Kalman filter course is indeed helpful.

With all my heart I shall thank all of my friends, and collaborators at UTA and universities back in China. Wenqian Xue gives me both love and support for my work and life. I cannot write these nice papers without her. Yusuf Kartal and Patrik Kolaric, as my colleagues at UTA, provide their useful helps on inverse reinforcement learning and extensions to applications on quadrotor unmanned aerial vehicles. Moreover, Xiao Zhang and Zhe Chen, as visiting scholars from China to UTA, provide me chances to work with them on their papers. This broadens my research areas to clusters and finite-time optimizations. Their passion and dedication always inspire me.

Last but not least I want to thank my parents and my sister for their advice and encouragements, and the unconditional support.

The research was supported by Lockheed Martin Contract, National Science Foundation Grant 1839804, Office of Naval Research Grant N00014-18-1-2221, and Army Research Office Grant W911NF-20-1-0132.

December 2021

DISTRIBUTED ESTIMATION AND INVERSE REINFORCEMENT LEARNING FOR MULTI-AGENT SYSTEMS

Bosen Lian, Ph.D.
The University of Texas at Arlington, 2021

Supervising Professor: Frank L. Lewis, Ph.D.

Abstract Consensus-based distributed Kalman filters for estimation with multiple targets have attracted considerable attention. Most of the existing Kalman filters use the average consensus approach, which tends to have a low convergence speed. They also rarely consider the impacts of limited sensing range and target mobility on the information flow topology. The robustness properties, i.e., gain margins and phase margins of distributed Kalman filtering algorithms are still open problems.

In the interactions of controlled dynamical agents, it is often assumed that the agents are "rational" in the sense of attempting to act in such a way as to optimize some prescribed performance reward functions. Optimal control and reinforcement learning solve optimal control input solutions given a performance index. Inverse optimal control and inverse reinforcement learning can reconstruct the performance index given demonstrations. However, inverse optimal control needs to know system dynamics while inverse reinforcement learning can be model-free.

This dissertation first presents new distributed estimation methods for multi-agent systems. A novel distributed Kalman consensus filter (DKCF) with an information-weighted and consensus-based structure is proposed for estimation with random mobile targets in continuous-time dynamics. A new moving target information-flow topology for the measurement of targets is developed based on the sensors' sensing ranges, targets' random mobility, and local information-weighted neighbors. This work also studies the robustness margins (i.e., gain margins and phase margins) of a DKCF. It shows that the robustness results of the DKCF are improved compared to the single-agent KF

This dissertation then studies new inverse reinforcement learning (RL) algorithms for multi-

agent systems. New inverse reinforcement learning algorithms are proposed to solve two-player zero-sum games by proposing both model-based and model-free algorithms. The games are solved by extracting the unknown cost function of an expert by a learner using demonstrated expert's behaviors. Next, this work extends these results to multiplayer non-zero-sum games, where both the expert and the learner have N-player noncooperate control input players.

TABLE OF CONTENTS

Acknowledgements . . . . . **iii**

Abstract . . . . . **iv**

List of Figures . . . . . **ix**

Chapter 1: Introduction **1**

1.1 Distributed Estimation for Multi-agent Systems 1

1.2 Inverse Reinforcement Learning for Multi-agent Systems 4

1.3 Contributions 6

Chapter 2: Distributed Kalman Consensus Filter for Estimation with Moving Targets . **9**

2.1 Introduction 9

2.2 Formulation of Distributed Consensus Filtering Problem for Multiple Moving Targets 10

2.2.1 Graph Theory in Sensor Networks and Notations 10

2.2.2 Distributed Kalman Consensus Filters for Multi-target Sensor Networks . . 10

2.3 Convergence Analysis 21

2.4 Simulation Studies 29

2.4.1 Performance of the DKCF 29

2.4.2 Comparison Between KCF in [82] and DKCF 32

2.5 Conclusion 37

Chapter 3: Robustness Margins of Distributed Kalman Consensus Filter **39**

3.1 Introduction 39

3.2 Kalman Filter Robustness Margins 40

3.3 Robustness Analysis of Distributed Kalman-Consensus Filter 43

3.3.1 Graph Theory 43

3.3.2 Distributed Kalman Consensus Filter . . . 44
3.3.3 Robustness Margins of DKCF with Direct Target Observation . . . 47
3.3.4 Robustness Margins of DKCF with Neighbor Estimates Only . . . 56
3.4 Simulation Studies . . . 61
3.4.1 DKCF with Unstable Target Dynamics . . . 61
3.4.2 DKCF with Stable Target Dynamics . . . 63
3.4.3 Robustness Margin Analysis of DKCF . . . 63
3.5 Conclusion . . . 67

Chapter 4: Inverse RL for Adversarial Apprentice Games . . . 68
4.1 Introduction . . . 68
4.2 Problem Formulation . . . 69
4.2.1 Adversarial Apprentice Games . . . 69
4.2.2 Two-Player Zero-sum Expert System . . . 69
4.2.3 Learner System . . . 71
4.3 Model-based Inverse RL . . . 72
4.3.1 Optimal Control Learning . . . 73
4.3.2 State-penalty Weight Q_l Revision Based on Inverse Optimal Control . . . 74
4.3.3 Convergence and Stability Analysis . . . 76
4.3.4 Implementing Model-based Inverse RL Algorithm 4.1 via NNs . . . 80
4.4 Model-free Inverse RL . . . 84
4.4.1 Model-free Integral Inverse RL Algorithm . . . 84
4.4.2 Implementing Model-free Integral Inverse RL Algorithm 4.3 via NNs . . . 87
4.5 Simulation Studies . . . 91
4.5.1 Example 1: Linear Systems . . . 91
4.5.2 Example 2: Nonlinear Systems . . . 95
4.5.3 Comparison to Existing Optimal Tracking by RL Technique in [73] . . . 99
4.6 Conclusion . . . 101

Chapter 5: Inverse RL for Multi-player Noncooperative Apprentice Games 102

5.1 Introduction 102

5.2 Problem Formulation 103

5.2.1 Multi-player Noncooperative Apprentice Games 103

5.2.2 Multi-player non-zero-sum game expert system 103

5.2.3 N-player game learner system 105

5.3 Model-based Inverse RL for Multi-player Noncooperative Apprentice Games . . . 105

5.3.1 Optimal Control Learning of Learner System 106

5.3.2 IOC Learning for Performance Weight Q_i 106

5.3.3 Convergence and Stability Analysis 108

5.4 Completely Model-free Inverse RL for Homogeneous Control Inputs 112

5.4.1 Off-policy Inverse RL for Homogeneous Control Inputs 113

5.4.2 Implementation of Inverse RL Algorithm 5.2 via NNs 115

5.5 Partially Model-free Inverse RL for Heterogeneous Control Inputs 118

5.5.1 Inverse RL for Heterogeneous Control Inputs 118

5.5.2 Implementation of Inverse RL Algorithm 5.3 via NNs 119

5.6 Simulation Studies 121

5.7 Conclusion 126

Chapter 6: Future Directions 127

Bibliography 128

LIST OF FIGURES

2.1 Multiple moving targets and field of stationary sensors 11

2.2 Communication topology of the sensor network 30

2.3 The average covariance value P_{22} of different sensors for all targets under Prob Scheme 1 by DKCF 32

2.4 The average covariance value P_{22} of different sensors for all targets under Prob Scheme 2 by DKCF 33

2.5 The average estimation errors of different sensors for all targets by DKCF in case 1 34

2.6 The average estimation errors of different sensors for all targets by KCF in case 1 34

2.7 The average estimation errors of different sensors for all targets by DKCF in case 2 35

2.8 The average estimation errors of different sensors for all targets by KCF in case 2 35

2.9 Larger and sparser sensor network configuration 36

2.10 The average estimation errors of different sensor for all targets by DKCF for sensor network in Figure 2.9 36

2.11 The variation of MSE by KCF and DKCF with different probabilities . . . 37

3.1 Communication graph of six sensors with unstable target 62

3.2 Evaluation of covariance value P_{22} in DKCF of 6 sensors for the unstable target (3.64) 62

3.3 Evaluation of covariance value P_{22} in DKCF of 4 sensors for the stable target (3.65) 64

3.4 Example 1: communication graph changing with the overall coupling strength d_1 increasing for $g_1 = 1$ 64

3.5 Evaluation of covariance value P_{22} of sensor 1 in four cases. 65
3.6 Evaluation of $|\mathrm{PM}|$ and the lower bound of the GM for 4 cases in example 1. 65
3.7 Example 2: communication graph changing with the overall coupling strength d_1 increasing for $g_1 = 0$ 66
3.8 Evaluation of $|\mathrm{PM}|$ and the lower bound of the GM for 3 cases in example 2. 67

4.1 Schematic diagram of inverse RL for Adversarial Apprentice Game. 74
4.2 Trajectories of the linear learner using Algorithm 4.2 93
4.3 Convergence of the linear learner's state-penalty weight Q_l and feedback gain K_l using Algorithm 4.2. 93
4.4 Trajectories of the linear learner using Algorithm 4.4. 95
4.5 Convergence of the linear learner's state-penalty weight Q_l and feedback gain K_l using Algorithm 4.4. 95
4.6 Trajectories of states and control inputs of the nonlinear learner using Algorithm 4.2 97
4.7 Convergence of the nonlinear learner's state-penalty weight Q_l using Algorithm 4.2. 97
4.8 Trajectories of states and control inputs of the nonlinear learner using Algorithm 4.4 98
4.9 Convergence of the nonlinear learner's state-penalty weight Q_l using Algorithm 4.4 99
4.10 Trajectories of the linear learner using off-policy algorithm [73] 100
4.11 Trajectories of the nonlinear learner using off-policy algorithm [73] 100

5.1 Trajectories of the learner and the expert. 122
5.2 (a) Iteration of performance weights Q_i^k; (b) Iteration of NN weights $H_{i_5}^k$ for all $i \in \{1, 2, 3, 4\}$ using Algorithm 5.2. 123
5.3 Trajectories of the learner and the expert. 124

5.4 (a) Iteration of performance weights Q_i^k; (b) Iteration of NN weights $W_{i_5}^k$ for all $i \in \{1, 2, 3, 4\}$ using Algorithm 5.3. 125

5.5 Trajectories of the learner and the expert using RL for (a) example 1 and (b) example 2. 126

Chapter 1: INTRODUCTION

1.1 Distributed Estimation for Multi-agent Systems

Distributed consensus filtering algorithm has been widely studied in theory [11, 48, 81, 82] and practical applications [49, 106]. The work in [81] presents the distributed consensus filtering algorithms to reach an average consensus for all sensor measurements. [82] proposes a Kalman consensus filter (KCF) that has become an effective distributed consensus filter for target estimation, in which each sensor receives information from its neighbors in sensor networks. Whereas, this consensus estimation is not optimal because the cross-covariances between each sensor's estimates are not easy to deal with. Then, [48] proposes an information consensus filter (ICF) algorithm to asymptotically achieve the optimal centralized performance, and it has been applied to camera networks [49]. It has been shown that ICF has better estimate performance in contrast to KCF and generalized Kalman filter (GKCF) [82]. [106] studies the consensus filters in mobile networks. [11] combines information consensus and measurement consensus to develop hybrid consensus filters. In the existing literatures, [81, 82] and [57] focus on the average consensus for a single target, while [11, 48, 49, 106] study the information-weighted consensus.

To the best knowledge, in realistic application areas of sensor networks, there are multiple moving targets. For instance, sea-based sensor networks must estimate the positions of multiple ships. [58] proposes a feature-based algorithm that uses the Kalman filter motion to track multiple objects. [128] introduces an average-weighted consensus protocol for the observer design in multi-target tracking missions. Inspired by these works, our paper develops distributed solution of estimation in multi-target sensor networks to achieve information-weighted consensus.

In realistic scenarios, the mission performance of sensors including detection [19, 20, 80] and measurements [107] can be deeply affected by the limited sensing range. In estimation tasks, when targets enter into the sensing range, sensors directly observe the target states [107]. The measurement topology studied in [48, 49, 81, 82, 92, 106] fail to consider the sensors' limited sensing range. [49] proposes a concept of naive node which is not able to observe the target directly, but

did not develop the measurement model to capture this concept. In [107], the limited sensing range and sensor mobility are considered for mobile robots. However, the mobility was deterministic and known. [41] develops a novel distributed information-weighted consensus filtering algorithm that found the local optimal estimates for targets. In addition, the filter has good convergence performance for the fixed information-flow topology. However, its information flow structure cannot capture features such as random moving targets and limited sensing range. [48, 49, 81, 82, 92, 106] study the fixed information-flow topology, too. As many search and rescue missions require the targets to move with rather random and unknown mobility [15, 66, 115, 117], this work uses random variables to formulate the direct measurement topology for multi-target estimation with random mobility in this paper. Related to this work, [127] uses one random variable to capture if a target can be measured by sensors or not. However, as they do not use neighbors' information, the measurement of sensors can not update once targets cannot be observed, which is adaptive. Therefore, a novel information-flow topology for multi-target measurement is developed in this paper to utilize neighbors' information as well as to reflect the effects of limited sensing range and random target mobility.

To ensure the performance of target estimates, the convergence of the filtering algorithm should be analyzed. The work in [48, 49] proposed Kalman filters without stability and convergence analysis of the algorithms. The works in [32, 90] study Kalman filters in multi-sensor networks but does not provide the convergence analysis. [126] provides convergence analysis under a condition that a bounded partial weight matrix should exists if estimation errors are bounded. In the recent work [110], the covariance matrices converge to unique stabilizing solutions if the sensor network is both detectable and stabilizable. In this discrete-time system, the convergence of filters require the knowledge of global topology. The performance of various Kalman filters are also studied for time-vary systems, stochastic systems and nonlinear systems [22, 33, 99], which need more requirements but are not our goal here.

Robustness plays a pivotal role in the control community [124]. Robustness properties including gain margin (GM) and phase margin (PM) characterize the robustness of the control system. It

is well known that linear quadratic regulators (LQR) have good robustness properties, such as $\pm 60^{\circ}$ phase margin (PM), infinite gain margin (GM), and 50 percent gain reduction tolerance [45,51,95]. The fundamental work shows how gain and phase margins characterize the robust stability of classes of variations in single-input open-loop feedback systems [13]. [93] and [71] then extends Nyquist criterion to multiloop feedback systems but they do not explore the robustness of multi-loop feedback systems. Later, [6] uses the return difference matrix for analyzing the guaranteed robustness margins of single-input LQR. [95] and [51] generalize the results to multivariable cases for LQR. [100] obtains the robustness margins of the discrete time LQR. [84] analyzes robustness margins of dynamic fuzzy control systems. [51] and [16] indicates that the guaranteed robustness margins can be destroyed when the control input weight matrix R is nondiagonal.

It is worth noting that one basic limitation associated with the guaranteed robustness margins for LQR is that the states in feedback loop are assumed to be fully observable and obtainable. As a matter of fact, it is impossible to achieve exact realization for state feedback even by providing enough sensors. Thus, one is motivated to design or use KF [46] to provide nondivergent estimation of the plant whose state is to be estimated. The works of [51] and [25] and have investigated the robustness margins of linear quadratic Gaussian with KF associated to provide state estimates for feedback. [119] designs robust KF that adapts to all admissible uncertain.

With the emergence and popularity of multi-agent systems [35, 56, 111], distributed Kalman filters have been developed to help all agents achieve consensus for estimation. The distributed consensus-based Kalman filter proposed in [82] becomes a popular and efficient filtering for dynamic state estimation. Later, [91], [47], [86,88], [59,60], [70] and [30] develop various distributed filters in different scenarios to improve the performance of filtering estimation algorithms. [29] gives a survey on distributed event-triggered estimation problem over the sensor networks. These algorithms need to be robust and reliable. Then, they can be integrated with feedback control to make the best decision [23, 94, 128] for particular tasks.

The key issue in the overall robustness of integrated systems is the stability of the distributed KF. It is known that the sensor-network based filtering systems are networked systems with the

feedback loop closed through communication networks. However, the communication link gain errors and symmetry breaking [112] of communication matrices. These issues cause the divergence of distributed KF. [112] investigates robust stability conditions with symmetric coupling weights. Motivated by these studies, this paper investigates the robustness margins of distributed KF to apply for these perturbations.

The distributed Kalman-consensus filter (DKCF) in recent work [59] uses information from neighbor estimates of the target or direct target observation to estimate target states. It guarantees that all sensors reach consensus on the estimates of target states.

1.2 Inverse Reinforcement Learning for Multi-agent Systems

To avoid manually specifying the preferences or cost functions of an agent, inverse reinforcement learning (RL) [79] has been proposed to reconstruct the hidden cost functions from its demonstrated behaviors. Inverse RL is widely used in apprentice learning [3,18,63,64,98,104,109], where with observations of an expert's behaviour, a learner uses inverse RL to infer the unknown expert cost functions and obtain the same control policy, thereby performing as well as the expert. As the expert control policy is less available than its behaviors, the learner uses inverse RL to reconstruct the expert cost function and obtain the same policy. Before inverse RL apprentice learning, many different approaches [8, 34, 96] try to learn the direct mapping from states to control inputs or learn the control policy using system identification techniques. However, comparing to inverse RL, these methods directly learn the control policy and are applicable only when the task is to mimic the expert's behavior. But for autonomous driving in real world, the pattern of the traffic circumstance can be different each moment [3]. The cost function is inherently more adaptive because even slight changes in the environment render learned policy unusable and these changes do not affect the transferability of the cost function [7]. It is known that adversarial inputs may affect system stability and imitation performance. However, note that [18, 63, 64, 98, 104] consider the same adversarial disturbance for the learner and the expert or disturbance only for one agent in inverse RL imitation learning, which are strong assumptions.

Given the specified cost function, it is desired to achieve the optimal control and adapt the worst adversary simultaneously, i.e., Nash equilibrium [28, 54, 65] between the control player and the adversarial player. Two-player zero-sum games are generally formulated to solve for the Nash equilibrium by using RL techniques [69, 73]. [69, 73] have developed model-free algorithms to solve Nash equilibrium of two-player zero-sum games.

Optimal control theory assumes that cost functions of an agent are known. However, the intentions or the cost functions of the expert are unknown for the learner in inverse RL problems [3, 18, 63, 64, 79, 98, 104, 109]. The learner thus needs to compute the intentions such that it behaves the same as the expert by using observed states and controls. As such, inverse RL has the same goal as Bayes Learning [67, 103], which computes the epistemic type, namely the cost functions, of an agent using Bayes rule in probability theory. However, inverse RL uses deterministic mathematics and so is more direct to apply than Bayes Learning. [53] studies inverse RL for the expert that has nonlinear cost functions. [37, 64] formulate a Bayesian inverse RL algorithm and obtained a probability distribution of cost functions to learn the right one. [38] proposes efficient inverse RL algorithms for large-scale systems. [14] studies the confidence of the quality of learned policy in inverse RL. Note that [3, 14, 37, 38, 53, 63, 64, 79, 104, 109] study inverse RL apprentice learning where state dynamics is Markov decision process (MDP). [62, 121, 122] study inverse RL for linear differential dynamic systems. [18, 61, 98] investigate inverse RL of the systems that are described by nonlinear differential equations, but they need either system identification or partial system dynamics.

Inverse optimal control [31, 45] technique enables to derive a cost function given states and control inputs. Furthermore, it also guarantees the agent's stability by finding a stabilizing cost to which the control policy is optimal. [44] uses inverse optimal control to reconstruct cost function for continuous-time nonlinear systems. [97] studies inverse optimal control to find cost functions that guarantee the stability of discrete-time systems. Recently, [114] introduces inverse optimal pinning control for trajectory tracking control of complex networks. Note however, these inverse optimal control methods [44, 97, 114] require system dynamics to infer the cost functions.

The relations between inverse RL and inverse optimal control are not fully understood. [52] claims that the two are same, note however, research [1, 31, 45] show that they are definitely not the same thing. Inverse optimal control requires system dynamics while inverse RL does not.

1.3 Contributions

- In Chapter 2, a novel distributed Kalman consensus filter (DKCF) with an information-weighted consensus structure is proposed for random mobile target estimation in continuous time. A novel information flow structure is designed for the measurement of targets that considers the features including the random mobility of targets and limited sensing ranges of sensors, while [41] failes to consider these features in its information flow structure for the distributed Kalman filter. Necessary and sufficient conditions are developed for the convergence of distributed Kalman consensus filters for estimation with randomly moving targets of continuous-time dynamics. This is in contrast to [32, 48, 49, 90] which have not provide rigorous convergence analysis for their proposed filters. The new DKCF in comparison simulation outperforms the existing distributed Kalman filtering algorithm in [82] on the estimation associated with random mobility of targets and limited sensing ranges of sensors.

- In Chapter 3, a new measurement structure is developed that allows each sensor to combine both direct target measurement and indirect neighbor estimates. The robustness margins of the new DKCF based on the new measurement structure are developed for sensor networks. To the best knowledge, the analysis of robustness margins for distributed filters has not been performed yet. The robustness margins, i.e., gain and phase margins are thus derived for DKCF. Both gain and phase margins of DKCF are better than that of the single-agent KF. The distributed transfer function of each agent are derived in two cases. In the first case, the sensor has direct measurements of the target. In the second one, the sensor cannot directly observe the target but only has information from its neighbors. Then, the singular value properties of the return difference matrix of each case are studied. Furthermore, the effects of graph overall coupling strengths on gain and phase margins are analyzed in the two cases,

since the graph overall coupling strengths may vary with the change of the communication graphs in sensor networks. The variation of minimum singular values of return difference matrices under different overall coupling strengths is then developed to study the robustness margin change in the two cases.

- In Chapter 4, first, a new model-based inverse RL algorithm is proposed for Adversarial Apprentice Games. Compared with inverse RL algorithms in [18, 63, 64, 104] for MDPs, the proposed algorithm applies for systems with adversarial disturbance that are described by affine nonlinear differential dynamical equations in continuous-time. The algorithm has two learning stages, an optimal control learning and a second learning based on inverse optimal control. This algorithm is further implemented via NNs. Moreover, the algorithm clarifies that inverse optimal control is solved as a subproblem of inverse RL, which is not clarified in existing inverse RL works [3, 14, 18, 37, 38, 53, 63, 64, 79, 98, 104, 109]. Then, a novel model-free integral inverse RL algorithm based on integral RL is developed to solve the Adversarial Apprentice Games without knowing the dynamics of the expert or the learner. The model-free algorithm is further implemented via NNs. By contrast, [17, 18, 61, 64, 83, 98] provide solutions to infer unknown cost functions but need to know or identify system dynamics. Different adversarial inputs are considered for the learner system and the expert system in the apprentice learning process, while [18, 63, 64, 98, 104] solve inverse RL problems by considering the same disturbance between two agents or disturbance only for one agent. Toward this end, two-player zero-sum games are formulated for both agents in our algorithms. The two-player zero-sum game of the learner is solved as a subproblem in Adversarial Apprentice Game Algorithms. It is shown that the state-penalty weights that the learner learns are not unique. The set of all equivalent state-penalty weights is explicitly characterized, while [3, 14, 18, 37, 38, 53, 63, 64, 98, 104, 109] fails to quantify the set of nonunique cost functions or state-penalty weights.

- In Chapter 5, new inverse RL algorithms are proposed for multi-player non-zero-sum game systems described by differential dynamic equations. This is in contrast to inverse RL in

MDPs [24, 75]. Inverse RL and IOC in different roles are used to solve the multi-player differential dynamic game problem. A new model-based inverse RL algorithm to learn the expert's reward functions. The algorithm has an optimal control learning stage first and a second IOC learning stage. Then, the research develops two new inverse RL algorithms: completely model-free for homogeneous control inputs; and partially model-free for heterogeneous control inputs. This is in contrast to [39, 74] which use IOC to infer the expert's reward functions given known dynamics and matrix design equations.

The publication resulting from this dissertation are listed as follows.

1. Bosen Lian, Wan Yan, Ya Zhang, Mushaung Liu, Frank L. Lewis, and Tianyou Chai, "*Distributed Kalman Consensus Filter for Estimation with Moving Targets*," IEEE Transactions on Cybernetics, to appear, 2020. DOI: 10.1109/TCYB.2020.3029007.

2. Bosen Lian, Frank L. Lewis, Gary Hewer, Katia Estabridis, and Tianyou Chai, "*Robustness Analysis of Distributed Kalman Filter for Estimation in Sensor Networks*," IEEE Transactions on Cybernetics, to appear, 2021. DOI: 10.1109/TCYB.2021.3082157.

3. Bosen Lian, Wenqian Xue, Frank L. Lewis, and Tianyou Chai, "*Inverse Reinforcement Learning for Adversarial Apprentice Games*," IEEE Transactions on Neural Networks and Learning Systems, to appear, 2021. DOI: 10.1109/TNNLS.2021.3114612

4. Bosen Lian, Wenqian Xue, Frank L. Lewis, and Tianyou Chai, "*Inverse Reinforcement Learning for Multi-player Noncooperative Apprentice Games*," submitted to Automatica, 11. 2020.

Chapter 2: DISTRIBUTED KALMAN CONSENSUS FILTER FOR ESTIMATION WITH MOVING TARGETS

2.1 Introduction

Consensus-based distributed Kalman filters for estimation with targets have attracted considerable attention. Most of the existing Kalman filters use the average consensus approach, which tends to have a low convergence speed. They also rarely consider the impacts of limited sensing range and target mobility on the information flow topology. This work addresses these issues by designing a novel distributed Kalman consensus filter (DKCF) with an information-weighted consensus structure for random mobile target estimation in continuous time.

First, a moving target information-flow topology for the measurement of targets is developed based on the sensors' sensing ranges, targets' random mobility, and local information-weighted neighbors. Then, based on the proposed measurement structure, this chapter designs a DCKF for estimation with multiple moving targets that considers information-weighted neighbors and distributed consensus for each sensor node in continuous time. Third, necessary and sufficient conditions about the convergence of the proposed DKCF are developed. Under these conditions, the estimates of all sensors converge to the consensus values. Eventually, simulation and comparative studies show the effectiveness and the superiority of this new DKCF compared to existing KCF.

This organization of this chapter is as follows. Section 2.2 introduces the new distance-based measurement structure and the new DKCF for target estimation. Section 2.3 first formulates a random mobility model of targets to characterize the probability of sensor measurement, and then analyzes the stability and the convergence of the consensus-based filters. Section 2.4 provides comparative simulations to support the analysis. Finally, the conclusions and discussions of further research directions are included in Section 2.5.

2.2 Formulation of Distributed Consensus Filtering Problem for Multiple Moving Targets

This section first provides basics of graph theory, and then develop a new distance-based information flow structure for measuring target and then a distributed Kalman consensus filtering algorithm for multiple moving targets.

2.2.1 Graph Theory in Sensor Networks and Notations

Consider a graph $Gr = (\mathcal{V}, \mathcal{E})$ for a sensor network of N sensors $\mathcal{V} = \{1, 2, \cdots N\}$. The set of edges $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ represents the communication channels. $\mathcal{A} = [a_{ij}] \in \mathbb{R}^{N \times N}$ denotes the associated weighted adjacency matrix, in which a_{ij} is the weight associated with edge (j, i) and denotes the information flow from sensor j to i. It is graphically represented by an arrow with head sensor node i and tail sensor node j, $a_{ij} = 1$ if $(j, i) \in \mathcal{E}$, and otherwise, $a_{ij} = 0$. Denote the set of local neighbors of sensor i as $N_i = \{j : (j, i) \in \mathcal{E}, \forall j \neq i\}$. Define the in-degree of sensor i as $d_i = \sum_{j=1}^{N} a_{ij}$ and the in-degree diagonal matrix $D = \text{diag}\{d_i\} \in \mathbb{R}^{N \times N}$. The graph Laplacian matrix is $L = D - \mathcal{A}$. It is called a directed path from sensor i to sensor j when there is a sequence of successive edges in the form $\{(i, m), (m, s), \ldots, (l, j)\}$. A directed tree is a connected digraph and every node except the root node in this digraph, has an in-degree equaling to one.

Notations: $\mathbb{R}^n$ and $\mathbb{R}^{n \times m}$ denote the n-dimensional Euclidean space and the set of all $n \times m$ matrices, respectively; the superscript *T* represents the matrix transposition; $P > 0$ means that P is a real symmetric and positive definite matrix; $\text{E}[x|y]$ denotes the expectation of x conditionally on y; $\text{P}\{x\}$ denotes the probability of x; $\|\cdot\|$ denotes the Euclidean norm of a vector or a matrix; $\text{diag}\{\cdots\}$ stands for a block diagonal matrix. All matrices are assumed to be compatible for algebraic operations if their dimensions are not explicitly stated.

2.2.2 Distributed Kalman Consensus Filters for Multi-target Sensor Networks

This subsection presents a new distance-based information flow structure for the estimation of

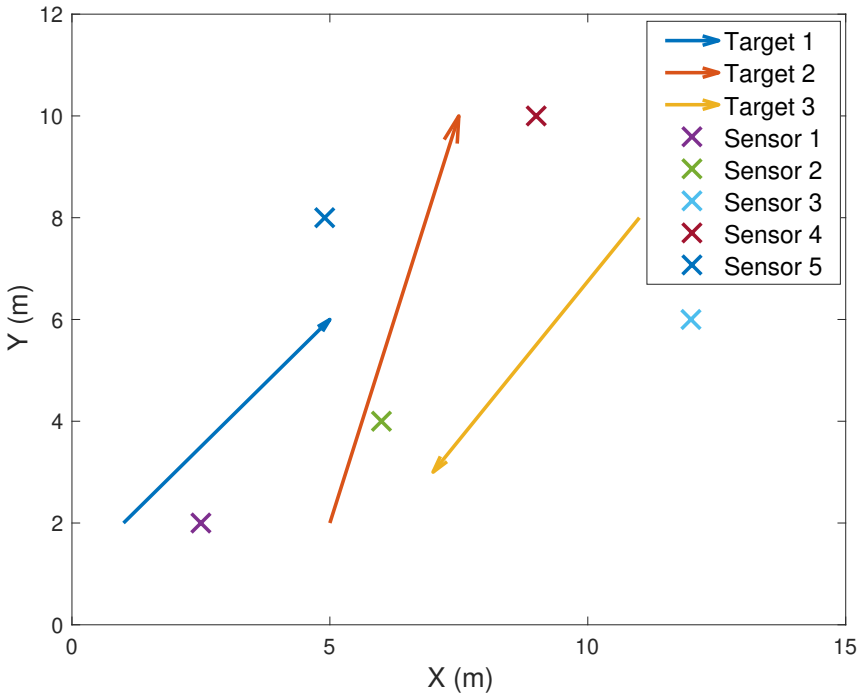


Figure 2.1: Multiple moving targets and field of stationary sensors

multiple moving targets and then develops a novel distributed Kalman consensus filtering algorithm for these moving targets. The motivation of this work is based on the estimation problem of multiple moving targets in field of stationary sensors as shown in Figure 2.1. It is seen that five stationary sensors are deployed in the field to estimate the states of three targets that move around the field.

Consider the target dynamics (2.1), extend the information-flow structure in [41], and apply the standard continuous-time Kalman filter in [118]. In addition, assume that the dynamics of targets are known, and the positions of all sensors in a certain area are fixed such that the adjacency matrix $\mathcal{A} = [a_{ij}]$ is known to all sensors in the topology network.

Consider the target k the CT linear time invariant system as

$$\dot{x}^k = A^k x^k + F^k \omega^k, \tag{2.1}$$

where $x^k \in \mathbb{R}^n$ denotes the state of target k, $k \in \{1,\ 2,\ \cdots,\ U\}$ with the initial condition $x^k(0) \sim$

$(x_0^k,\ P_0^k)$,and ω^k denotes a zero-mean Gaussian process noise, which subjects to $\omega^k \sim (0, W^k)$.

The main goal of this chapter is to design a distributed Kalman consensus filter for estimation with multiple mobile targets. To address the random target mobility and limited sensing range of sensors, a new distance-based information flow structure is designed to estimate the targets, which has the dynamics (2.1). All nodes need to reach a consensus value about the estimation of each target. In addition, a necessary and sufficient condition is given for the convergence of the filter. Now build a novel distance-based information flow structure for multiple targets and the DKCF model by using information from both the targets and the local information-weighted neighbors.

To comprehensively consider limited sensing range of sensors and random mobility of targets, the measurement of sensor i for the moving target k is given by the novel distance-based information-flow topology structure

$$z_i^k = \begin{bmatrix} \lambda_i^k(d_i^k, t)(G_i^k x^k + \mu_i^k) \\ \sum\limits_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k + \omega_{ij}^k) \end{bmatrix}, \tag{2.2}$$

where $G_i^k \in \mathbb{R}^{m_i \times n}$ are distributed observation matrices. $\mu_i^k \in \mathbb{R}^{m_i}$ are zero-mean Gaussian observation noises, which subject to $\mu_i^k \sim (0, R_i^k)$. $\hat{x}_j^k \in \mathbb{R}^n$ and $P_j^k \in \mathbb{R}^{n \times n}$ denote the estimated position and the estimation error covariance of sensor j for target k, respectively. The conditionally expected estimation error covariance is defined as $P_i^k = \mathrm{E}\left\{(\hat{x}_i^k - x^k)(\hat{x}_i^k - x^k)^T | \lambda_1^k, \lambda_2^k, \cdots, \lambda_N^k\right\}$. ω_{ij}^k are communication channel noises with $\Xi_{ij}^k = \mathrm{E}\left\{\omega_{ij}^k (\omega_{ij}^k)^T\right\}$, for any $i =\in \mathcal{V}, j \in N_i$. The work of [12] estimates unknown process and measurement noises in order to improve the target estimates. However, similar to distributed estimation for target tracking in [125, 130], the research focuses on the development of filtering algorithm to solve distributed estimation solution with assuming the known the process and measurement noises for sensor networks.

Here, $\lambda_i^k(d_i^k, t)$ is a distance-based observation index which depends on the relative distance between sensors and true positions of targets. $\lambda_i^k = 1$ if target k enters into the sensing range of sensor i and thus sensor i can directly observe target k, and $\lambda_i^k = 0$, otherwise. The formulation of

λ_i^k and d_i^k are built as

$$\lambda_i^k = \begin{cases} 1, & d_i^k \leq R_i \\ 0, & d_i^k > R_i, \end{cases} \tag{2.3}$$

$$d_i^k = \|x_i^k - p_i\|, \tag{2.4}$$

where p_i and R_i are the position and the sensing range of sensor i, respectively.

This measurement structure (2.2) uses not only the estimates of the neighbors $\hat{x}_j^k, \forall j \in N_i$, but also the direct information from the target k if directly observable by sensor i. In addition, $(P_j^k)^{-1}$ is known as the Fisher information matrix [118] of node j. It increases with the increasing accuracy of accuracy of sensor j's estimates. Then the measurement of node i for target k depends more on the neighbor node j. The information-flow structure (2.2) is called to be information-weighted.

Here, $\mathcal{A}^k \equiv [a_{ij}^k]$ denotes the sensor network topology for the target k, and define Laplacians $L^k = D^k - \mathcal{A}^k$.

Assumption 2.1. *The communication graph in sensor networks is strongly connected.*

Assumption 2.2. *Noises ω^k, ω_{ij}^k and μ_i^k are white. Time signals λ_i^k, ω^k, ω_{ij}^k and μ_i^k are independent.*

Remark 2.1. *Different from [41] in which the estimated target states assume fixed information flow structure, the research here allows the targets to move randomly. It is seen from* (2.1) *the sensing model for targets depend on the distance-based observation index λ_i^k which can be 0 or 1 as shown in* (2.3)*. These values depend on the change of relative distance between sensors and true positions of targets in* (2.4)*. Thus, the measurement structure* (2.2) *is distance-based sensing model.*

The next result presents the novel DKCF for estimation with multiple moving targets based on the distance-based information flow structure (2.2).

Theorem 2.1. *Considering the multiple targets' dynamics* (2.1)*, the information flow structure for moving targets* (2.2) *with communication noises $\omega_{ij}^k = 0$, a DKCF for multiple moving targets is*

given as

$$\begin{aligned}\dot{\hat{x}}_i^k = A^k\hat{x}_i^k - \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1}(G_i^k(x^k - \hat{x}_i^k) + \mu_i^k) \\ + 2d_i^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(P_i^k + P_j^k)]^{-1} \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - \hat{x}_i^k)\end{aligned} \tag{2.5}$$

and the DKCF covariance propagation equation is

$$\begin{aligned}\dot{P}_i^k = A_i^k P_i^k + P_i^k (A_i^k)^T - \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1} G_i^k P_i^k + F^k W^k (F^k)^T \\ - 4\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(P_i^k + P_j^k)]^{-1} \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} P_i^k,\end{aligned} \tag{2.6}$$

where $d_i^k = \sum_{j=1}^{N} a_{ij}^k$, $A_i^k = A^k + \sum_{j=1}^{N} a_{ij}^k I_n$, *and* $i \in \mathcal{V}, j \in N_i$, $k \in \{1,\ 2,\ \cdots,\ U\}$.

Proof. The distance-based information-flow measurement structure (2.2) can be rewritten as

$$\begin{aligned} z_i^k &= \begin{bmatrix} \lambda_i^k G_i^k x^k + \lambda_i^k \mu_i^k \\ \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} x^k + \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - x^k + \omega_{ij}^k) \end{bmatrix} \\ &= H_i^k x^k + v_i^k, \end{aligned} \tag{2.7}$$

where

$$H_i^k = \begin{bmatrix} \lambda_i^k G_i^k \\ \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} \end{bmatrix}, v_i^k = \begin{bmatrix} \lambda_i^k \mu_i^k \\ \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - x^k + \omega_{ij}^k) \end{bmatrix}.$$

Define $\bar{z}_i^k$ as the term consisting of direct target state information

$$\bar{z}_i^k = \lambda_i^k G_i^k x^k + \lambda_i^k \mu_i^k = \bar{H}_i^k x^k + \bar{v}_i^k, \tag{2.8}$$

where $\bar{H}_i^k = \lambda_i^k G_i^k$ denotes the direct information in terms of the distance-based observation index λ_i^k and $\bar{v}_i^k = \lambda_i^k v_i^k$ is corresponding direct noise of sensor i for the target k. Denote $\tilde{z}_i^k$ as the

indirect information due to the information-weighted estimates of the neighbors of sensor i

$$\begin{aligned}\tilde{z}_i^k &= \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k + \omega_{ij}^k) \\ &= \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} x^k + \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - x^k + \omega_{ij}^k) \\ &= \tilde{H}_i^k x^k + \tilde{v}_i^k, \end{aligned} \tag{2.9}$$

where $\tilde{H}_i^k = \sum_{j=1}^{N} a_i^k (P_j^k)^{-1}$ is indirect information of neighbors, and $\tilde{v}_i^k = \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - x^k + \omega_{ij}^k)$ is corresponding indirect noise.

Considering the multiple targets (2.1) with direct target information (2.8) and indirect information weighted neighbors (2.9), the distributed Kalman filter for state estimates are given as

$$\begin{aligned}\dot{\hat{x}}_i^k &= A^k \hat{x}_i^k + K_i^k (z_i^k - H_i^k \hat{x}_i^k) \\ &= A^k \hat{x}_i^k + \bar{K}_i^k (\bar{z}_i^k - \bar{H}_i^k \hat{x}_i^k) + \tilde{K}_i^k (\tilde{z}_i^k - \tilde{H}_i^k \hat{x}_i^k) \\ &= A^k \hat{x}_i^k + \lambda_i^k \bar{K}_i^k (G_i^k (x^k - \hat{x}_i^k) + \mu_i^k) + \tilde{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - \hat{x}_i^k + \omega_{ij}^k), \end{aligned} \tag{2.10}$$

where $K_i^k = (\bar{K}_i^k,\ \tilde{K}_i^k)$, in addition $\bar{K}_i^k$ and $\tilde{K}_i^k$ are gains of directed information from targets and indirect information from neighbors, respectively.

Rewrite (2.10) as

$$\begin{aligned}\dot{\hat{x}}_i^k = A_i^k \hat{x}_i^k &+ \lambda_i^k \bar{K}_i^k (G_i^k (x^k - \hat{x}_i^k) + \mu_i^k) \\ &+ \tilde{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - \hat{x}_i^k + \omega_{ij}^k) - \sum_{j=1}^{N} a_{ij}^k \hat{x}_i^k, \end{aligned} \tag{2.11}$$

where $A_i^k = A^k + \sum_{j=1}^{N} a_{ij}^k I_n$.

Denote $\tilde{x}_i^k = x^k - \hat{x}_i^k$ as sensor i's distributed state estimation error of for the target k. Then,

based on (2.1) and (2.11), the distributed state error dynamics are

$$\begin{aligned}\dot{\tilde{x}}_i^k &= A^k x^k + F^k \omega^k - A_i^k \hat{x}_i^k - \lambda_i^k \bar{K}_i^k (G_i^k (x^k - \hat{x}_i^k) + \mu_i^k) \\ &\quad - \bar{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} (\hat{x}_j^k - \hat{x}_i^k + \omega_{ij}^k) + \sum_{j=1}^{N} a_{ij}^k \hat{x}_i^k \\ &= A_i^k \tilde{x}_i^k - \lambda_i^k \bar{K}_i^k G_i^k \tilde{x}_i^k - \lambda_i^k \bar{K}_i^k \mu_i^k \\ &\quad + \tilde{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} (\tilde{x}_j^k - \tilde{x}_i^k + \omega_{ij}^k) + F^k \omega^k. \end{aligned} \tag{2.12}$$

Rewrite (2.12) with $\omega_{ij}^k = 0$ as the following form

$$\dot{\tilde{x}}_i^k = M_i^k \tilde{x}_i^k + N_i^k, \tag{2.13}$$

where

$$M_i^k = A_i^k - \lambda_i^k \bar{K}_i^k G_i^k - \tilde{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} \tag{2.14}$$

and

$$N_i^k = -\lambda_i^k \bar{K}_i^k \mu_i^k + \tilde{K}_i^k \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} \tilde{x}_j^k - d_i^k \tilde{x}_i^k + F^k \omega^k. \tag{2.15}$$

As Assumption 2.2 states, time signals λ_i^k, ω^k and μ_i^k are independent, so their cross correlations are zero. However, $\hat{x}_j^k$ and $\tilde{x}_i^k$ are correlated. Let $\gamma_i^k = \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} \tilde{x}_j^k$. To quantify these connections, define the following covariances

$$S^k = E\{\omega^k (\omega^k)^T\}, \tag{2.16}$$

$$\overline{T}_i^k = E\{\mu_i^k (\mu_i^k)^T\} = R_i^k, \tag{2.17}$$

$$\tilde{T}_i^k = E\{\gamma_i^k (\gamma_i^k)^T\} = \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}. \tag{2.18}$$

Build a discrete-time model for (2.13) with (2.14) and (2.15) and follow Euler discretization. Then, one has

$$\tilde{x}_i^k(h+1) = e^{M_i^k T}\tilde{x}_i^k(h) + \int_0^T e^{M_i^k(T-\tau)}d\tau N_i^k(h), \tag{2.19}$$

where $(h+1)T = t$ and T is the sampling period of discretization.

It is known from Taylor series that one has

$$\begin{aligned} e^{M_i^k T} &= I + TM_i^k + \frac{(TM_i^k)^2}{2!} + \frac{(TM_i^k)^3}{3!} + \dots \\ &= I + TM_i^k + O((TM_i^k)^2), \end{aligned} \tag{2.20}$$

where $O((TM_i^k)^2)$ represents the terms of higher order than the 1th degree.

Similar, one writes Taylor series for $\int_0^T e^{M_i^k(T-\tau)}d\tau$ as

$$\begin{aligned} \int_0^T e^{M_i^k(T-\tau)}d\tau &= \int_0^T (I + M_i^k(T-\tau) + O\{[M_i^k(T-\tau)]^2\}d\tau \\ &= TI + O^{'}((TM_i^k)^2), \end{aligned} \tag{2.21}$$

where the notation $O^{'}(T)$ in integral operator represents the terms of equal and higher order than 2th degree of TM_i^k.

Based on the above Taylor series (2.20) and (2.21), one can rewrite (2.19) as

$$\begin{aligned} \tilde{x}_i^k(h+1) &= (I + TM_i^k)\tilde{x}_i^k(h) + (TI + O^{'}((TM_i^k)^2))N_i^k(h) + O((TM_i^k)^2)\tilde{x}_i^k(h) \\ &= (I + TM_i^k)\tilde{x}_i^k(h) + TN_i^k(h) + O((TM_i^k)^2)\tilde{x}_i^k(h) + O^{'}((TM_i^k)^2)N_i^k(h). \end{aligned} \tag{2.22}$$

Substitute (2.22) into $P_i^k(h+1) = \mathrm{E}\{\tilde{x}_i^k(h+1)[\tilde{x}_i^k(h+1)]^T|\lambda_1^k, \cdots, \lambda_N^k\}$. Then, one obtains

$$\begin{aligned}
P_i^k(h+1) = &(I+TM_i^k)P_i^k(h)(I+TM_i^k)^T + T^2\mathrm{E}\{N_i^k(N_i^k)^T\} \\
&+ \mathrm{E}\{T(I+TM_i^k)\tilde{x}_i^k(N_i^k)^T\} \\
&+ \mathrm{E}\{TN_i^k(\tilde{x}_i^k)^T(I+TM_i^T)\} \\
&+ (I+TM_i^k)x_i^k(h)[O(\cdot^2)\tilde{x}_i^k(h) + O^{'}(\cdot^2)N_i^k(h)]^T \\
&+ TN_i^k[O(\cdot^2)\tilde{x}_i^k(h) + O^{'}(\cdot^2)N_i^k(h)]^T \\
&+ [O(\cdot^2)\tilde{x}_i^k(h) + O^{'}(\cdot^2)N_i^k(h)][(I+TM_i^k)\tilde{x}_i^k(h) \\
&+ TN_i^k(h) + O(\cdot^2)\tilde{x}_i^k(h) + O^{'}(\cdot^2)N_i^k(h)]^T, \qquad (2.23)
\end{aligned}$$

where $O(\cdot^2)$ and $O^{'}(\cdot^2)$ denote $O((TM_i^k)^2)$ and $O^{'}((TM_i^k)^2)$, respectively. The number 2 inside of $O(\cdot^2)$ and $O^{'}(\cdot^2)$ denotes the least degree.

Based on the definition of differentiation, one obtains the $\dot{P}_i^k(t)$ to be

$$\begin{aligned}
\dot{P}_i^k(t) &= \lim_{T\to 0}\frac{P_i^k(h+1) - P_i^k(h)}{T} \qquad (2.24) \\
&= \lim_{T\to 0}\Big\{M_i^k(t)P_i^k(t) + P_i^k(t)(M_i^k(t))^T + T\mathrm{E}[N_i^k(t)(N_i^k(\tau))^T] \\
&\quad + \mathrm{E}[\tilde{x}_i^k(t)(N_i^k(t))^T] + \mathrm{E}[N_i^k(t)(\tilde{x}_i^k(t))^T] \\
&\quad + N_i^k[O(\cdot^2)\tilde{x}_i^k(t) + O^{'}(\cdot^2)N_i^k(t)]^T \\
&\quad + [O(T(M_i^k)^2)\tilde{x}_i^k(t) + O^{'}(T(M_i^k)^2)N_i^k(t)][(I+TM_i^k)\tilde{x}_i^k(t) \\
&\quad + TN_i^k(t) + O(T(M_i^k)^2)\tilde{x}_i^k(t) + O^{'}(T(M_i^k)^2)N_i^k(t)]^T\Big\}.
\end{aligned}$$

Note that as $T \to 0$, the last three terms of right side of (2.24) tend to be zeros. Thus, one has

$$\begin{aligned}
\dot{P}_i^k(t) = &M_i^k(t)P_i^k(t) + P_i^k(t)(M_i^k(t))^T \\
&+ \mathrm{E}[\tilde{x}_i^k(t)(N_i^k(t))^T] + \mathrm{E}[N_i^k(t)(\tilde{x}_i^k(t))^T] \\
&+ \lim_{T\to 0}\Big\{T\mathrm{E}[N_i^k(t)(N_i^k(\tau))^T]\Big\} \qquad (2.25)
\end{aligned}$$

After discretization sampling and back to continuous-time domain, write

$$\begin{aligned}&\mathrm{E}\{N_i^k(t)(N_i^k(\tau))^T\}\\&=\mathrm{E}\left\{\eta_i^k(\eta_i^k)^T\right\}+F^k(W^k/T)(F^k)^T\delta(t-\tau)+\lambda_i^k\bar{K}_i^k(\bar{T}_i^k/T)(\bar{K}_i^k)^T\delta(t-\tau),\end{aligned}\tag{2.26}$$

where $\eta_i^k=\tilde{K}_i^k\sum_{j=1}^{N}a_{ij}^k(P_j^k)^{-1}\tilde{x}_j^k-d_i^k\tilde{x}_i^k$. The second and third term are covariance of process and measurement noise, respectively. They are white noise, which means $\delta(t-\tau)=0$ when $t\neq\tau$.

When $T\to 0$, one derives (2.25) as

$$\begin{aligned}\dot{P}_i^k=&M_i^kP_i^k+P_i^k(M_i^k)^T+\lambda_i^k\bar{K}_i^k\bar{T}_i^k(\bar{K}_i^k)^T+F^kW^k(F^k)^T\\&+\mathrm{E}\{\tilde{x}_i^k(N_i^k)^T\}+\mathrm{E}\{N_i^k(\tilde{x}_i^k)^T\}.\end{aligned}\tag{2.27}$$

Assume $\tilde{K}_i^k=(\tilde{K}_i^k)^T$ and let

$$\mathrm{E}\{\tilde{x}_i^k(N_i^k)^T\}+\mathrm{E}\{N_i^k(\tilde{x}_i^k)^T\}=0.\tag{2.28}$$

Then, one obtains suboptimal form for indirect information gain $\tilde{K}_i^k$ as

$$\tilde{K}_i^k=2\sum_{j=1}^{N}a_{ij}^kP_i^k[\sum_{j=1}^{N}a_{ij}^k(P_j^k)^{-1}(P_i^k+P_j^k)]^{-1}.\tag{2.29}$$

Then, rewrite (2.27) as

$$\begin{aligned}\dot{P}_i^k=&(A_i^k-\lambda_i^k\bar{K}_i^k\bar{H}_i-\tilde{K}_i^k\sum_{j=1}^{N}a_{ij}^k(P_j^k)^{-1})P_i^k\\&+P_i^k(A_i^k-\lambda_i^k\bar{K}_i^k\bar{H}_i-\tilde{K}_i^k\sum_{j=1}^{N}a_{ij}^k(P_j^k)^{-1})^T\\&+\lambda_i^k\bar{K}_i^k\bar{T}_i^k(\bar{K}_i^k)^T+F^kW^k(F^k)^T.\end{aligned}\tag{2.30}$$

In order to solve the gain $\bar{K}_i^k$, the value of λ_i^k should be discussed. Because of the random

mobility of targets and limited sensing range, sensors may not be able to observe the targetsâ dynamics all the time.

1) If $\lambda_i^k = 0$, as it is shown in (2.12), there will be no item of $\bar{K}_i^k \bar{H}_i$ for the direct update information from target x^k. Thus, one has

$$\bar{K}_i^k = 0. \tag{2.31}$$

2) If $\lambda_i^k = 1$, the filter updates. One then computes $\partial Tr(\dot{P}_i^k)/\partial \bar{K}_i^k = 0$ to obtain gain $\bar{K}_i^k$ as

$$2\bar{K}_i^k \bar{T}_i^k - 2P_i^k (G_i^k)^T = 0 \tag{2.32}$$

$$\bar{K}_i^k = P_i^k (G_i^{k)} T (R_i^k)^{-1}. \tag{2.33}$$

Compensively considering two situations (2.31) and (2.33), one has optimal direct gains $\bar{K}_i^k$ as

$$\bar{K}_i^k = \lambda_i^k P_i^k (G_i^{k)} T (R_i^k)^{-1}. \tag{2.34}$$

Substituting (2.29) and (2.34) into (2.11) and (2.30), one derives DKCF (2.5) and (2.6). □

It is seen that indirect information gain in (2.29) is not optimal while the direct target measurement gain is optimal. Thus, in general, the covariance propagation (2.6) is not optimal. However, the indirect information gain is given by removing the complicated cross correlation between $\tilde{x}_i^k$ and the estimates $\tilde{x}_j^k$, $j \in N_i$ from its neighbors.

Remark 2.2. *The difficulties associated with the estimation for random mobile targets are two main aspects. The first is how to build the connection between random moving targets and the measurement. This work designs the parameter λ_i^k to determine whether the target k is observed by sensor i. Considering the sensing range R_i of sensor i, λ_i^k is determined. Thus, with formulations from* (2.1) *to* (2.4)*, sensor networks are able to measure random moving target. The second difficulty is to derive the propagation of the estimation error covariance* (2.6)*. The reason is that when $\lambda_i^k = 0$, one cannot do the derivative for* (2.30) *to seek gain $\bar{K}_i^k$. Therefore, one needs to*

seek this gain in two cases as shown from (2.31) *to* (2.33)*. Then, this works combine two situations as shown in* (2.34)*.*

Remark 2.3. *The comparison of the DKCF* (2.5) *and* (2.6) *with the Kalman filter in [41] show two differences. First, the Kalman filter in [41] is extended to multiple targets in a distributed way. Second,* (2.5) *uses the information from targets directly when targets move into the sensing range of sensors, i.e.,* $\lambda_i^k = 1$*. When the targets move, they may be out of the range for which the sensors can observe.* λ_i^k *is modulated by target location and hence is distance-based. However, [41] did not consider the varying observations of sensors for the target.*

Remark 2.4. *If there exists communication noise* ω_{ij}^k *between sensor* i *and neighbor sensor* j *for the estimation of target* k*, the DKCF* (2.5) *and* (2.6) *can be derived to be*

$$\begin{aligned}\dot{\hat{x}}_i^k = {} & A^k\hat{x}_i^k + \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1}(G_i^k(x^k - \hat{x}_i^k) + \mu_i^k) \\ & + 2\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(P_i^k + P_j^k)]^{-1} \\ & \times \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(\hat{x}_j^k - \hat{x}_i^k + \omega_{ij}^k),\end{aligned} \tag{2.35}$$

$$\begin{aligned}\dot{P}_i^k = {} & A_i^k P_i^K + P_i^K (A_i^k)^T - \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1} G_i^k P_i^k + F^k W^k (F^k)^T \\ & - 4\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(P_i^k + P_j^k)]^{-1} \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} P_i^k \\ & + 4\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} \\ & \times \Xi_{ij}^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} P_i^k.\end{aligned} \tag{2.36}$$

2.3 Convergence Analysis

This section analyzes and proves the convergence of the distributed Kalman consensus filter

estimation method.

To accomplish this, more details are needed about the statistics of $\lambda_i^k(t)$ in (2.2). In order to analyze the convergence of the proposed DKCF on random moving targets, this work simply assumes the independent stochastic process of $\lambda_i^k(t)$ at every time instant. The time domain is decomposed into a finite set of m disjoint random intervals $[t_s,\, t_{s+1})$ for $0 \leq s \leq m-1$. In each small interval, assume independent stochastic process for $\lambda_i^k(t)$ with $\lambda_i^k = 1$ or 0. Then

$$\mathrm{P}\left\{\lambda_i^k(t) = 1\right\} = p_i^k, \quad \mathrm{E}\left\{\lambda_i^k(t)\right\} = p_i^k \tag{2.37}$$

where $t \in [t_s,\, t_{s+1})$. Additionally, $\lambda_i^k(t)$ and $\lambda_j^k(t)$ are independent, such that $\mathrm{Cov}(\lambda_i^k(t), \lambda_j^k(t)) = 0$. $\lambda_i^k(t_1)$ and $\lambda_i^k(t_2)$ are independent such that $\mathrm{Cov}(\lambda_i^k(t_1), \lambda_i^k(t_2)) = 0$. t_1 and t_2 are in different intervals.

$\lambda_i^k(t)$ is time-varying and modeled by the random moving targets. To capture random mobility, stochastic models such as random mobility models, including Random Waypoint, and Smooth Turn are widely used. Interested readers please refer to paper [115] for detailed descriptions of these random models. The distributions of $\lambda_i^k(t)$ can be obtained by examining the stationary node distributions of these models. For instance, as the Random Waypoint and Smooth Turn have uniform stationary distributions, the stationary probability of $\lambda_i^k(t) = 1$ can be obtained as the ratio between sensor i's coverage area divided by the target k's whole exploration area. For the Random Waypoint, the stationary node distribution is non-uniform, and in particular bell-shaped due to the boundary effects. The stationary probability of $\lambda_i^k(t) = 1$ in this case can be obtained from approximated expressions developed in e.g., [36]. For each different target, because of the varying $\lambda_i^k(t)$, Laplacian matrix L^k differs for any $k \in \{1,\, 2,\, \cdots,\, U\}$. Then, the following results are needed and extended for the convergence analysis.

Due to Assumption 2.1, one has $\mathrm{rank}(L^k) = N - 1$ [56, 89], i.e., the eigenvalue of L^k which equals zero is not repeated. Therefore, under Assumption 2.1, the L^k is a singular M-matrix.

Lemma 2.1. *[89] Given the singular M-matrix L^k (Laplacian matrix), there exists a positive*

vector $q^k = \begin{bmatrix} q_1^k & q_2^k & \cdots & q_N^k \end{bmatrix}^T$ *with* $q_i^k > 0$*, for* $\forall i \in \mathcal{V}$ *such that* $q^k L^k \geq 0$*, where* $k \in \{1,\ 2,\ \cdots,\ U\}$.

Lemma 2.2. *[41] Give the singular* M*-matrix* L^k *(Laplacian matrix). Define the vector* $q^k = \begin{bmatrix} q_1^k & q_2^k & \cdots & q_N^k \end{bmatrix}^T$*, where* $q_i^k > 0$ *in Lemma 2.1. Then,* $q_i^k \sum_{j=1}^{N} a_{ij}^k \geq \sum_{j=1}^{N} q_j a_{ij}^k > 0$ *for* $\forall i \in \mathcal{V}$ *and* $k \in \{1,\ 2,\ \cdots,\ U\}$.

Based on these constructions, the result of the convergence of the DKCF for state estimates of moving targets is obtained now.

Definition 2.1. *Define the estimation error as* $\tilde{x}_i(t) = x(t) - \hat{x}_i(t), i \in \mathcal{V}$*. The dynamics of* $\tilde{x}_i$ *is called uniformly ultimately bounded (UUB) in mean square with ultimate bound* c_i*, if there exist positive constants* r_i*,* $T(r_i, \epsilon_i)$ *and* c_i*, such that for any given constant* $\epsilon_i > 0$*,* $\|\tilde{x}_i(0)\| \geq r_i$ *implies that* $E\{\|\tilde{x}_i\|^2\} \leq c_i + \epsilon_i$ *for all* $t > T(r_i, \epsilon_i)$.

Theorem 2.2. ***(Proof of convergence to consensus of DKCF for estimation of moving targets).*** *Under Assumption 2.1, consider the information-flow structure* (2.2) *and the DKCF in Theorem 2.1 for estimates of multiple moving targets. Suppose noises exist. Then, if and only if the pair* (A^k, G^k) *with* $G^k = \begin{bmatrix} (p_1^k G_1^k)^T & (p_2^k G_2^k)^T & \cdots & (p_N^k G_N^k)^T \end{bmatrix}^T$ *is observable, the state estimates reach to consensus in a small bound in the mean square, i.e., the distributed state estimation errors of each sensor defined as* $\tilde{x}_i^k(t) = x^k(t) - \hat{x}_i^k(t)$ *are UUB in the mean square as* $t \to \infty$*,* $\forall\ i,\ k$.

Proof. The dynamics of targets are

$$\dot{x}^k = A^k x^k + F^k \omega^k. \tag{2.38}$$

The DKCF for the state estimation of moving targets with noises is given by (2.35). Since the graph Gr is strongly connected, there exists a directed path from any sensor i to any other sensor j, where $i, j \in \mathcal{V}$. Then, the local estimation error dynamics and estimation covariance are obtained

respectively, as

$$\begin{aligned}\dot{\tilde{x}}_i^k &= \dot{x}^k - \dot{\hat{x}}_i^k \\ &= (A^k - \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1} G_i^k)\tilde{x}_i^k + \lambda_i^k P_i^k G_i^k (R_i^k)^{-1}\mu_i^k + 2d_i^k P_i^k Q_i^k(\tilde{x}_j^k - \tilde{x}_i^k + \omega_{ij}^k) + F^k\omega^k,\end{aligned} \tag{2.39}$$

$$\begin{aligned}\dot{P}_i^k = {} & A_i^k P_i^k + P_i^k (A_i^k)^T - \lambda_i^k P_i^k (G_i^k)^T (R_i^k)^{-1} G_i^k P_i^k - 4\sum_{j=1}^{N} a_{ij}^k P_i^k Q_i^k P_i^k + F^k W^k (F^k)^T \\ & + 4\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} \Xi_{ij}^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} P_i^k.\end{aligned} \tag{2.40}$$

Select a Lyapunov function candidate as

$$V(\tilde{X}^k(\lambda_i^k, t)) = \mathrm{E}\left[\sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} \tilde{x}_i^k\right], \tag{2.41}$$

where q_i^k is defined in Lemma 2.1. Given the observability condition that the pair $(A^k,\ G^k)$ is observable when $t \to \infty$, $P_i^k(t)$ converge to positive definite constant matrices. Then, $\mathrm{diag}\{(P_i^k)^{-1}\} > 0$ such that $V(\tilde{X}^k) > 0$ holds for any $\tilde{X}^k = \begin{bmatrix}\tilde{x}_1^k & \tilde{x}_2^k & \dots & \tilde{x}_N^k\end{bmatrix}^T \neq 0$.

Taking the derivative of $V(\tilde{X}^k)$ yields

$$\dot{V}(\tilde{X}^k(\lambda_i^k, t)) = E\{\sum_{i=1}^{N} q_i^k (\dot{\tilde{x}}_i^k)^T (P_i^k)^{-1} \tilde{x}_i^k + \sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (\dot{P}_i^k)^{-1} \tilde{x}_i^k + \sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} \dot{\tilde{x}}_i^k\}. \tag{2.42}$$

Substituting (2.39) and (2.40) into (2.42) leads to

$$\begin{aligned}\dot{V}(\tilde{X}^k(\lambda_i^k,t)) = E\{\sum_{i=1}^{N} q_i^k[A_i^k - \lambda_i^k P_i^k \Sigma_i^k + 2d_i^k P_i^k Q_i^k(\tilde{x}_j^k - \tilde{x}_i^k) \\ - d_i^k \tilde{x}_i^k]^T (P_i^k)^{-1} \tilde{x}_i^k\} - E\{\sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1}[A_i^k P_i^k \\ + P_i^k (A_i^k)^T - \lambda_i^k P_i^k \Sigma_i^k P_i^k - 4d_i^k P_i^k Q_i^k P_i^k + F^k W^k (F^k)^T] \\ \times (P_i^k)^{-1} \tilde{x}_i^k\} + E\{\sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1}[A_i^k - \lambda_i^k P_i^k \Sigma_i^k \\ + 2d_i^k P_i^k Q_i^k(\tilde{x}_j^k - \tilde{x}_i^k) - d_i^k \tilde{x}_i^k]\} - \sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T \Theta_i^k \tilde{x}_i^k \\ + \sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T \phi_i^k + \sum_{i=1}^{N} q_i^k (\phi^k)^T \tilde{x}_i^k, \end{aligned} \tag{2.43}$$

where

$$\begin{aligned}\Sigma_i^k &= p_i^k (G_i^k)^T (R_i^k)^{-1} G_i^k, \\ Q_i^k &= [\sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1}(P_i^k + P_j^k)]^{-1} \sum_{j=1}^{N} a_{ij}^k (P_j^k)^{-1} \\ \Theta_i^k &= 4\sum_{j=1}^{N} a_{ij}^k P_i^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} \Xi_{ij}^k [\sum_{j=1}^{N} a_{ij}^k (P_i^k + P_j^k)]^{-1} P_i^k, \\ \phi_i^k &= \lambda_i^k G_i^k (R_i^k)^{-1} \mu_i^k + 2d_i^k Q_i^k \omega_{ij}^k + F^k W^k. \end{aligned}$$

With simplification, one has

$$\begin{aligned}
\dot{V}(\tilde{X}^k(\lambda_i^k,t)) = &-\sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\Sigma_i^k\tilde{x}_i^k - \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T F^k W^k (F^k)^T\tilde{x}_i^k \\
&+ E\{\sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T(P_i^k)^{-1}[2d_i^k P_i^k Q_i^k(\tilde{x}_i^k-\tilde{x}_j^k) - d_i^k\tilde{x}_i^k]\} \\
&+ E\{\sum_{i=1}^{N} q_i^k[2d_i^k P_i^k Q_i^k(\tilde{x}_i^k-\tilde{x}_j^k) - d_i^k\tilde{x}_i^k]^T(P_i^k)^{-1}\tilde{x}_i^k\} \\
&- E\{\sum_{i=1}^{N} 4q_i^k(\tilde{x}_i^k)^T Q_i^k\tilde{x}_i^k\} - \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\Theta_i^k\tilde{x}_i^k \\
&+ \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\phi^k + \sum_{i=1}^{N} q_i^k(\phi^k)^T\tilde{x}_i^k \\
= &-\sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\Sigma_i^k\tilde{x}_i^k - \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T F^k W^k (F^k)^T\tilde{x}_i^k \\
&- \sum_{i=1}^{N} 2q_i^k d_i^k(\tilde{x}_i^k)^T Q_i^k\tilde{x}_j^k - \sum_{i=1}^{N} q_i^k d_i^k(\tilde{x}_i^k)^T(P_i^k)^{-1}\tilde{x}_i^k \\
&- \sum_{i=1}^{N} 2q_i^k d_i^k(\tilde{x}_j^k)^T Q_i^k\tilde{x}_i^k - \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\Theta_i^k\tilde{x}_i^k \\
&+ \sum_{i=1}^{N} q_i^k(\tilde{x}_i^k)^T\phi^k + \sum_{i=1}^{N} q_i^k(\phi^k)^T\tilde{x}_i^k \\
&- \sum_{i=1}^{N} q_i^k d_i^k(\tilde{x}_i^k)^T(P_i^k)^{-1}\tilde{x}_i^k. \qquad (2.44)
\end{aligned}$$

According to Lemma 2.2, the last term of (2.44) becomes

$$\begin{aligned}
&-\sum_{i=1}^{N} q_i^k \sum_{j=1}^{N} a_{ij}^k(\tilde{x}_i^k)^T(P_i^k)^{-1}\tilde{x}_i^k \qquad (2.45) \\
&\le -\sum_{i=1}^{N}\sum_{j=1}^{N}(q_j^k a_{ji}^k)(\tilde{x}_i^k)^T(P_i^k)^{-1}\tilde{x}_i^k \\
&= \sum_{i=1}^{N} q_i^k \sum_{j=1}^{N} a_{ij}^k(\tilde{x}_j^k)^T(P_j^k)^{-1}\tilde{x}_j^k.
\end{aligned}$$

Because $0 < 2Q_i^k P_i^k \leq I$ and $0 < 2Q_i^k P_j^k \leq I$, one has

$$
\begin{aligned}
&-\sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_i^k)^T Q_i^k \tilde{x}_j^k - \sum_{i=1}^{N} q_i^k d_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} \tilde{x}_i^k \\
&-\sum_{i=1}^{N} q_i^k d_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} \tilde{x}_i^k - \sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_j^k)^T Q_i^k \tilde{x}_i^k \\
&\leq -\sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_i^k)^T Q_i^k \tilde{x}_j^k - \sum_{i=1}^{N} q_i^k d_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} \tilde{x}_i^k \\
&\quad -\sum_{i=1}^{N} q_i^k d_i^k (\tilde{x}_j^k)^T (P_j^k)^{-1} \tilde{x}_j^k - \sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_j^k)^T Q_i^k \tilde{x}_i^k \\
&\leq -\sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_i^k)^T Q_i^k \tilde{x}_j^k - \sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_i^k)^T Q_i^k \tilde{x}_i^k \\
&\quad -\sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_j^k)^T Q_i^k \tilde{x}_j^k - \sum_{i=1}^{N} 2q_i^k d_i^k (\tilde{x}_j^k)^T Q_i^k \tilde{x}_i^k \\
&\leq -\sum_{i=1,j=1}^{N} 2q_i^k a_{ij}^k (\tilde{x}_i^k + \tilde{x}_j^k)^T Q_i^k (\tilde{x}_i^k + \tilde{x}_j^k) \\
&\leq 0.
\end{aligned} \tag{2.46}
$$

Finally, one has

$$
\begin{aligned}
\dot{V}(\tilde{X}^k) \leq & -\sum_{i=1}^{N} q_i^k (\tilde{x}_i^k)^T (P_i^k)^{-1} F^k W^k (F^k)^T (P_i^k)^{-1} \tilde{x}_i^k \\
& -\sum_{i=1}^{N} p_i^k q_i^k (\tilde{x}_i^k)^T (G_i^k)^T (R_i^k)^{-1} G_i^k \tilde{x}_i^k \\
& -\sum_{i=1,j=1}^{N} 2q_i^k a_{ij}^k (\tilde{x}_i^k + \tilde{x}_j^k)^T Q_i^k (\tilde{x}_i^k + \tilde{x}_j^k) \\
& -\lambda_{min}(\Theta^k) \|\tilde{X}^k\|^2 + \|\Phi^k\| \|\tilde{X}^k\| \\
\leq & -\lambda_{min}(\Theta^k) \|\tilde{X}^k\|^2 + \|\Phi^k\| \|\tilde{X}^k\|,
\end{aligned} \tag{2.47}
$$

where $\Theta^k = \text{diag}\{q_i^k \Theta_i^k\}$ and $\Phi^k = [q_1^k \phi^k, \cdots, q_N^k \phi^k]^T$.

It is seen that when $t \to \infty$, if $\|\tilde{X}^k\| \geq \|\Phi^k\| / \lambda_{min}(\Theta^k)$, one has $\dot{V}(\tilde{X}^k) \leq 0$. According to

UUB, after a lapsed time t_e, estimation errors trajectory eventually reaches a bounded neighborhood of its equilibrium point $\tilde{X}_e^k$. In addition, note that the approximation errors go to zeros due to the limit on the sampling period. Thus, it has no influence to this bound. Furthermore, as $t \to \infty$, one has diag$\{(P_i^k)^{-1}\} > 0$ such that $V(\tilde{X}^k) > 0$ Therefore, (2.41) is a Lyapunov function and estimation error dynamics of $\tilde{x}_i^k$ (see (2.39)) is UUB in the mean square. It implies that the errors between estimates and the target states reach a bounded neighborhood, i.e., $\tilde{x}_i^k(t) = x^k(t) - \hat{x}_i^k(t)$ is UUB.

The above proof provides the sufficiency of the condition for the convergence of estimation errors. For necessity, it is noted that when $\dot{V}(\tilde{X}^k) \leq 0$ and (2.41) is a Lyapunov function, one concludes that diag$\{(P_i^k)^{-1}\} > 0$ such that the pair (A^k, G^k) with $G^k = \begin{bmatrix}(p_1^k G_1^k)^T & (\mathrm{p}_2^k G_2^k)^T & \cdots & (p_N^k G_N^k)^T\end{bmatrix}^T$ is observable. Thus, the condition that the pair (A^k, G^k) is observable is a necessary and sufficient condition for the convergence of estimation errors by using the DKCF. □

Remark 2.5. *Note that estimation errors are UUB and the bound of the errors depend on the process, observation and communication noises (see* (2.47)*). When these noises are smaller, the estimation errors will be reduced. It is seen that the estimation errors $\tilde{x}_i^k$ converge to zeros when $t \to \infty$ if noises are zeros, which implies that the estimated states via DKCF converge to the moving target states.*

Remark 2.6. *The computation complexity will not scale a lot with the increasing numbers of targets and sensors. A slight increase is possible. First, the DKCF is first order. Second, this work considers no coupling of targets and therefore the estimation of each sensor for each target is distributed and independent. Third, the value of stochastic variable λ_i^k is either 0 or 1, which brings less computational complexity.*

Remark 2.7. *As it is shown in Theorem 2.2, the convergence condition is under Assumption 2.1, if only if the pair (A^k, G^k) with $G^k = \begin{bmatrix}(p_1^k G_1^k)^T & (p_2^k G_2^k)^T & \cdots & (p_N^k G_N^k)^T\end{bmatrix}^T$ is observable. This means that some sensor, such as sensor i for target k, is allowed to have probability 0, i.e., $p_i^k = 0$ to observe the target. The convergence can still be guaranteed in this case. Note that the DKCF (6) and (7) together not only use direct information from targets, but also estimations from all the*

sensor nodes. If target k is not in the sensing range of sensor i (i.e., not being the root node), $\lambda_i^k = 0$, and the information all comes from its neighbors. As no direct information from targets is used, and the estimation error in (2.6) *and the uncertainties about the estimation can be large. However, as information from neighbors is still available, at convergence one has $P_i^k \to \delta$ where δ is some constant and $\hat{x}_i^k \to \hat{x}_j^k$.*

Theorem 2.2 is based on the strongly connected topology. Next, this work weakens the condition on the communication topology. Then the following corollaries hold.

Corollary 2.1. ***(Necessary and Sufficient Condition).*** *Suppose that 1) Assumption 2.1 does not hold, and 2) (A^k, G_i^k) is observable for the target k where $k = \{1, 2, \cdots, O\}$, and $i \in \mathcal{V}$. Then, the estimation of sensors by using DKCF for multiple moving targets converges to consensus if only if $\sum_{i=1}^{N} p_i^k > 0$, namely, there exists at least one root node which has a nonzero observation probability.*

Corollary 2.2. ***(Sufficient Condition).*** *If Assumptions 2.1-2.2 are that 1) the graph is not strongly connected and 2) $r_1, r_2, \cdots, r_m$ are root nodes among all sensors. Then, the estimation converges to the consensus, if the pair (A^k, G_r^k) with $G_r^k = \begin{bmatrix} (p_{r1}^k G_{r_1}^k)^T & (p_{r2}^k G_{r_2}^k)^T & \cdots & (p_{rm}^k G_{r_m}^k)^T \end{bmatrix}^T$ is observable.*

2.4 Simulation Studies

Now, several illustrative examples are presented to demonstrate that the new DKCF for estimation with multiple moving targets in Theorem 2.1 have good performance. And comparison simulations show DKCF has better convergence than the Kalman consensus filter (KCF) in [82].

Sensors are stationary deployed in the area and the communication topology is given in Figure 5.5.

2.4.1 Performance of the DKCF

The effectiveness of the proposed DKCF is verified in this subsection. This work assumes three moving vehicles as targets in certain area. Consider the dynamics and sensing model be linear time

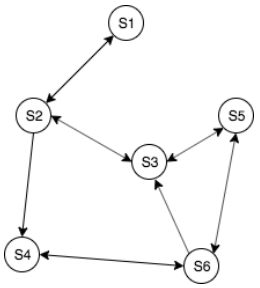


Figure 2.2: Communication topology of the sensor network

invariant, which has $x^1, x^2, x^3, x^4 \in \mathbb{R}^4$. The standard system dynamic of two degrees-of-freedom vehicle model of Ford Taurus [85] is

$$A = \begin{bmatrix} \frac{ac_1 - bc_2}{I_z u} & \frac{a^2 c_1 + b^2 c_2}{I_z u} & 0 & 0 \\ \frac{a^2 c_1 + b^2 c_2}{mu} & \frac{bc_2 - ac_1}{mu} & 0 & 0 \\ \frac{c_1 + c_2}{mu} & \frac{ac_1 - bc_2}{mu} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The state is $x = \begin{bmatrix} \dot{v} & \dot{r} & \dot{s} & s \end{bmatrix}^T$, where v, r, $\dot{s}$ and s denote the vehicle lateral velocity along body, the yaw rate of the vehicle body in vertical axis, the vehicle lateral position and velocity, respectively. Several physical parameters of vehicles for system dynamics A_1 A_2 and A_3 are presented in Table 2.1.

Table 2.1: Taurus nominal vehicle parameters

Parameters	Vehicle 1	Vehicle 2	Vehicle 3
Vehicle mass m	1820kg	1500kg	2000kg
Moment of inertia I_z	2922kg-m^2	2002kg-m^2	1930kg-m^2
Forward speed u	15.50m/s	27.08m/s	35.00m/s
Brake system time τ_{abs}	0.5s	0.5s	0.5s
CG to front wheel a	1.07m	1.07m	1.07m
CG to rear wheel b	1.62m	1.62m	1.62m
Front cornering coefficient c_1	-555lb/deg	-500lb/deg	-492lb/deg
Rear cornering coefficient c_2	-366lb/deg	-330lb/deg	-325lb/deg

The other parameters such as the process, measurement and communication noises for each target are selected as $\omega^k \sim (0, W)$, where $W = \begin{bmatrix} 2 & 1 \end{bmatrix}^T$. Additionally $G_i^k = I_2$, $\mu_i^k \sim (0, 0.5\mathbf{1}_2)$, $\omega_{ij}^k \sim (0, 0.5\mathbf{1}_2)$, $F^k = I_2$, $S^k = \text{diag}\{2, 1\}$, $R_i^k = 0.25I_2$, $\Xi_{ij}^k = 0.5I_2$, $k = \{1, 2, 3\}$ and $i, j = \{1, 2, 3, 4, 5, 6\}$, where $\mathbf{1}_n \in \mathbb{R}^n$ represents a column vector with all ones. I_n represents a $n \times n$ identity matrix.

To study the estimation of DKCF for random moving targets, this work can use random models that are provided in Section 2.3, to obtain different observation probabilities of sensors for the three targets. Any observation probability value from 0-1 is possible, based on different settings of the sensing range and exploration area.

The diagonal element $P_{(i)22}^k$ of the covariance matrix of sensor i for target k represents the covariance matrix to show the estimation evaluation. Here, $P_i^k = \text{E}\{(\tilde{x}_i^k)^T \tilde{x}_i^k | \lambda_i^k\}$. This work sets two observation Prob (probability) Schemes as follows.

- **Prob Scheme 1.** $p_1^k = 0$, $p_2^k = 0.1$, $p_3^k = 0.5$, $p_4^k = 0.7$, $p_5^k = 0.9$, $p_6^k = 1$ for all k.
- **Prob Scheme 2.** $p_1^k = 0.2$, $p_2^k = 0.1$, $p_3^k = 0.5$, $p_4^k = 0.7$, $p_5^k = 0.9$, $p_6^k = 1$ for all k.

The average covariance values $P_{(i)22} = \frac{1}{Q}\sum_{k=1}^{Q} P_{(i)22}^k$ of sensor i for all targets are are shown in Figure 2.3 under Prob Scheme 1. The initials of S1 with Prob 0 denote that sensor 1 has 0 observation probability for three targets. The relevant covariance values under the Prob Scheme 2 are presented in the Figure 2.4. Note that the distributed estimation error covariance of sensor 1 converges when $p_1^k = 0$ while this covariance value is larger than other that of other probabilities. This is because the estimation filter in this work uses the weighted information from neighbors. In addition, the covariance value decreases with the increase of observation probabilities by comparing Figure 2.3 and Figure 2.4.

Figure 2.5 gives the average estimated errors of center of mass lateral position for all three targets given the Prob Scheme 1. With the zero observation probability of sensor 1, the estimation errors still converge. In addition, the average estimate errors quickly reach a small bound. With the increase of observation probabilities of the sensors, the filter updates with more precise

information. Thus, there will be fewer errors. These results are consistent with the theoretical analysis.

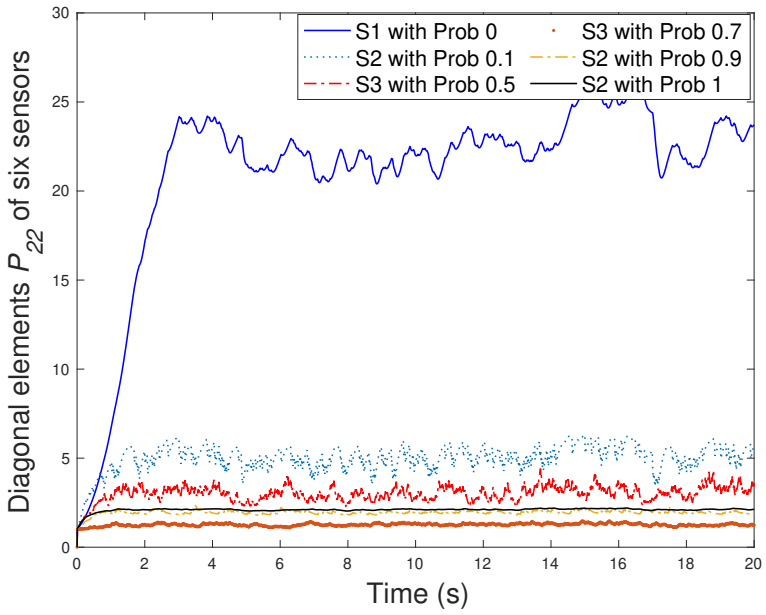


Figure 2.3: The average covariance value P_{22} of different sensors for all targets under Prob Scheme 1 by DKCF

2.4.2 Comparison Between KCF in [82] and DKCF

In this subsection, the DKCF in Theorem 2.1 is compared with the KCF in [82] for multiple targets. Two cases below are considered. The convergence rates of the DKCF and the KCF are compared.

The parameters for the distributed KCF are selected as $G_1 = I_2$, $G_2 = I_2$, and $G_3 = I_2$ for those sensors who can observe the targets' states. $\theta_i^k = \alpha/(1 + ||P_i^k||_F)$ with $\alpha = 0.05$ for $i = 1, 2, 3$ as stated in [81]. The other parameters are selected as the same as those given in Section 2.4.1.

Case 1. Considering Prob Scheme 1 for sensors in Section 2.4.1, the convergence of KCF for the estimation is illustrated in Figure 2.6. In this case, $p_1^k = 0$ which means sensor 1 is a naive node [82]. It can be seen that the estimation error of all sensors converges within a small bound except for that of sensor 1. It has a converge for oscillation. The reasons are two aspects. First,

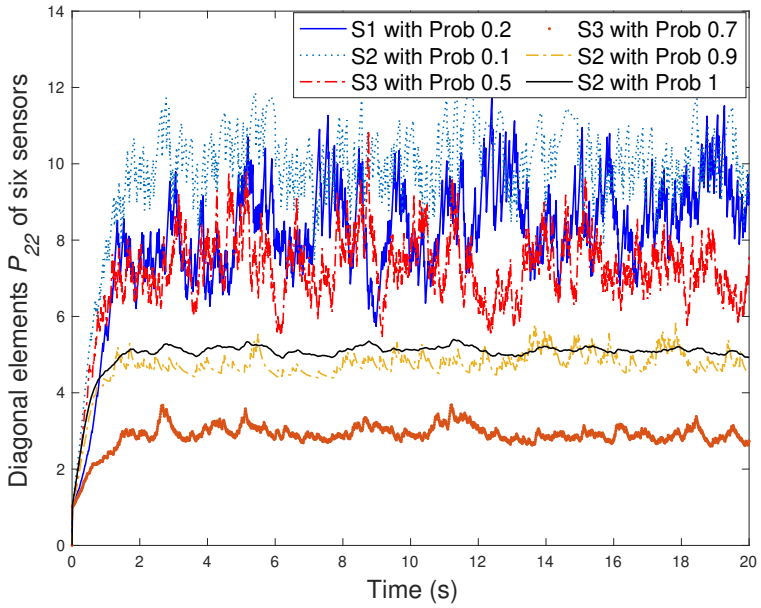


Figure 2.4: The average covariance value P_{22} of different sensors for all targets under Prob Scheme 2 by DKCF

considering the parameters in Table I, the target states are oscillating. Second, the KCF algorithm does not observe the targets directly, nor does it utilize the weighted information from neighbors $(P_j^k)^{-1}$ to reduce the effect of inaccurate estimates.

Case 2. To study the convergence rate between DKCF and KCF, all nodes are assumed to have the same nonzero observation probability for three targets, e.g., $p_i^k = 0.2$ for $\forall i = \{1, 2, \cdots 6\}$ and $k = \{1, 2, 3\}$. Figure 2.7 and Figure 2.8 show the average estimation error of each sensor for all targets using the two filters. As it is shown, the two filters converge within the sampling time, whereas the proposed DKCF has a faster convergence speed and a smaller error bound compared with KCF. In order to show the further effectiveness of the DKCF, Figure 2.9 gives a larger and sparser sensor network configuration with 12 sensors. The convergence of estimation errors is shown in Figure 2.10.

Another evaluation criteria is given by the Mean Square Error (MSE) of the tracking state error. The MSE of all state estimates of each sensor for each target in time interval $[0, T]$ is defined as $MSE_i^k = \frac{1}{T}\int_0^T (x^k(t) - \hat{x}_i^k(t))^2 dt$. The MSE of all sensors for target k is defined as $MSE^k = \frac{1}{N}\sum_{i=1}^N MSE_i^k$. The MSE of all sensors for all targets is defined as $MSE = \frac{1}{M}\sum_{k=1}^M MSE^k$.

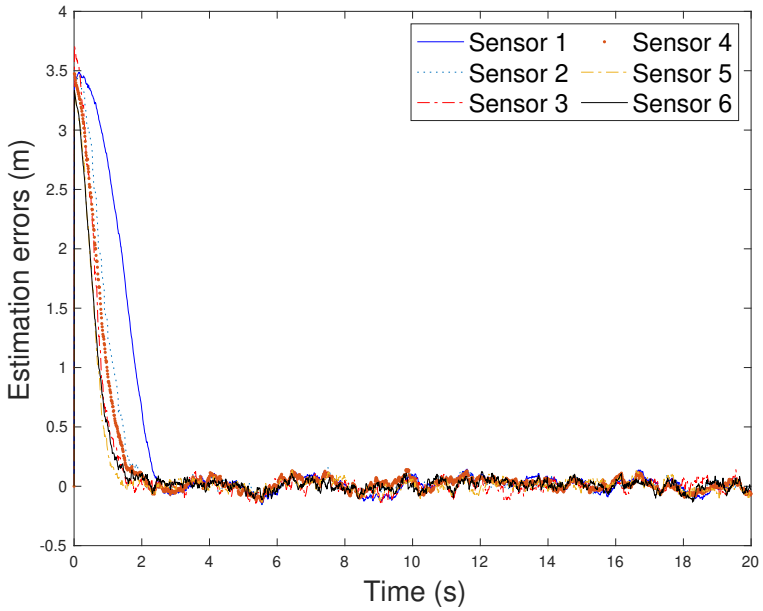


Figure 2.5: The average estimation errors of different sensors for all targets by DKCF in case 1

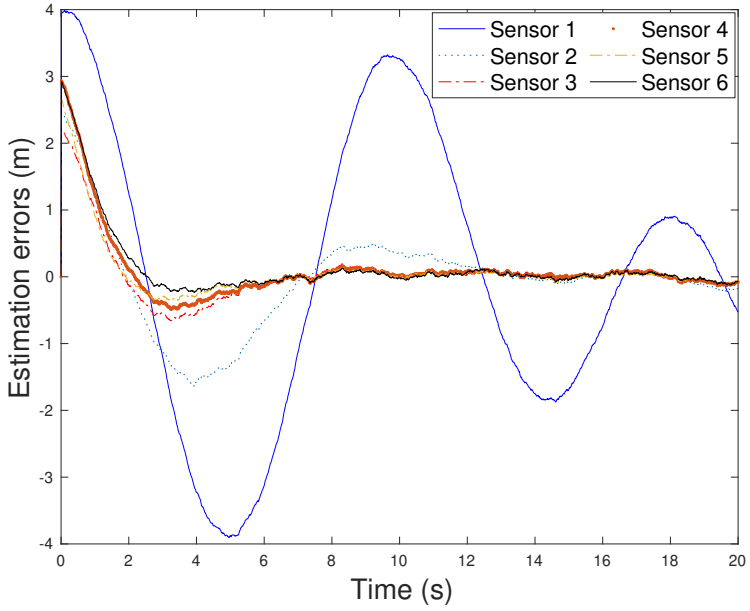


Figure 2.6: The average estimation errors of different sensors for all targets by KCF in case 1

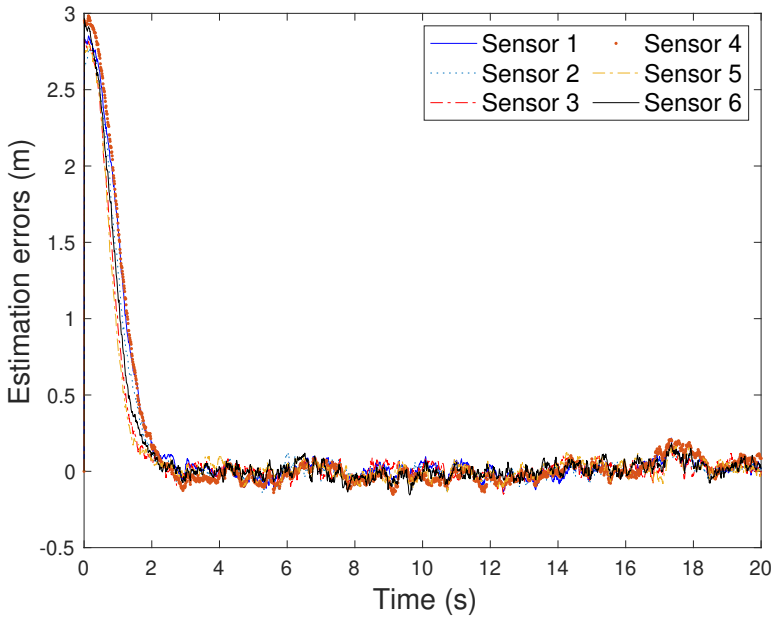


Figure 2.7: The average estimation errors of different sensors for all targets by DKCF in case 2

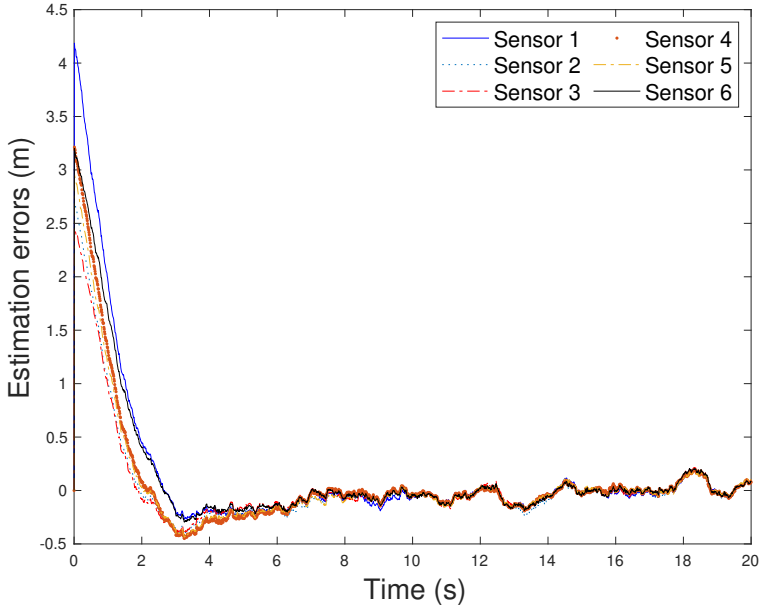


Figure 2.8: The average estimation errors of different sensors for all targets by KCF in case 2

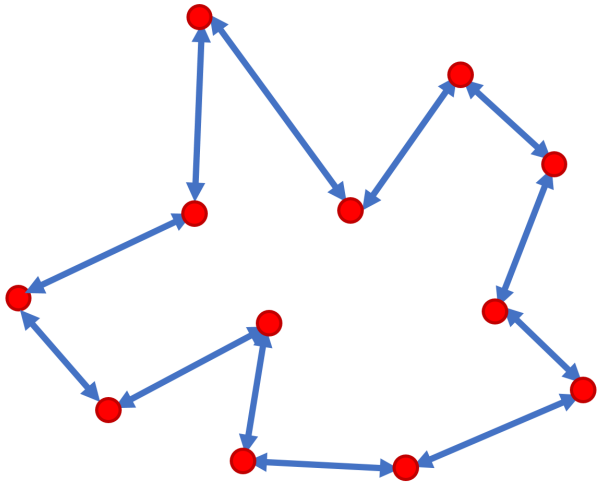


Figure 2.9: Larger and sparser sensor network configuration

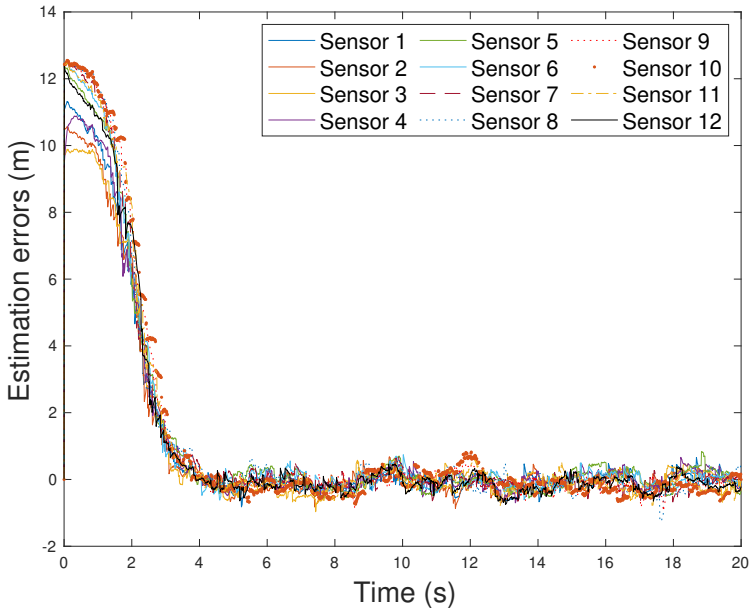


Figure 2.10: The average estimation errors of different sensor for all targets by DKCF for sensor network in Figure 2.9

Figure 2.11 illustrates the change of MSE with different probabilities for the two algorithms. It is clear that the MSE by using DKCF is always smaller than that by using KCF. Note that when all targets always stay in the sensing range of all sensors, i.e., $p_i^k = 1$ for KCF, the MSE value is about 0.0718. This is the MSE using the DKCF algorithm with $p_i^k = 0.5$.

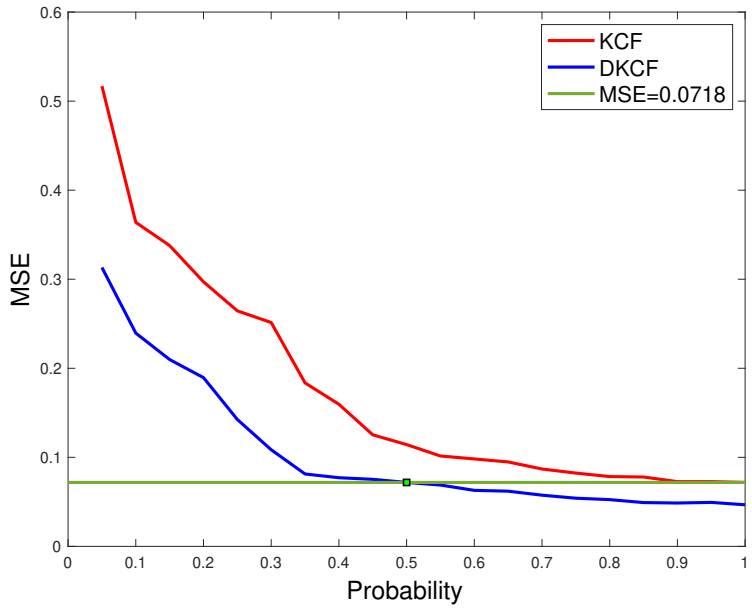


Figure 2.11: The variation of MSE by KCF and DKCF with different probabilities

Therefore, with the distance-based information-flow measurement structure and the weighted-information structure, the proposed DKCF for the estimates of random moving targets has improved convergence performance in contrast to the KCF in [82].

2.5 Conclusion

This chapter studies the distributed Kalman consensus filter (DKCF) for the state estimates of moving targets in continuous-time dynamics. Considering the limited sensing range of sensors and random mobility of targets, a novel distance-based information-flow measurement structure is proposed for the DKCF. This structure consists of two parts including the observation with random moving targets and information-weighted estimates of neighbors. Necessary and sufficient conditions are given so as to guarantee the convergence of DKCF by Lyapunov method. It is

further demonstrated in the simulation studies that the proposed DKCF has a good convergence performance.

Chapter 3: ROBUSTNESS MARGINS OF DISTRIBUTED KALMAN CONSENSUS FILTER

3.1 Introduction

Motivated by the guaranteed stability margins of linear quadratic regulators (LQR) and standard Kalman filter (KF) in frequency domain, this chapter extends these results to the distributed Kalman-consensus filter (DKCF) for distributed estimation in sensor networks. In the first part, this chapter studies the robustness margins of DKCF in two cases, one of which is based on the direct target observation while the other uses estimates from neighbor sensors in the network. The loop transfer functions of the two cases are established, and gain margin (GM) and phase margin (PM) robustness results are derived for both. The robustness margins of DKCF are improved compared to the single-agent KF, i.e., GM of $(\frac{1}{2}, \infty)$ and PM of $\pm 60^\circ$. In the second part, as communication topology varies in sensor networks, graph overall coupling strengths change. This chapter also analyzes the correlation between overall coupling strengths and the robustness margins of DKCF.

This chapter is organized as follows. Section 3.2 introduces the standard KF and outline the GM and PM results of KF. Section 3.3 introduces the new measurement structure and DKCF with convergence proof. Then, robustness margins of DKCF are studied in two cases. Section 3.4 gives simulation results. The conclusion is given in Section 3.5.

Notations. $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ denote the sets of n and $n \times m$ dimensional real vectors and real matrices, respectively. A^{-T} denotes the inverse of the complex conjugate transpose of the complex matrix A. The determinant of the matrix A is denoted by $\det(A)$. $\text{diag}\{\cdots\}$ stands for a block-diagonal matrix. $\|\cdot\|$ defines the induced 2-norm of a matrix. $A > 0$ and $A \geq 0$ means that the matrix A is positive definite and semi-definite, respectively. $\underline{\sigma}(A)$ and $\bar{\sigma}(A)$ denote the minimum and maximum singular values of a matrix A, respectively. $\text{E}\{\}$ denotes the expectation operator. The real part of the complex number $a(s)$ is denoted by $\text{Re}(a(s))$. The initial CRHP denotes closed right half plane.

3.2 Kalman Filter Robustness Margins

Robust margins guarantee the gain margins and phase margins of a feedback control system. They are key to demonstrate the robustness of the systems when systems have issues of gain perturbations or inaccurate phase. Robustness has been well studied for the LQR in the works of [26, 51, 95, 124]. This section reviews robustness margins including GM and PM for the KF.

Consider a target with dynamics

$$\dot{x} = Ax + Fw, \tag{3.1}$$

where $x \in \mathbb{R}^n$ is the state. $A \in \mathbb{R}^{n\times n}$ and $F \in \mathbb{R}^{n\times m}$ are constant matrices. $w \in \mathbb{R}^m$ is the zero-mean Gaussian noise source with power spectral density matrix S. Note that A is not necessarily stable.

To estimate the target state, consider the measurement of sensor for the target

$$z = Hx + \mu, \tag{3.2}$$

where $z \in \mathbb{R}^m$ is the measurement. $H \in \mathbb{R}^{m\times n}$ is the observation matrix. $\mu \in \mathbb{R}^m$ is the zero-mean Gaussian noise source with the diagonal spectral density matrix $R > 0$.

The continuous-time optimal Kalman filtering estimation algorithm, KF [55], is given by

$$\dot{\hat{x}} = Ax + L(z - H\hat{x}), \tag{3.3}$$

$$L = PH^T R^{-1}, \tag{3.4}$$

$$0 = AP + PA^T + Q - PH^T R^{-1} HP, \tag{3.5}$$

where $\hat{x}$ is the estimate of the target state and L is the KF gain. $Q = FSF^T \geq 0$, $R > 0$. (A, H) and $(A, Q^{1/2})$ are detectable.

Define the state estimation error as $\tilde{x} = x - \hat{x}$. Then, one has

$$\dot{\tilde{x}} = \dot{x} - \dot{\hat{x}} = A(x - \hat{x}) + L(z - \hat{z}) = (A - LH)\tilde{x}. \tag{3.6}$$

It is seen that as long as the KF gain L is selected so that the closed-loop KF observer matrix $A - LH$ is asymptotically stable, the estimation error $\tilde{x}$ goes to zero asymptotically [54].

Compared with state variable feedback for LQR design, KF (3.3)-(3.5) is the dual by replacing the parameters (A, B, K) with (A^T, H^T, L^T), where B is the control input dynamics and K is the control gain in LQR. If the pair (A^T, H^T) is reachable, then L^T can be selected to make $A - LH$ asymptotically stable. Since the robustness results of LQR are obtained in Safonov [95] and Lehtomaki [51], and the KF is the dual of the LQR, the robustness results of KF are obtainable. Based on the KF (3.3)-(3.5), the loop transfer function of input v to output $\hat{z}$ is

$$G(s) = \frac{\hat{z}(s)}{v(s)} = H(sI - A)^{-1}L. \tag{3.7}$$

Note that $G(s)$ in (3.7) is the dual of the transfer function of LQR.

The next definition and results are needed to derive the main KF robustness margins in Theorem 3.2.

Definition 3.1. (A, H, L) *is called a state-space realization for* $G(s)$ *in* (3.7) *when linear systems are given as* (3.3)-(3.5)*.*

The robustness is the extent to which the elements of the loop transfer function matrix $G(s)$ can vary from their nominal design values without affecting the stability of the system [51, 95]. It is characterized by GM and PM. The standard LQR has infinite GM, 50 percent gain reduction tolerance and 60° of PM. Now, this work introduces the robustness GM and PM results of standard KF (3.3)-(3.5). These results follow from the literature [16, 51].

Theorem 3.1. ***(Minimum singular value bound of KF).*** *Given Kalman filter algebraic Riccati*

equation (ARE)

$$AP + PA^T - PH^T R^{-1} HP + Q = 0, \tag{3.8}$$

where $Q \geq 0$ *and* $R > 0$*, one has the return difference relation*

$$[I + G(s)]R[I + G(s)]^{-T} = R + F(s), \quad s = \alpha\omega \in \alpha\mathbb{R}. \tag{3.9}$$

Furthermore, one has

$$\underline{\sigma}[I + \bar{G}(s)] \geq 1, \tag{3.10}$$

where $G(s) = H(sI-A)^{-1}PH^T R^{-1}$*,* $\bar{G}(s) = R^{-1/2}G(s)R^{1/2}$ *and* $F(s) = H(sI-A)^{-1}Q(-sI-A^T)^{-1}H^T$.

Proof. The proof is similar to that in [51] and [16]. □

When a perturbation $B(s)$ is added to the KF (3.3)-(3.5), the loop transfer function becomes

$$\begin{aligned} \tilde{G}(s) = G(s)B(s) &= H(sI - A)^{-1}LB(s) \\ &= \tilde{H}(sI - \tilde{A})^{-1}\tilde{L}, \end{aligned} \tag{3.11}$$

where $B(s)$ is assumed as a diagonal matrix, such that the perturbations in the multiple loops do not interact between each other. This is followed by the assumption of perturbations for robustness margins for LQR in Safonov [95], and Lehtomaki [51]. In addition, the state-space realization (A, H, L) becomes the realization $(\tilde{A}, \tilde{H}, \tilde{L})$. Thus, KF (3.3)-(3.5) becomes

$$\dot{\hat{x}} = \tilde{A}x + \tilde{L}(z - \tilde{H}\hat{x}), \tag{3.12}$$

$$\tilde{L} = P\tilde{H}^T R^{-1}, \tag{3.13}$$

$$0 = \tilde{A}P + P\tilde{A}^T + Q - P\tilde{H}^T R^{-1}\tilde{H}P. \tag{3.14}$$

Theorem 3.2. ***(Robustness Margins of KF).*** *Suppose Theorem 3.1 and the following conditions hold:*

a) det($sI - A$) and det($sI - \tilde{A}$) have the same number of CRHP zeros, and if det($\alpha\omega_0 I - \tilde{A}$)=0, det($\alpha\omega_0 I - A$)=0,

b) $RB(s) + B^{-T}(s)R \geq R$, $s = \alpha\omega \in \alpha\mathbb{R}$.

Then one has:

1) The perturbed system $\tilde{G}(s)$ *is closed-loop asymptotically stable,*

2) The optimal KF (3.3)-(3.5) *has a guaranteed GM of* $(\frac{1}{2}, \infty)$*, that is 50 percent gain reduction and the infinite gain margin,*

3) The optimal KF (3.3)-(3.5) *has a guaranteed PM of* $\pm 60^{\circ}$.

Proof. The proof is similar to that in Lehtomaki [51]. □

3.3 Robustness Analysis of Distributed Kalman-Consensus Filter

This section first introduces some basics of graph theory for multi-sensor networks and present a novel distributed Kalman consensus filter (DKCF) for continuous-time systems.

A new measurement structure is developed that allows each sensor to combine both direct target measurement and indirect neighbor estimates. DKCF robustness margins are then studied. Distributed Kalman filters are designed to achieve the stability and the consensus of the target estimation in sensor networks. Robustness margins are key to show the robustness of the consensus. They are rarely studied in distributed estimation for sensor networks. It further shows in this section that robustness margins of DKCF improve in contrast to robustness margins of standard KF.

3.3.1 Graph Theory

Please see graph theory knowledge in Chapter 2.2.1.

3.3.2 Distributed Kalman Consensus Filter

Give the sensor graph $\mathcal{G}$. In a sensor network, consider the novel measurement structure of sensor i, $i \in \mathcal{V}$ for the target dynamics (3.1) as

$$z_i = \begin{bmatrix} g_i(H_i x + \mu_i) \\ \sum_{j=1}^{N} a_{ij} P_j^{-1}(\hat{x}_j + \omega_{ij}) \end{bmatrix}, \ i \in \mathcal{V}, \tag{3.15}$$

where $z_i \in \mathbb{R}^{m \times n}$ is the measurement of sensor i for the target x. The pinning gain $g_i \in \{0, 1\}$ represents the communication link between the target and sensor i. If $g_i = 1$, sensor i has direct measurements of the target. Otherwise, it does not. $H_i \in \mathbb{R}^{m \times n}$ is the distributed direct observation matrix. μ_i is a zero-mean Gaussian observation noise with diagonal covariance matrix $R_i > 0$. Define $\hat{x}_i, \hat{x}_j \in \mathbb{R}^n$ as the estimates of sensor i and j for the target, respectively. Define $P_i = \mathrm{E}\{(x - \hat{x}_i)^T (x - \hat{x}_i)\} \in \mathbb{R}^{n \times n}$ as the covariance value of estimation error of sensor i. ω_{ij} is the communication channel noise between sensors i and j, $j \in N_i$. The estimate $\hat{x}_j$ of sensor i's neighbors is weighted by P_j^{-1} through the coupling strength a_{ij}.

Remark 3.1. *The measurement structure in* (3.15) *has two parts: the direct measurement of the target states with gain g_i and the indirect measurement from neighbor estimates based on weighted trust P_j^{-1}. This measurement structure enables to avoid naive nodes [47] that do not have direct or indirect measurement of the target.*

Remark 3.2. *A major advantage of the measurement structure* (3.15) *of the DKCF this work derives based on* (3.15)*, is that the target state does not need to be fully observable by each sensor. Instead, a milder form of collective observability by all the sensors is required, as given in the following Definition 3.2.*

Definition 3.2. *The target state $x(t)$ in* (3.1) *is said to be collectively observable by all the sensors in* (3.15) *when the pair (A, H) with $H = (H_1^T, H_2^T, \cdots H_N^T)^T$ is observable.*

Based on the measurement structure (3.15) and the distributed Kalman filters in chapter 2, the DKCF is developed in Algorithm 3.1 below for target estimation in multi-sensor network.

Algorithm 3.1 Distributed Kalman Consensus Filter Algorithm

1 **Initialization:** $P_i(0), \hat{x}_i(0)$, where $i \in \mathcal{V}$.

2 **Distributed Covariance Update:**

$$\dot{P}_i = A_i P_i + P_i A_i^T + Q - g_i^2 P_i H_i^T R_i^{-1} H_i P_i - P_i \sum_{j=1}^{N} a_{ij}(P_j^{-1} - P_i^{-1}) P_i, \quad \forall i \in \mathcal{V}. \quad (3.16)$$

3 **Distributed Trust Consensus Update:**

$$\dot{\hat{x}}_i = A\hat{x}_i + g_i P_i H_i^T R_i^{-1}(H_i(x - \hat{x}_i) + \mu_i) + P_i \sum_{j=1}^{N} a_{ij} P_j^{-1}(\hat{x}_j - \hat{x}_i), \quad \forall i \in \mathcal{V}, \quad (3.17)$$

where $Q = FSF^T \geq 0$, $S > 0$ and diagonal matrix $R_i > 0$, $i \in \mathcal{V}$.

The next result shows that by using DKCF (3.16)-(3.17), the state estimate $\hat{x}_i, \forall i \in \mathcal{V}$, achieves consensus with the target state under the condition of collective observability.

Theorem 3.3. ***(Consensus of DKCF in Algorithm 3.1).*** *Consider the measurement structure* (3.15)*, the target* (3.1) *and the associated DKCF in Algorithm 3.1. Suppose that 1) the sensor communication graph $\mathcal{G}$ is strongly connected and 2) the target in DKCF* (3.16)-(3.17) *is collectively observable by all sensors. The estimation error $\tilde{x}_i(t) = x(t) - \hat{x}_i(t) \in \mathbb{R}^n$, $\forall i \in \mathcal{V}$ is UUB in the mean square as $t \to \infty$ with observation noise μ_i assumed to be zero.*

Proof. With (3.1) and (3.17), the dynamic of estimation error $\tilde{x}_i(t)$ is given by

$$\begin{aligned} \dot{\tilde{x}}_i &= \dot{x}_i - \dot{\hat{x}}_i \\ &= A\tilde{x}_i - g_i P_i H_i^T R_i^{-1} H_i \tilde{x}_i + P_i \sum_{j=1}^{N} a_{ij} P_j^{-1}(\tilde{x}_j - \tilde{x}_i) + Fw. \end{aligned} \quad (3.18)$$

Define a Lyapunov function as $V(\tilde{x}_i) = \sum_{i=1}^{N} q_i \tilde{x}_i^T P_i^{-1} \tilde{x}_i$, where the positive vector $q = [q_1, q_2, \cdots, q_N]$ such that $q^T \mathcal{L} \geq 0$ [89]. This work gives the observability condition, i.e., the pair (A, H) is observable when $t \to \infty$, such that $P_i(t)$ in (3.16) converges to the positive definite constant matrix, $i \in \mathcal{V}$. Then, $V(\tilde{x}_i) \geq 0$ holds, since $P_i^{-1} > 0$. Taking the derivative of $V(\tilde{x}_i^k)$

yields

$$\begin{aligned}\dot{V}(\tilde{x}_i) &= \mathrm{E}\left\{\sum_{i=1}^{N} q_i \dot{\tilde{x}}_i^T P_i^{-1}\tilde{x}_i + \sum_{i=1}^{N} q_i \tilde{x}_i^T \dot{P}_i^{-1}\tilde{x}_i + \sum_{i=1}^{N} q_i \tilde{x}_i^T P_i^{-1}\dot{\tilde{x}}_i\right\} \\ &= -\mathrm{E}\left\{\sum_{i=1}^{N} q_i(\tilde{x}_i^T P_i^{-1} F S F^T P_i^{-1}\tilde{x}_i + g_i \tilde{x}_i^T H_i^T R_i^{-1} H_i \tilde{x}_i)\right\} \\ &\quad - \mathrm{E}\left\{2\sum_{i=1}^{N} q_i d_i \tilde{x}_i^T P_i^{-1}\tilde{x}_i\right\} + \mathrm{E}\left\{\sum_{i=1}^{N} q_i w^T F^T \tilde{x}_i\right\} \\ &\quad + \mathrm{E}\left\{\sum_{i=1}^{N} q_i \tilde{x}_i^T \sum_{j=1}^{N} a_{ij} P_j^{-1}(\tilde{x}_j - \tilde{x}_i)\right\} \\ &\quad + \mathrm{E}\left\{\sum_{i=1}^{N} q_i \sum_{j=1}^{N} a_{ij}(\tilde{x}_j^T P_j^{-1} - \tilde{x}_i^T P_i^{-1})\tilde{x}_i\right\}. \end{aligned} \tag{3.19}$$

By using Lemma 4 of [41], one has

$$\begin{aligned}\dot{V}(\tilde{x}_i) \leq & -\mathrm{E}\left\{\sum_{i=1}^{N} q_i(\tilde{x}_i^T P_i^{-1} F S F^T P_i^{-1}\tilde{x}_i + g_i \tilde{x}_i^T H_i^T R_i^{-1} H_i \tilde{x}_i)\right\} \\ & - \mathrm{E}\left\{\sum_{i,j=1}^{N} q_i(\tilde{x}_i - \tilde{x}_j)^T P_j^{-1}(\tilde{x}_i - \tilde{x}_j)\right\} \\ & - \sum_{i=1}^{N} q_i \mathrm{E}\{d_i \|P_i\|^{-1}\|\tilde{x}_i\|^2\} + \sum_{i=1}^{N} q_i \mathrm{E}\{\|Fw\|\|\tilde{x}_i\|\}. \end{aligned} \tag{3.20}$$

It is seen that when $t \to \infty$, for each $i \in \mathcal{V}$, if $\mathrm{E}\{\|\tilde{x}_i\|\} \geq \|Fw\|/(d_i\|P_i\|^{-1})$, one has $\dot{V}(\tilde{x}_i) \leq 0$. Notice that $\dot{V}(\tilde{x}_i) = 0$ only when $\tilde{x}_i = 0$. According to the La Salle extension and Definition 2.1, the dynamics of $\tilde{x}_i$ is UUB in the mean square. Furthermore, $\mathrm{E}\{\|\tilde{x}_i\|^2\}$ will stay inside of the neighborhood of $\|Fw\|^2/(d_i\|P_i\|^{-1})^2$. □

Remark 3.3. *As seen in DKCF* (3.16)-(3.17)*, the pinning gain g_i could be 0 or 1. DCKF updates by different resources with different g_i in measurement structure* (3.15)*. When $g_i = 1$, according to* (3.15)*, the measurement of sensor i includes two parts, i.e., direct information from the target and indirect information from neighboring estimates. When $g_i = 0$, sensor i updates the DKCF by weighted neighbor estimates only. Due to different measurement resources, the covariance updates*

including filtering gains are different. Thus, in the following parts, this work studies the robustness margins of the DKCF separately for $g_i = 1$ *and* $g_i = 0$.

3.3.3 Robustness Margins of DKCF with Direct Target Observation

This work studies the robustness proprieties of Algorithm 3.1 at steady state. That means $\dot{P}_i = 0$ for $\forall i \in \mathcal{V}$. One now provides robustness margins for two cases (e.g. $g_i = 1$ and $g_i = 0$). First, set $g_i = 1$ in (3.15), which means that sensor i has direct measurement of the target. Then, it is shown that in this case, the robustness margins of DKCF (3.16)-(3.17) are better than the robustness margins of the single-agent KF in Section 3.2. In next section, set $g_i = 0$, which means sensor i has information from its neighbors only.

In this section, let $g_i = 1$ in (3.15)-(3.17) and thus direct target measurement is obtained by sensor i. In this case, one can consider the covariance update (3.16) to be the same as the single-agent update (5.13), with an additional last term in (3.17) showing coupling from neighbors through a_{ij}.

The following development parallels that of single-agent KF in Section 3.2. Referring to the loop transfer function of single-agent KF (3.7), when sensor i is able to obtain the direct information from target, its loop transfer function of estimation error dynamics in DKCF in Algorithm 3.1 is given by

$$T_i(s) = H_i(sI - A)^{-1}L_i, \quad i \in \mathcal{V} \tag{3.21}$$

with the state-space realization (A, H_i, L_i) with $L_i = P_i H_i^T R_i^{-1}$.

Based on Theorem 3.3, one has the following results.

Lemma 3.1. *Give the DKCF* (3.16) *and* (3.17)*. Define* Φ_i *as follows. Then one has:*

$$\Phi_i = 2d_i P_i - P_i \sum_{j=1}^{N} a_{ij}(P_j^{-1} - P_i^{-1})P_i. \tag{3.22}$$

1) $\Phi_i \geq 0$ *with* $g_i = 1, i \in \mathcal{V}$*,*

2) $\|P_i\|$ *increases as the overall coupling strength* d_i *increases.*

3) $\bar{\sigma}(\Phi_i)$ *increases with the increase of* d_i *with* $g_i = 1, i \in \mathcal{V}$.

Proof. 1). Note that sensor i can directly observe the target. Then, it has no more expected estimation errors than other sensors [59], i.e., $\mathrm{E}\{\|\tilde{x}_i\|\} \leq \mathrm{E}\{\|\tilde{x}_j\|\}$, which implies that $P_i \leq P_j$. Since $P_i > 0$ and $P_j > 0$, one has that $P_i^{-1} \geq P_j^{-1}$. Then, one concludes that $\Phi_i \geq 0$.

2) According to Theorem 3.3, the estimation error dynamics is UUB with certain bound. That is $\mathrm{E}\{\|\tilde{x}_i\|\} \geq \|Fw\|/(d_i\|P_i\|^{-1})$. Considering the definition that $P_i = \mathrm{E}\{(x - \hat{x}_i)^T(x - \hat{x}_i)\}$, one obtains that $\|P_i\|^{\frac{1}{2}} \geq \|Fw\|/(d_i\|P_i\|^{-1})$. Thus, one has $\|P_i\| \leq (d_i/\|Fw\|)^2$, which gives the bound of $\|P_i\|$. It is seen that $\|P_i\|$ increases as d_i increases when $\|Fw\|$ is fixed.

3) The static covariance ARE of (3.16) for sensor i, $\forall i \in \mathcal{V}$ is given by

$$0 = AP_i + P_iA^T - P_iH_i^TR_i^{-1}H_iP_i + Q + \Phi_i \tag{3.23}$$

which gives the same constant solution P_i as that in (3.16). Thus, it is unbiased to study this static covariance ARE. Since $\Phi_i \geq 0$ and $Q > 0$, it is known that in static ARE (3.16), $\|P_i\|$ is increase with the increase of $\|Q + \Phi_i\|$ [50] when A, H_i and R_i are fixed. According to 2) of Lemma 3.1, as d_i increases, $\|P_i\|$ increases, which implies that $\|Q + \Phi_i\|$ increases in (3.23). Since $\|Q\|$ is fixed, one concludes that $\|\Phi_i\|$ increases. Therefore, with the increase of overall coupling strength d_i, $\bar{\sigma}(\Phi_i) = \|\Phi_i\|$ increases. The proof is completed. □

Remark 3.4. *The robustness of the DKCF* (3.16)-(3.17) *is defined as the extent to which the elements of the loop transfer function matrix* $T_i(s)$ *of sensor* i *can vary from their nominal design values without affecting the stability or the convergence of DKCF* (3.16)-(3.17)*. It is characterized by GM and PM. The robustness is said to be improved when the systems have infinite GM, gain reduction tolerance of more than 50 percents and PM of more than* 60°.

Remark 3.5. *It is seen that when sensor* i *directly measures the target, the steady-state error coveriance becomes larger with more neighboring estimates. The reason is that sensor* i *has no worse estimates than its neighbors. With less accurate information entering into measurement*

structure, the sensor i has worse target estimates. However, the following results show that with more neighboring information, the robustness margins including GM and PM improve compared to the single-agent KF.

The next result extends Theorem 3.1 to the multi-sensor DKCF (3.16)-(3.17) with direct target observation.

Theorem 3.4. ***(Minimum Singular Value of DKCF*** (3.16)**-**(3.17)***with Direct Target Observation, i.e.,*** $g_i = 1$***). Give the DKCF in Algorithm 3.1 with*** $g_i = 1$*. Suppose Lemma 3.1 hold. Consider the static covariance ARE of* (3.16) *for sensor i,* $\forall i \in \mathcal{V}$ *as*

$$0 = AP_i + P_iA^T + Q - P_iH_i^TR_i^{-1}H_iP_i + \Phi_i, \tag{3.24}$$

where $\Phi_i = 2d_iP_i - P_i\sum_{j=1}^{N} a_{ij}(P_j^{-1} - P_i^{-1})P_i$*. Then, one has the multi-agent return difference relation*

$$[I + T_i(s)]R_i[I + T_i(s)]^{-T} = R_i + F_i(s). \tag{3.25}$$

Furthermore, one has

$$\underline{\sigma}[I + \bar{T}_i(s)] \geq 1 + \alpha_i^*, \quad s = \alpha\omega \in \alpha\mathbb{R}, \tag{3.26}$$

where α_i^* *is some constant such that* $0 < \alpha_i^* \leq \underline{\sigma}[\bar{F}_i(s)]$*. In addition,* $T_i(s) = H_i(sI-A)^{-1}P_iH_i^TR_i^{-1}$*,* $\bar{T}_i(s) = R_i^{-1/2}T_i(s)R_i^{1/2}$*,* $F_i(s) = H_i(sI - A)^{-1}(Q + \Phi_i)^{1/2}$ *and* $\bar{F}_i(s) = R_i^{-1/2}F_i(s)$*.*

Proof. Let $g_i = 1$ for sensor i, where $i \in \mathcal{V}$. Write static distributed covariance ARE with $\dot{P}_i = 0$ in (3.16) as

$$0 = AP_i + P_iA^T - P_iH_i^TR_i^{-1}H_iP_i + 2d_iP_i - P_i\sum_{j=1}^{N} a_{ij}(P_j^{-1} - P_i^{-1})P_i + FSF^T. \tag{3.27}$$

With Φ_i defined in (3.22), one rewrites (3.27) as

$$0 = AP_i + P_iA^T - P_iH_i^TR_i^{-1}H_iP_i + \Phi_i + Q. \tag{3.28}$$

By introducing frequency domain variable s, $s = \alpha\omega \in \alpha\mathbb{R}$, one has

$$\begin{aligned} 0 &= AP_i + P_iA^T - P_iH_i^TR_i^{-1}H_iP_i + Q + \Phi_i + sP_i - sP_i \\ &= -(sI - A)P_i - P_i(-sI - A^T) - P_iH_i^TR_i^{-1}H_iP_i + Q + \Phi_i. \end{aligned} \tag{3.29}$$

Multiplying the left side of (3.29) by $H_i(sI - A)^{-1}$ and the right side by $(-sI - A^T)^{-1}H_i^T$ yields

$$\begin{aligned} &H_i(sI - A)^{-1}(Q + \Phi_i)(-sI - A^T)^{-1}H_i^T \\ &= H_iP_i(-sI - A_i^T)^{-1}H_i^T + H_i(sI - A_i)^{-1}P_iH_i^T \\ &\quad + H_i(sI - A_i)^{-1}P_iH_i^TR_i^{-1}H_iP_i(-sI - A_i^T)^{-1}H_i^T. \end{aligned} \tag{3.30}$$

Adding R_i to each side of (3.30) yields

$$\begin{aligned} &R_i + H_i(sI - A)^{-1}(Q + \Phi_i)(-sI - A^T)^{-1}H_i^T \\ &= H_iP_i(-sI - A^T)^{-1}H_i^T + H_i(sI - A)^{-1}P_iH_i^T \\ &\quad + R_i + H_i(sI - A)^{-1}P_iH_i^TR_i^{-1}H_iP_i(-sI - A^T)^{-1}H_i^T \\ &= [I + H_i(sI - A)^{-1}P_iH_i^TR_i^{-1}]R_i[I + R_i^{-1}H_iP_i(-sI - A^T)^{-1}H_i^T]. \end{aligned} \tag{3.31}$$

From Lemma 3.1, one has $\Phi_i > 0$. Then, one denotes $F_i(s) = H_i(sI - A)^{-1}(Q + \Phi_i)^{1/2}$ and rewrites (3.31) as

$$[I + T_i(s)]R_i[I + T_i(s)]^{-T} = R_i + F_i(s)F_i(s)^{-T} \tag{3.32}$$

with T_i given in (3.21).

Let $\bar{T}_i(s) = R_i^{-1/2} T_i(s) R_i^{1/2}$ and $\bar{F}_i(s) = R_i^{-1/2} H_i (sI - A)^{-1} (Q + \Phi_i)^{1/2}$, so that

$$[I + \bar{T}_i(s)][I + \bar{T}_i(s)]^{-T} = I + \bar{F}_i(s)\bar{F}_i(s)^{-T}. \tag{3.33}$$

which is expressed in the form of the singular value as

$$\sigma[I + \bar{T}_i(s)] = 1 + \sigma[\bar{F}_i(s)]. \tag{3.34}$$

In terms of the minimum singular value, one has

$$\underline{\sigma}[I + \bar{T}_i(s)] = 1 + \underline{\sigma}[\bar{F}_i(s)]. \tag{3.35}$$

It is seen that $\sigma(\Phi_i) > 0$, and thus there exists some constant α_i^* such that $0 < \alpha_i^* \leq \underline{\sigma}(\bar{F}_i(s))$. Then, one concludes (3.26). The proof is completed. □

Remark 3.6. *The robustness margin problem is to investigate the extent to which the elements of the loop transfer function matrix $T_i(s)$ can vary from their nominal design values without compromising the convergence of DKCF. In order to study the robustness margins, the return difference matrices $I + T_i$ is investigated in* (3.24). *Note that P_i in $T_i(s)$ is the constant solution of the static ARE* (3.24). *This steady solution must be solved by using propagation equation* (3.16). *Furthermore, it is seen that if there is no coupling a_{ij} for sensor i, static distributed covariance ARE* (3.24) *reduces to the standard single-agent KF covariance ARE* (5.13).

Remark 3.7. *With the overall coupling strength $d_i = \sum_{j=1}^{N} a_{ij}$ increasing, $\bar{\sigma}(\Phi_i) > 0$ increases, and $\underline{\sigma}[\bar{F}_i(s)] > 0$ increases so that $\underline{\sigma}[I + \bar{T}_i(s)]$ goes up.*

When a perturbation $B_i(s)$ is added to $T_i(s)$ of sensor i, the loop transfer function becomes

$$\begin{aligned} \tilde{T}_i(s) = T_i(s) B_i(s) &= H_i (sI - A)^{-1} L_i B_i(s) \\ &= \tilde{H}_i (sI - \tilde{A})^{-1} \tilde{L}_i \end{aligned} \tag{3.36}$$

with the state-space realization (A, H_i, L_i) becoming the realization $(\tilde{A}, \tilde{H}_i, \tilde{L}_i)$.

The next main result provides robustness margins for DKCF (3.16)-(3.17) with direct target observation. It extends to the distributed filter case from the single-agent KF in Theorem 3.2. It is seen that both the GM and PM are better for the multi-sensor DKCF than that for the single target Kalman Filter in Theorem 3.2 (see Remark 3.8).

Theorem 3.5. ***(Robustness Margins of DKCF*** (3.16)**-**(3.17) ***with Direct Target Observation,*** $g_i = 1$***).*** *Assume the hypothesis for DKCF in Theorem 3.3. Suppose Theorem 3.4 and the following conditions hold for* $s = \alpha\omega \in \alpha\mathbb{R}$*:*

a) det($sI - A$*) and det(*$sI - \tilde{A}$*) have the same number of CRHP zeros, and if det(*$j\omega_0 I - \tilde{A}$*)=0, det(*$j\omega_0 I - A$*)=0,*

b) $B_i(s)R_i + R_i B_i^{-T}(s) \geq R_i + F_i(s)F_i^{-T}(s)$*,*

c) $B_i(s) + B_i^{-T}(s) \geq 0$*,*

where $F_i(s) = H_i(sI - A)^{-1}(Q + \Phi_i)^{1/2}, i \in \mathcal{V}$*.*

Then one has:

1) The distributed perturbed system $\tilde{T}_i(s)$ *is closed-loop asymptotically stable,*

2) The DKCF (3.16)-(3.17) *has a guaranteed GM given by* $(\frac{1}{2+\alpha_i^*}, \infty)$ *and the GM improves as the overall coupling strength* d_i *increases,*

3) The DKCF (3.16)-(3.17) *has a guaranteed PM given by* $\pm\arccos(\frac{1}{2} - \frac{(\alpha_i^*+2)\alpha_i^*}{2})$ *and the PM improves as the overall coupling strength* d_i *increases, where* α_i^* *is some constant such that* $0 < \alpha_i^* \leq \underline{\sigma}[\bar{F}_i(s)] \leq \sqrt{2} - 1$ *with* $\bar{F}_i(s)$ *defined in Theorem 3.4.*

Before proving Theorem 3.5, two lemmas are introduced.

Lemma 3.2. *The perturbed system* $\tilde{G}(s)$ *is closed-loop asymptotically stable if the following conditions hold [51]:*

1) det($sI - A$*) and det(*$sI - \tilde{A}$*) have the same number of CRHP zeros and if det(*$j\omega_0 I - \tilde{A}$*)=0, det(*$j\omega_0 I - A$*)=0,*

2) The original system $G(s)$ *is closed-loop asymptotically stable,*

3) $\overline{\sigma}[B^{-1}(s) - I] < \alpha = \underline{\sigma}[I + G(s)]$*, where* $\alpha \leq 1$*.*

Lemma 3.3. *The perturbed system $\tilde{G}(s)$ is closed-loop asymptotically stable if the following conditions hold [51]:*

1) conditions 1) and 2) of Lemma 3.2 hold,

2) $\overline{\sigma}[B^{-1}(s) - I] < \alpha = \underline{\sigma}[I + G(s)]$,

3) $B(s) + B^{-T}(s) \geq 0$.

A proof of Theorem 3.5 is given as follows.

Proof of Theorem 3.5. 1). The determinant of the loop transfer function (3.21) can be rewritten as

$$\det[I + T_i(s)] = \frac{\det(sI - A + L_i H_i)}{\det(sI - A)}. \tag{3.37}$$

It is proven in Lian [59] that distributed estimation error dynamic using DKCF (3.16)-(3.17) is closed-loop asymptotically stable by Lyapunov function method. This stability ensures the condition 1) of Lemma 3.3 together with the hypothesis a) in Theorem 3.5.

When a perturbation $B_i(s)$ is added to the system (3.21), (5.26) becomes

$$\det[I + \tilde{T}_i(s)] = \frac{\det(sI - \tilde{A} + \tilde{L}_i \tilde{H}_i)}{\det(sI - \tilde{A})}. \tag{3.38}$$

Moreover, the hypothesis b) in Theorem 3.5 implies that

$$\bar{\sigma}[R_i^{-1/2} B_i^{-1}(s) R_i^{1/2} - I] \leq 1 + \underline{\sigma}[R_i^{-1/2} F_i(s)]. \tag{3.39}$$

which is rewritten as

$$\bar{\sigma}[\bar{B}_i^{-1}(s) - I] \leq 1 + \underline{\sigma}[\bar{F}_i(s)]. \tag{3.40}$$

Besides, Theorem 3.4 gives us

$$\underline{\sigma}[I + \bar{T}_i(s)] \geq 1 + \underline{\sigma}[\bar{F}_i(s)]. \tag{3.41}$$

Based on (3.40) and (3.41), one concludes that the condition 2) of Lemma 3.3 is satisfied.

In addition, with the hypothesis b) in Theorem 3.5, $B_i(s) + B_i^{-T}(s) \geq 0$, and according to Lemma 3.3, the distributed perturbed system $\tilde{T}_i(s)$ in DKCF (3.16)-(3.17) is closed-loop asymptotically stable.

2). Consider closed-loop asymptotic stability of $\tilde{T}_i(s)$. It is similar to the analysis of GM in 2) of Theorem 3.2. Give the minimum singular value bound in (3.26). One obtains a following guaranteed GM for DKCF (3.16)-(3.17) with direct target observation

$$\frac{1}{2+\alpha_i^*} < GM < \infty. \tag{3.42}$$

As topology changes, the robustness margins may change. Define $\underline{\sigma}(Q+\Phi_i)_{(d_i)}$ as the minimum singular value of $Q+\Phi_i$ with d_i, where $d_i \in \{1, 2, \cdots, N-1\}$. Based on 2) of Lemma 3.1, with d_i increasing, $\underline{\sigma}(Q+\Phi_i)_{(d_i)}$ increases while matrices A, H_i and R_i keep unchanged. Furthermore, for sensor i, when d_i reaches the maximum, namely $d_i = N-1$, $\underline{\sigma}(Q+\Phi_i)_{(d_i=N-1)}$ reaches the maximum.

Due to $\bar{F}_i(s) = R_i^{-1/2} H_i (sI - A)^{-1} (Q+\Phi_i)^{1/2}$, one finds

$$\underline{\sigma}[\bar{F}_i(s)]_{(d_i=1)} < \underline{\sigma}[\bar{F}_i(s)]_{(d_i=2)} < \cdots < \underline{\sigma}[\bar{F}_i(s)]_{(d_i=N-1)}. \tag{3.43}$$

It is seen that as d_i increases, $\underline{\sigma}[\bar{F}_i(s)]$ increases, so that α_i^* that satisfies $\alpha_i^* \leq \underline{\sigma}[\bar{F}_i(s)]_{(d_i)}$ becomes larger. From (3.42), note that the lower bound of the GM will become smaller. However, the interval of GM becomes larger and GM improves.

3). The hypothesis b) in Theorem 3.5 can be rewritten as

$$\bar{\sigma}[\bar{B}_i^{-1}(s) - I] \leq 1 + \alpha_i^*. \tag{3.44}$$

Consider $B_i(s)$ as a diagonal matrix $B_i(s) = \mathrm{diag}\{b_{i1}(s), b_{i2}(s), \cdots, b_{ik}(s), \cdots, b_{in}(s)\}$. To

obtain PM, let $b_{ik}(s) = b_{ik}$ and $\mathrm{Re}[b_{ik}(s)] = b_{ik}$. Then, the hypothesis c) in Theorem 3.5 gives

$$b_{ik} > 0 \tag{3.45}$$

which implies

$$|PM| < 90^\circ. \tag{3.46}$$

Let $b_{ik}(s) = e^{j\beta_{ik}(s)}$, where $\beta_{ik}(s)$ is real. It is similar to the analysis of the PM of KF in 3) of Theorem 3.2. One has a guaranteed PM

$$\cos\beta_{ik}(s) > 1 - \frac{(1+\alpha_i^*)^2}{2} \tag{3.47}$$

or

$$\begin{aligned} |PM| &< \arccos(1 - \frac{(1+\alpha_i^*)^2}{2}) \\ &= \arccos(\frac{1}{2} - \frac{(\alpha_i^*+2)\alpha_i^*}{2}). \end{aligned} \tag{3.48}$$

It is seen that as the overall coupling strengths d_i increases, the α_i^* increases and the $|\mathrm{PM}|$ increases. According to (3.48), the $|\mathrm{PM}|$ is no larger than 90°, which means $0 < \alpha_i^* \leq \sqrt{2} - 1$, and $60^\circ < |\mathrm{PM}| < 90^\circ$. The proof is completed. □

Remark 3.8. *Compare Theorem 3.5 with Theorem 3.2 for the single-agent KF. It is seen that the robustness of DKCF* (3.16)-(3.17) *with direct target measurement is better than that of the single-agent KF. Specifically, as the overall coupling strength* $d_i = \sum_{j=1}^{N} a_{ij}$ *increases,* α_i^* *in Theorem 3.4 increases, and the lower bound* $\frac{1}{2+\alpha_i^*}$ *in Theorem 3.5 item 2) decreases. This means the lower bound of the GM becomes smaller than* $\frac{1}{2}$*. Therefore, the guaranteed GM improves. Moreover, the guaranteed* $|\mathrm{PM}|$ *in Theorem 3.5 item 3) increases and is larger than* 60°*. Therefore, the PM for multi-sensor DKCF* (3.16)-(3.17) *is better than the single-agent PM in Theorem 3.2. In contrast to*

robustness performance analysis in [101, 112], this chapter rigorously studies the stability margins and shows the robustness of DKCF (3.16)-(3.17).

Remark 3.9. *Note that α in Lemma 3.2 is less than 1 and α in Lemma 3.3 is larger than 1. The reason is that, in single-agent KF case, each sensor has direct observation of the target states without an additional neighbor estimation enlarging its estimation error covariance. This leads to that the minimum singular value of the return difference matrix $I+\bar{G}(s)$ in* (3.10) *could be less than 1. On the other hand, in distributed estimation filter, namely DKCF* (3.16)-(3.17)*, when $g_i = 1$, sensor i obtains measurements from both the target and neighbor estimates. Neighbor estimates brings errors to the estimation errors of sensor i. This could enlarge the minimum singular value of the return difference matrix $I + \bar{T}_i(s)$ in* (3.26)*. Detailed derivation of DKCF is seen in the previous work [59].*

3.3.4 Robustness Margins of DKCF with Neighbor Estimates Only

This section considers $g_i = 0$ in (3.15)-(3.17), so that sensor i cannot obtain direct target measurements, but only estimates from neighboring sensors. In this case, one can consider the covariance update (3.16) with the last term in (3.17) showing coupling from neighbors through a_{ij}. It is shown again that in this case, the robustness margins of DKCF (3.16)-(3.17) are better than the robustness margins of the single-agent KF.

When $g_i = 0$, the nominal loop transfer function of DKCF (3.16)-(3.17) of sensor i is given as

$$T_i^* = H_i^*(sI - A)^{-1}L_i^*, \tag{3.49}$$

where $H_i^* = (\sum_{j=1}^{N} a_{ij}P_j^{-1})^{\frac{1}{2}}$, $R_I^* = I_2$ and $L_i^* = P_i(H_i^*)^T(R_i^*)^{-1}$. The state-space realization is (A, H_i^*, L_i^*).

The next result extends Theorems 3.1 and 3.4 to the multi-sensor DKCF (3.16) and (3.17) without direct target observation.

Theorem 3.6. ***(Minimum Singular Value of DKCF*** (3.16)**-**(3.17) ***without Target Observation,***

i.e., $g_i = 0$***). Give the DKCF in Algorithm 3.1 with*** $g_i = 0$*. Consider the static covariance ARE of* (3.16) *for sensor* i*,* $\forall i \in \mathcal{V}$ *as*

$$0 = A_i P_i + P_i A_i^T + Q - P_i \sum_{j=1}^{N} a_{ij}(P_j^{-1} - P_i^{-1}) P_i. \tag{3.50}$$

Then, one has the multi-agent return difference relation

$$[I + T_i^*(s)] R_i^* [I + T_i^*(s)]^{-T} = R_i^* + F_i^*(s). \tag{3.51}$$

Furthermore, one has

$$\underline{\sigma}[I + \bar{T}_i^*(s)] \geq 1 + \gamma_i^*, \quad s = \alpha\omega \in \alpha\mathbb{R}, \tag{3.52}$$

where γ_i^* *is some constant such that* $0 < \gamma_i^* \leq \underline{\sigma}[\bar{F}_i^*(s)]$*.* $R_i^* = I_2$*,* $T_i^*(s) = (\sum_{j=1}^{N} a_{ij} P_j^{-1})^{\frac{1}{2}}(sI - A)^{-1} P_i$*,* $\bar{T_i}^*(s) = (R_i^*)^{-\frac{1}{2}} T_i^*(s) (R_i^*)^{\frac{1}{2}}$*,* $F_i^*(s) = (\sum_{j=1}^{N} a_{ij} P_j^{-1})^{\frac{1}{2}}(sI - A)^{-1}(3d_i P_i + Q)^{\frac{1}{2}}$ *and* $\bar{F}_i^*(s) = (R_i^*)^{-\frac{1}{2}} F_i^*(s)$.

Proof. If sensor i does not get information from target, $g_i = 0$. The DKCF (3.16) becomes (3.50). (3.50) can be rewritten as

$$0 = A_i P_i + P_i A_i^T + Q + d_i P_i - P_i (\sum_{j=1}^{N} a_{ij} P_j^{-1}) P_i. \tag{3.53}$$

Let $H_i^* = (\sum_{j=1}^{N} a_{ij} P_j^{-1})^{\frac{1}{2}}$ and $R_I^* = I_2$, so that

$$0 = AP_i + P_i A^T + Q + 3d_i P_i - P_i (H_i^*)^T (R_i^*)^{-1} H_i^* P_i. \tag{3.54}$$

Then, it is similar to the analysis in Theorem 3.1, and one obtains the results (3.50)-(3.52). The proof is completed. □

When a perturbation $B_i^*(s)$ is added to sensor i, the loop transfer function becomes

$$\begin{aligned}\tilde{T}_i^*(s) = T_i^*(s)B_i^*(s) &= H_i^*(sI-A)^{-1}L_i^*B_i^*(s) \\ &= \tilde{H}_i^*(sI-\tilde{A})^{-1}\tilde{L}_i^* \end{aligned} \tag{3.55}$$

with the state-space realization (A, H_i^*, L_i^*) becoming the realization $(\tilde{A}, \tilde{H}_i^*, \tilde{L}_i^*)$.

The next main result provides robustness margins for DKCF (3.16)-(3.17) without direct target observation. It should be compared to Theorems 3.2 and 3.5. It is seen that in this case, both the GM and PM are better for the multi-sensor DKCF than for the single target Kalman Filter in Theorem 3.2 (see Remark 3.6).

Theorem 3.7. ***(Robustness Margins of DKCF*** (3.16)**-**(3.17) ***without Target Observation, i.e.,*** $g_i = 0$***).*** *Assume the hypothesis for DKCF in Theorem 3.3. Suppose Theorem 3.6 and the following conditions hold for* $s = \alpha\omega \in \alpha\mathbb{R}$*:*

a) det($sI - A$*) and det(*$sI - \tilde{A}$*) have the same number of CRHP zeros, and if det(*$j\omega_0 I - \tilde{A}$*)=0, det(*$j\omega_0 I - A$*)=0,*

b) $B_i^*(s)R_i^* + R_i^*(B_i^*(s))^{-T} \geq R_i^* + F_i^*(s)(F_i^*(s))^{-T}$*,*

c) $B_i^*(s) + (B_i^*(s))^{-T} \geq 0$*,*

where $R_i^* = I_2$ *and* $F_i^*(s) = (\sum_{j=1}^{N} a_{ij}P_j^{-1})^{\frac{1}{2}}(sI-A)^{-1}(3d_iP_i + Q)^{1/2}$*.* $B_i^*(s)$ *is defined in* (3.55)*. Define* γ_i^* *as the constant real number such that* $0 < \gamma_i^* \leq \underline{\sigma}[\bar{F}_i^*(s)]$ *with* $\bar{F}_i^*(s) = (R_i^*)^{-1/2}F_i^*(s)$*. One then has:*

1) The distributed perturbed system $\tilde{T}_i^*(s)$ *is closed-loop asymptotically stable,*

2) The DKCF (3.16)-(3.17) *has a guaranteed GM given by* $(\frac{1}{2+\gamma_i^*}, \infty)$ *and the GM improves as the overall coupling strength* d_i *increase,*

3) The DKCF (3.16)-(3.17) *has a guaranteed PM given by* $\pm\arccos(\frac{1}{2} - \frac{(\gamma_i^*+2)\gamma_i^*}{2}), 0 \leq \gamma_i^* \leq \sqrt{2}-1$ *and the PM improves as the overall coupling strength* d_i *increases.*

Proof. 1). It is similar to the proof of 1) in Theorem 3.5. First, as seen in 1) of Theorem 3.6, the minimum singular value of $[I + \bar{T}_i^*(s)]$ is larger than 1. Second, given the hypothesis, $B_i^* +$

$(B_i^*)^{-T} \geq 0$ in Theorem 3.7. These two conditions ensure conditions 2) and 3) of Lemma 3.3 hold. Third, this work proves the asymptotic stability of DKCF when $g_i = 0$ based on Lyapunov method in Lian [59]. In addition, combining these with the hypothesis a) in Theorem 3.7, one guarantees that Lemma 3.3 holds so that the perturbed system $\tilde{T}_i^*(s)$ is closed-loop asymptotically stable for DCKF when $g_i = 0$.

2). Consider closed-loop asymptotic stability of $\tilde{T}_i^*(s)$. According to (3.51), one has

$$\sigma[I + \bar{T}_i^*(s)] = 1 + \sigma[\bar{F}_i^*(s)], \tag{3.56}$$

where $\bar{T_i}^*(s) = (R_i^*)^{-1/2} T_i^*(s) (R_i^*)^{1/2}$, $R_i^* = I_2$, $T_i^*(s) = (\sum_{j=1}^{N} a_{ij} P_j^{-1})^{\frac{1}{2}} (sI - A)^{-1} P_i$, $\bar{F}_i^*(s) = (R_i^*)^{-1/2} F_i^*(s)$ and $F_i^*(s) = (\sum_{j=1}^{N} a_{ij} P_j^{-1})^{\frac{1}{2}} (sI - A)^{-1} (3 d_i P_i + Q)^{1/2}$.

With a fixed d_i, there exist some positive constant γ_i^* such that $0 < \gamma_i^* \leq \underline{\sigma}[\bar{F}_i^*(s)]$, so that

$$\underline{\sigma}[I + \bar{T}_i^*(s)] \geq 1 + \gamma_i^*. \tag{3.57}$$

Select $B_i^*(s)$ as a diagonal matrix $B_i^* = \mathrm{diag}\{b_{i1}^*, b_{i2}^*, \cdots, b_{ik}^*, \cdots, b_{in}^*\}$ which simplifies the hypothesis b) in Theorem 3.7 to

$$|(b_{ik}^*)^{-1}(s) - 1| \leq 1 + \gamma_i^*, \quad \forall k \in \{1, 2, \cdots n\}. \tag{3.58}$$

To obtain GM, let $b_{ik}^*(s) = b_{ik}^*$ and $\mathrm{Re}[b_{ik}^*(s)] = b_{ik}^*$.

In addition, the hypothesis c) in Theorem 3.7 gives

$$b_{ik}^* > 0. \tag{3.59}$$

Based on (3.58) and (3.59), one has a guaranteed GM

$$0 < \frac{1}{2 + \gamma_i^*} < GM < \infty. \tag{3.60}$$

Note that as d_i increases, $\|P_i\|$ increases and $\underline{\sigma}(3d_iP_i) > 0$ increases so that γ_i^* increases. The lower bound of the GM becomes smaller. Thus, the GM improves.

3). It is similar to the analysis of 3) in Theorem 3.3.

In particular, (3.58) gives

$$|PM| < \arccos(1 - \frac{(1+\gamma_i^*)^2}{2}) \tag{3.61}$$

while (3.59) gives

$$|PM| < 90^\circ. \tag{3.62}$$

Based on (3.61) and (3.62), the PM of DKCF without direct target estimation is given by

$$|PM| < \arccos(\frac{1}{2} - \frac{(\gamma_i^*+2)\gamma_i^*}{2}), \quad 0 \le \gamma_i^* \le \sqrt{2} - 1. \tag{3.63}$$

Furthermore, as it is analyzed in 2) of Theorem 3.7, when d_i increases, γ_i^* and $\frac{(\gamma_i^*+2)\gamma_i^*}{2}$ increase so that PM improves. However, it should be noted that $60^\circ < |\mathrm{PM}| < 90^\circ$. The proof is completed.

□

Remark 3.10. *Compare Theorem 3.7 and Theorem 3.2. It is seen that the robustness of DKCF* (3.16)-(3.17) *with indirect neighbor estimates improves compared to that of the single-agent KF. Specifically, the lower bound of GM* $\frac{1}{2+\gamma_i^*}$ *is smaller than 0.5 and the PM is larger than* 60°*, which means that gain and phase margins are better for DKCF* (3.16)-(3.17) *than the standard KF. Furthermore, as overall coupling strengths increase, PM and GM improves.*

Remark 3.11. *It should be noted that this work analyzes the robustness margins of DKCF* (3.16)-(3.17) *in two cases, where* $g_i = 1$ *and* $g_i = 0$*. The reason is that in these two cases, DKCF updates by different filtering gains. Particularly, when* $g_i = 1$*, the filtering gain is given as* L_i *(see* (3.21)*). When* $g_i = 0$*, the filtering gain is* L_i^* *(see* (3.49)*).*

3.4 Simulation Studies

This section first verifies the effectiveness of the estimation of DKCF (3.16)-(3.17) in sensor networks for an unstable target dynamics and a stable target dynamics, respectively. Then, the robustness margins of DKCF in two cases (i.e., $g_i = 1$ and $g_i = 0$) are analyzed for the stable target dynamics.

3.4.1 DKCF with Unstable Target Dynamics

Consider an unstable target dynamic system as

$$\dot{x} = \begin{bmatrix} 0.5 & 1 & 0 & 0 \\ -0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & -0.5 \\ 0 & 0 & 0.5 & 0 \end{bmatrix} x + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \omega, \tag{3.64}$$

where ω is the white noise with the power spectral density $S = \mathrm{diag}\{0.5, 0.5, 0.5, 0.5\}$ and $\omega \sim (0, 0.2\mathbf{1}_4)$, and $\mathbf{1}_4$ is four dimensional vector with all elements equaling to 1.

To verify the effectiveness of DKCF (3.16)-(3.17) given the unstable target dynamics in (3.64), this work sets a multi-sensor network, where there are six sensors and the communication graph is shown in Figure 3.1. It is seen that the graph is strongly connected. The initials, T , S1 to S6 represent target, sensor 1 to sensor 6, respectively. Figure 3.1 shows that $g_3 = g_4 = 1$ and $g_1 = g_2 = g_5 = g_6 = 0$. The other parameters of DKCF covariance update (3.16) in Algorithm 3.1 are selected as: $H_i = I_4$, $R_i = I_4$, and the initial conditions $P_i(0) = I_4$, $i \in \{1, 2, 3, 4, 5, 6\}$.

Figure 3.2 shows the convergence of covariance value P_{22} of six sensors using DKCF (3.16)-(3.17) for the unstable target (3.64). It is seen that DKCF converges within a short period. Note that sensors 3 and 4 has lower estimation corvariance than the other sensors since they directly measure the target.

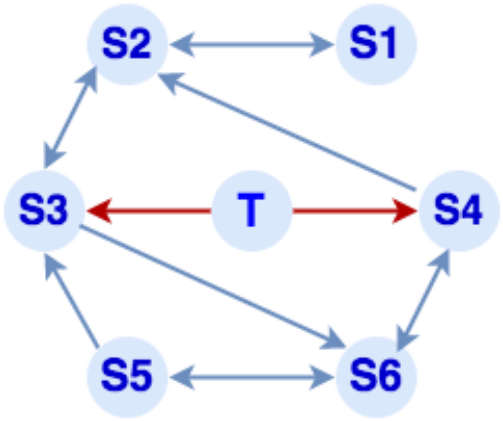


Figure 3.1: Communication graph of six sensors with unstable target

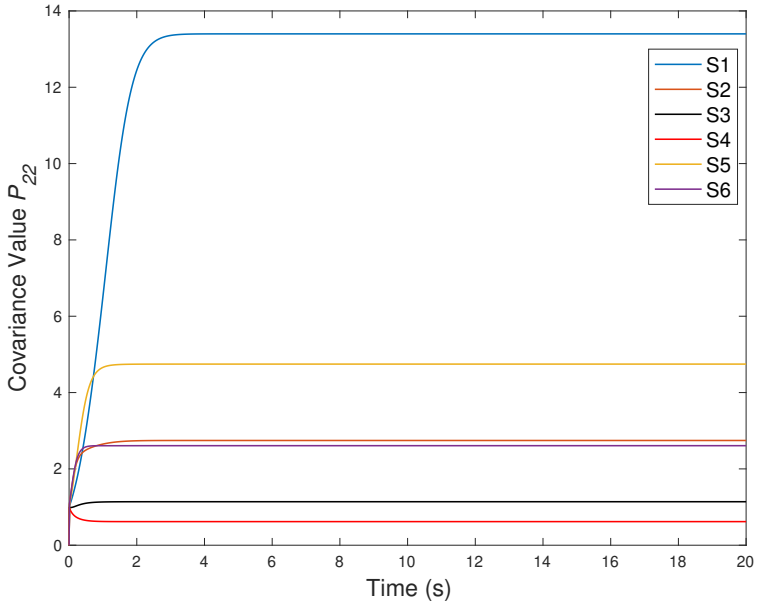


Figure 3.2: Evaluation of covariance value P_{22} in DKCF of 6 sensors for the unstable target (3.64)

3.4.2 DKCF with Stable Target Dynamics

Consider a stable target dynamics system as

$$\dot{x} = \begin{bmatrix} -2 & 1 \\ -4 & -2 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \omega, \tag{3.65}$$

where ω is the white noise with the power spectral density $S = \text{diag}\{2, 1\}$ and $\omega \sim (0, 0.5\mathbf{1}_2)$, and $\mathbf{1}_2$ is two dimensional vector with all elements equaling to 1. The parameters are selected as: $H_i = I_2$, $R_i = I_2$, and the initial conditions $P_i(0) = I_2$, where $i \in \{1, 2, 3, 4\}$.

In KF, the parameters for covariance update (3.5) are $H = I_2$, $R = I_2$ and $P(0) = I_2$. In multi-sensor network, there are four, $N = 4$, sensors distributed in the network to estimate the target state. The initial communication graph is shown as the case 2 in Figure 3.4. It is seen that the graph is strongly connected. In this case, $g_1 = g_2 = 1$ and $g_3 = g_4 = 0$. Other parameters of DKCF covariance update (3.16) in Algorithm 3.1 are selected as: $H_i = I_2$, $R_i = I_2$, and the initial conditions $P_i(0) = I_2$, where $i \in \{1, 2, 3, 4\}$.

Figure 3.3 shows the convergence of covariance value P_{22} of four sensors using DKCF (3.16)-(3.17). It is seen that DKCF converges within a short period. Note that sensors 1 and 2 has lower estimation corvariance than the other sensors since they directly measure the target.

3.4.3 Robustness Margin Analysis of DKCF

Section 3.3.2 and 3.3.3 study the robustness margins of DKCF (3.16)-(3.17) for different information resources, $g_i = 1$ and $g_i = 0$, respectively. The effects of the overall coupling strength $d_i = \sum_{j=1}^{4} a_{ij}$ on GM and PM are analyzed.

The following two examples are given to verify the analysis results of Theorems 3.2, 3.5 and 3.7 given the stable target dynamics (3.65). Example 1 studies the robustness margins of sensor 1 that has direct observation from the target, i.e., $g_1 = 1$. In Example 2, sensor 1 has weighted-neighbor estimates only, i.e., $g_1 = 0$.

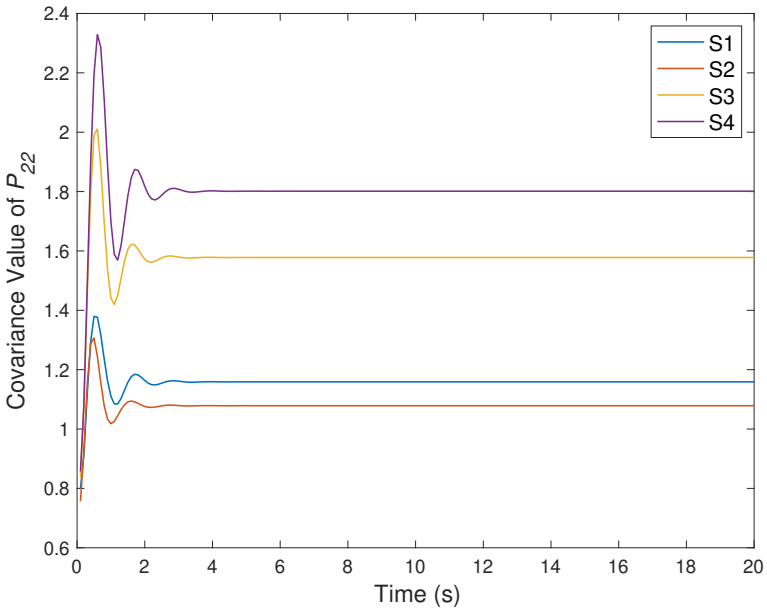


Figure 3.3: Evaluation of covariance value P_{22} in DKCF of 4 sensors for the stable target (3.65)

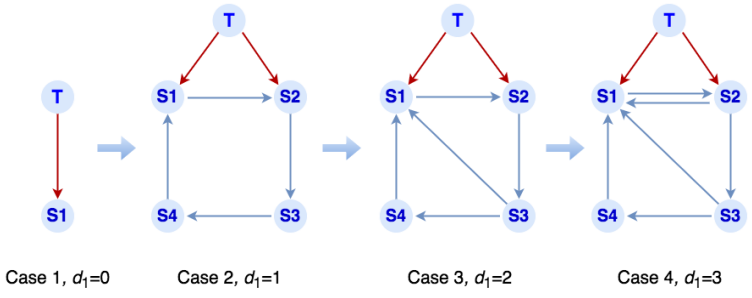


Figure 3.4: Example 1: communication graph changing with the overall coupling strength d_1 increasing for $g_1 = 1$

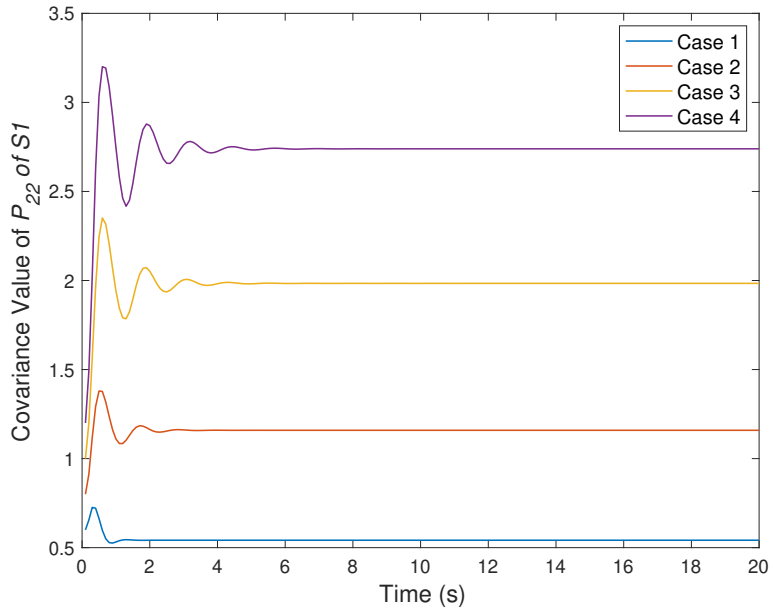


Figure 3.5: Evaluation of covariance value P_{22} of sensor 1 in four cases.

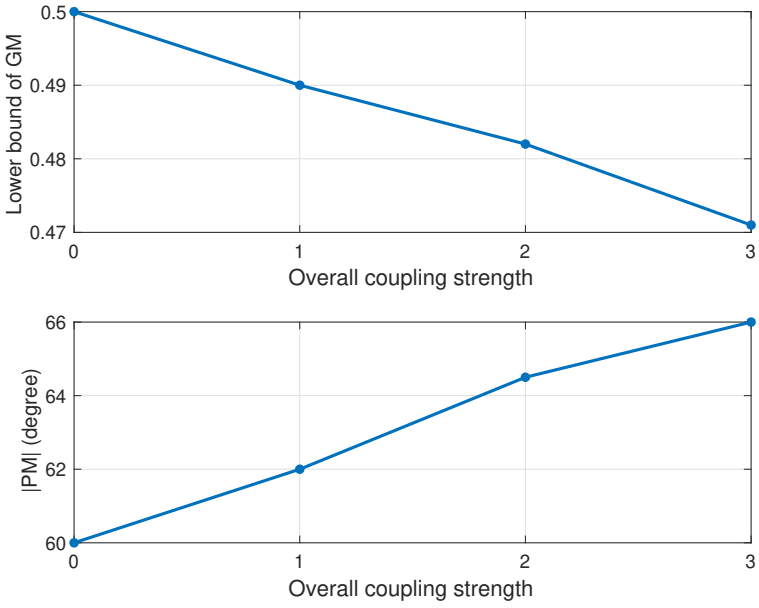


Figure 3.6: Evaluation of $|\mathrm{PM}|$ and the lower bound of the GM for 4 cases in example 1.

Example 1 (Sensor 1 has direct target observation with $g_1 = 1$). Figure 3.4 displays four cases of different overall coupling strengths for sensor 1, $d_1 = 0, 1, 2, 3$, respectively. Particularly, in case 1, $d_1 = 0$. It implies that sensor 1 observes the target directly and has no neighbors so that it estimates the target state by using KF. It has a guaranteed downward GM 0.5 and |PM| 60° according to Theorem 3.2. In the other three cases, sensor 1 uses DKCF to estimate the target state. Figure 3.5 shows the covariance value P_{22} of sensor 1 in four cases. It is seen that P_{22} increases with the coupling increases. Figures 3.6 shows the GM and PM with the overall coupling strength d_1 increasing by 1 from 0 to 3. When $d_1 = 0$, it has a guaranteed GM 0.5 and |PM| 60°. In addition, it is seen that the lower bound of the GM decreases so that the interval of the GM becomes larger. Also, the PM climbs. It means that GM and PM improve as the overall coupling strength increases. These results are consistent with the analysis in Theorems 3.2 and 3.5.

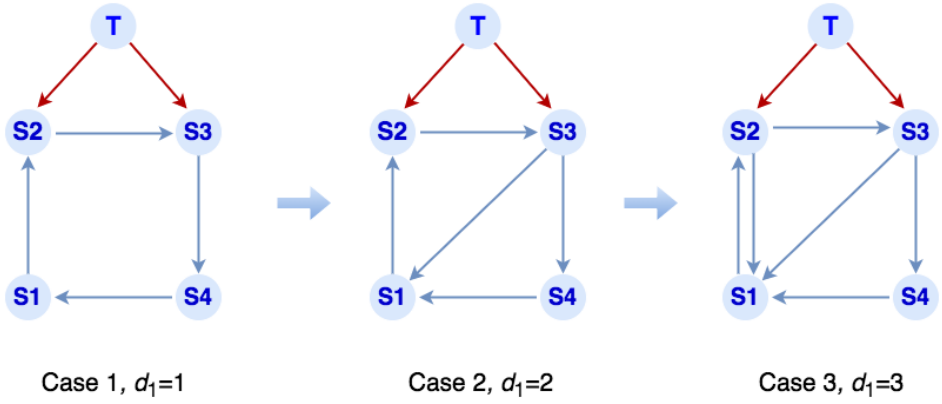


Figure 3.7: Example 2: communication graph changing with the overall coupling strength d_1 increasing for $g_1 = 0$

Example 2 (Sensor 1 has neighbor estimates only with $g_1 = 0$). The overall coupling strength d_1 goes up from 1 to 3 in three cases as shown in Figure 3.7. Figures 3.8 shows the GM and PM of three cases in Example 2. It is seen that when sensor 1 has neighbors' estimates only, with the overall coupling strength increasing, the GM and PM of DKCF improve. The results of this example are consistent with the analysis in Theorem 3.7.

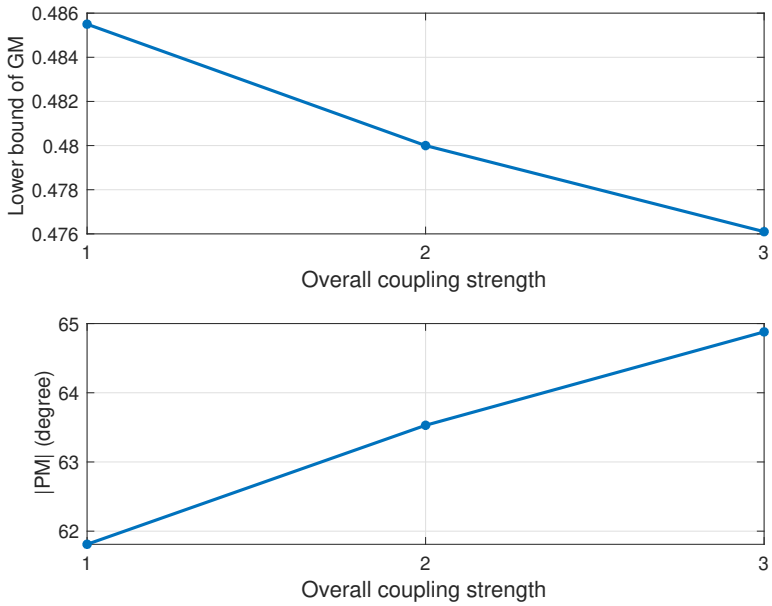


Figure 3.8: Evaluation of $|\mathrm{PM}|$ and the lower bound of the GM for 3 cases in example 2.

3.5 Conclusion

This chapter studies the robustness stability margins including gain margins and phase margins of continuous-time single-agent Kalman filter in frequency domain. The minimum singular value of the return difference is rigorously analyzed for the robustness margins. Margin results are then extended to DKCF in sensor networks. Particularly, two cases are studied for the robustness margins of DKCF. One is based on the direct target observation while the other is not. In addition, the influence of graph overall coupling strengths on the robustness margins of DKCF is rigorously analyzed. In two cases, as the overall coupling strength increases, gain and phase margins improve.

Chapter 4: INVERSE RL FOR ADVERSARIAL APPRENTICE GAMES

4.1 Introduction

This chapter proposes new inverse reinforcement learning (RL) algorithms to solve to be defined Adversarial Apprentice Games for nonlinear learner and expert systems. The games are solved by extracting the unknown cost function of an expert by a learner using demonstrated expert's behaviors. This chapter first develops a model-based inverse RL algorithm that consists of two learning stages; an optimal control learning and a second learning based on inverse optimal control. This algorithm also clarifies the relations between inverse RL and inverse optimal control. Then, a model-free integral inverse RL algorithm is developed to reconstruct the unknown expert cost function. The model-free algorithm only needs online demonstration of the expert and learner's trajectory data without knowing system dynamics of either the learner or the expert. These two algorithms are further implemented using neural networks (NNs). In Adversarial Apprentice Games, the learner and the expert are allowed to suffer from different adversarial attacks. A two-player zero-sum game is formulated for each of these two agents and is solved as a subproblem for the learner in inverse RL. Furthermore, it is shown that the cost functions that the learner learns to mimic the expert's behavior are stabilizing and not unique. Finally, simulations and comparisons show the effectiveness and the superiority of the proposed algorithms.

The chapter is organized as follows. Section 4.2 describes the Adversarial Apprentice Games, the learner and the expert systems. Section 4.3 proposes a model-based inverse RL algorithm and implements this algorithm using NNs. In Section Section 4.4, a model-free inverse RL algorithm is developed and further implemented using NNs. Section 4.5 verifies the proposed algorithms with simulation examples. Section 4.6 concludes the work.

4.2 Problem Formulation

4.2.1 Adversarial Apprentice Games

Various practical applications of apprentice-learning theories [2, 3, 109] require that a learner imitates demonstrated expert behaviors by learning unknown expert cost function. This work further supposes that both the learner and the expert suffer from different adversarial disturbances in their dynamics. For example, in imitation learning of autonomous helicopter [2] from human pilots, the helicopter has disturbances including aerodynamic forces or moments [21], and human pilots are influenced by noises and distractions. Given the cost of states, control and adversarial inputs, each agent computes its Nash equilibrium by solving a two-player zero-sum game. The learner then uses inverse optimal control to infer the unknown expert cost function. The apprentice learning based on two-player zero-sum games and inverse optimal control is called *Adversarial Apprentice Games*.

In the games, the expert has computed desired behaviors found by solving Nash equilibrium of its two-player zero-sum game. Given the expert's behaviors (states and control inputs), the learner expects to reconstruct the unknown expert cost function based on inverse optimal control and solves a formed two-player zero-sum game using RL, thereby imitating the demonstrated behavior. Note that both inverse optimal control and two-player zero-sum games are solved as subproblems in Adversarial Apprentice Games.

Specifically, an Adversarial Apprentice Game is defined as a tuple $(\mathcal{S}, \mathcal{A}, Q)$, where $\mathcal{S}$ and $\mathcal{A}$ are state and action sets, respectively, and $Q : \mathcal{S} \to \mathbb{R}$ denotes a quadratic function of states and is called the *state penalty* of the cost function.

4.2.2 Two-Player Zero-sum Expert System

This subsection formulates an expert as a standard two-player zero-sum game. This provides a basis for Adversarial Apprentice Games. Consider an expert with the nonlinear time-invariant

affine dynamics

$$\dot{x}_e = f(x_e) + g(x_e)u_e + h(x_e)v_e, \tag{4.1}$$

where $x_e \in \mathbb{R}^n$, $u_e \in \mathbb{R}^m$ and $v_e \in \mathbb{R}^k$ denote the expert's state, control input and adversarial input, respectively; $f \in \mathbb{R}^n$, $g \in \mathbb{R}^{n\times m}$ and $h \in \mathbb{R}^{n\times k}$ denote the drift function, the control input function, and the adversarial input function, respectively. Assume that f, g and h are Lipschitz functions with $f(0) = 0$, and that there exists a continuous control input which stabilizes (4.1).

The expert has desired behaviors found by optimizing its performance cost function and simultaneously attenuating the effects of the adversary. Toward this end, its performance cost function is

$$V_e(x_e, u_e, v_e) = \int_{t_0}^{\infty} \left(Q_e(x_e) + u_e^T R_e u_e - v_e^T S_e v_e\right) d\tau, \tag{4.2}$$

where $Q_e(x_e) = q^T(x_e)Q_e q(x_e) \in \mathbb{R}$ is expert's state penalty with the state-penalty weight $Q_e = Q_e^T \in \mathbb{R}^{n\times n} \geq 0$ and a function $q(x_e) = [x_{e_1}^s \; x_{e_2}^s \; \cdots \; x_{e_N}^s] \in \mathbb{R}^n$ with the power s; $R_e = R_e^T \in \mathbb{R}^{m\times m} > 0$ and $S_e = S_e^T \in \mathbb{R}^{k\times k} > 0$ are weights of inputs u_e and v_e, respectively.

Differentiating (4.2) yields the expert's Bellman equation

$$H_e(V_e, u_e, v_e) \triangleq Q_e(x_e) + u_e^T R_e u_e - v_e^T S_e v_e + \nabla V_e^T (f(x_e) + g(x_e)u_e + h(x_e)v_e) = 0. \tag{4.3}$$

According to [54], the saddle point (u_e^*, v_e^*) is found as

$$u_e^* = -\frac{1}{2} R_e^{-1} g^T(x_e) \nabla V_e^*(x_e), \tag{4.4}$$

$$v_e^* = \frac{1}{2} S_e^{-1} h^T(x_e) \nabla V_e^*(x_e), \tag{4.5}$$

which are substituted into (4.3) to yield the expert's Hamilton-Jacobi-Isaacs (HJI) equation

$$0 = Q_e(x_e) - \frac{1}{4}\nabla V_e^{*T}(x_e) g(x_e) R_e^{-1} g^T(x_e) \nabla V_e^*(x_e) + \frac{1}{4}\nabla V_e^{*T}(x_e) h(x_e) S_e^{-1} h^T(x_e) \nabla V_e^*(x_e) + \nabla V_e^{*T}(x_e) f(x_e). \tag{4.6}$$

The expert obtains the optimal input u_e^* and the worst adversary v_e^* by solving the two-player zero-sum game

$$V_e^*(x_e) = V_e^*(u_e^*, v_e^*) = \min_{u_e} \max_{v_e} V_e(u_e, v_e), \tag{4.7}$$

where V_e^* is the optimal cost and satisfies the Nash equilibrium condition [54], i.e.,

$$V_e(x_e(t_0), u_e^*, v_e) \leq V_e(x_e(t_0), u_e^*, v_e^*) \leq V_e(x_e(t_0), u_e, v_e^*). \tag{4.8}$$

4.2.3 Learner System

Consider a learner with nonlinear time-invariant affine dynamics

$$\dot{x}_l = f(x_l) + g(x_l)u_l + h(x_l)v_l, \tag{4.9}$$

where $x_l \in \mathbb{R}^n$ denotes the learner state, $u_l \in \mathbb{R}^m$ denotes its control input and $v_l \in \mathbb{R}^k$ denotes its adversarial input. Assume the expert and learner have identical dynamics, that is f, g and h have the same mapping as those in (4.1).

The learner intends to optimize a performance cost function, while simultaneously attenuating the adversarial effects. Toward this end, its performance cost function is defined as

$$V_l(x_l, u_l, v_l, Q_l) = \int_{t_0}^{\infty} (Q_l(x_l) + u_l^T R_l u_l - v_l^T S_l v_l) d\tau, \tag{4.10}$$

where $Q_l(x_l) = q^T(x_l) Q_l q(x_l) \in \mathbb{R}$ is learner's state penalty with a state-penalty weight $Q_l =$

$Q_l^T \in \mathbb{R}^{n\times n} \geq 0$ and a function $q(x_l) \in \mathbb{R}^n$ of the state x_l; $R_l = R_l^T \in \mathbb{R}^{m\times m} > 0$ and $S_l = S_l^T \in \mathbb{R}^{k\times k} > 0$ are arbitrarily selected weights. The function $q(\cdot)$ in (4.2) and (4.10) have the same mapping.

To solve the Adversarial Apprentice Games, The next standard assumption is provided before proposing the game goals.

Assumption 4.1. *The learner knows its own cost function weights, i.e., Q_l, R_l, S_l in* (4.10) *and the mapping $q(\cdot)$, but does not know the expert's cost function weights, i.e., Q_e, R_e and S_e in* (4.2). *R_l and S_l can be different from R_e and S_e, respectively. The learner does not need to know the expert's R_e and S_e. Additionally, the learner and expert have the same initial states, i.e., $x(t_0) = x_e(t_0)$.*

Remark 4.1. *Assuming that the learner and expert have the same initial state is equivalent to assuming that the learner knows the initial state $x_e(t_0)$ of the expert. This is far milder that other works [3, 14, 18, 37, 38, 53, 63, 64, 79, 98, 104, 109], which assume that the full state $x_e(t)$ of the expert is measured for all time. In fact, this chapter only assumes that the learner measures the expert's input $u_e^*(t)$.*

Adversarial Apprentice Game Goals. Consider the expert (4.1) with an adversary and the resulting desired behavior (x_e, u_e^*) that satisfies the two-player zero-sum game (4.7). Then, given the expert's demonstrated desired control input u_e^*, the learner aims to learn an equivalent weight (to be defined in Definition 4.1 below) to Q_e that satisfies (4.6) such that 1) it performs the expert's behavior, i.e., (x_l, u_l^*) = (x_e, u_e^*) with actual $v_l = v_e$ and 2) its dynamics (4.9) is stabilized.

4.3 Model-based Inverse RL

In order to achieve the above Adversarial Apprentice Game Goals, this section develops a model-based inverse RL algorithm by combining optimal control learning and inverse optimal control learning. First, given $R_l > 0$, $S_l > 0$ and current $Q_l \geq 0$ in (4.10), the learner uses optimal control learning to solve the solutions of optimal control inputs and the worst-case adversarial inputs. Then, with these solutions and expert's demonstration u_e^* in (4.4), the learner revises Q_l

based on inverse optimal control while keeping R_l and S_l unchanged. By iterating these two learning process, the learner learns an equivalent weight to Q_e, such that the optimal control learning obtains (x_l, u_l^*) = (x_e, u_e^*) with actual $v_l = v_e$. The convergence and the stability of the algorithm are further analyzed. The model-based inverse RL algorithm is implemented using NNs.

4.3.1 Optimal Control Learning

Differentiating (4.10) yields the learner's Bellman equation

$$H_l(V_l, u_l, v_l) \triangleq Q_l(x_l) + u_l^T R_l u_l - v_l^T S_l v_l + \nabla V_l^T (f(x_l) + g(x_l)u_l + h(x_l)v_l) = 0. \tag{4.11}$$

The learner's Nash solution (u_l^*, v_l^*) is found as

$$u_l^* = -\frac{1}{2} R_l^{-1} g^T(x_l) \nabla V_l^*(x_l), \tag{4.12}$$

$$v_l^* = \frac{1}{2} S_l^{-1} h^T(x_l) \nabla V_l^*(x_l), \tag{4.13}$$

which yields the learner's HJI equation

$$\begin{aligned} 0 = Q_l(x_l) &- \frac{1}{4} \nabla V_l^{*T}(x_l) g(x_l) R_l^{-1} g^T(x_l) \nabla V_l^*(x_l) \\ &+ \frac{1}{4} \nabla V_l^{*T}(x_l) h(x_l) S_l^{-1} h^T(x_l) \nabla V_l^*(x_l) + \nabla V_l^{*T}(x_l) f(x_l), \end{aligned} \tag{4.14}$$

where V_l^* is the optimal cost and (u_l^*, v_l^*) is Nash equilibrium of the learner. They are the solutions of a two-player zero-sum game, i.e., $V_l^*(x_l) = V_l^*(u_l^*, v_l^*) = \min_{u_l} \max_{v_l} V_l(u_l, v_l)$.

It is known that different cost functions can lead to the same control policy [40]. That is, to obtain $u_l^* = u_e^*$, the learner can find the following equivalent weight to Q_e while R_l, S_l in (4.10) are different from R_e, S_e in (4.2).

Definition 4.1. ***(Equivalent weight to** Q_e**)*** *The expert's Q_e satisfies expert's HJI equation* (4.6)*. Suppose one finds a Q_l^∞ in learner's HJI equation* (4.14) *in which there exists a $V_l^\infty(x_e)$ such that 1) u_l^∞ in the form of* (4.12) *equals to u_e^** (4.4) *and 2) $x_l = x_e$ holds with $v_l = v_e$ in* (4.1) *and* (4.9)*.*

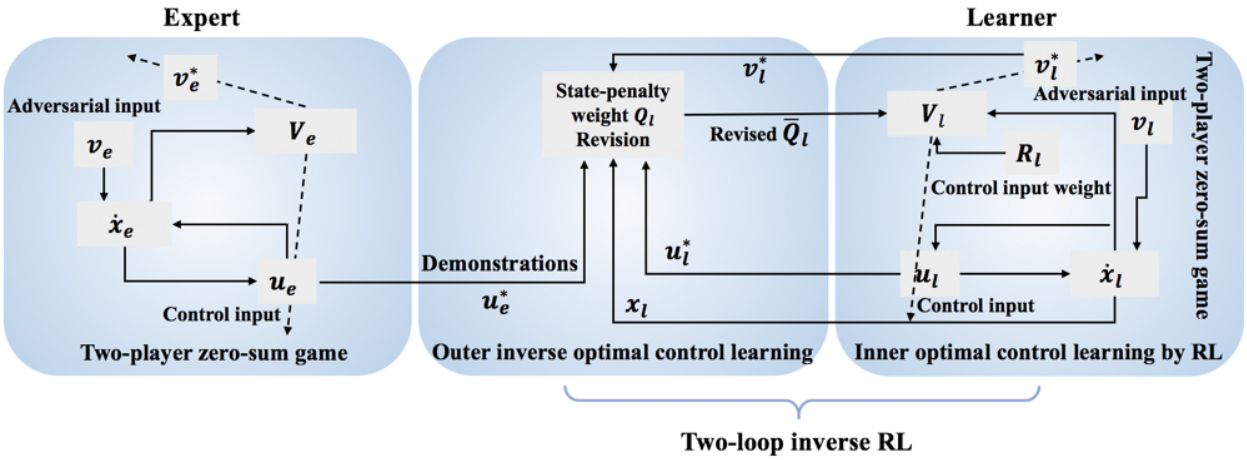


Figure 4.1: Schematic diagram of inverse RL for Adversarial Apprentice Game.

Then, Q_l^∞ is called to be an equivalent weight to Q_e.

Given a current estimate Q_l and fixed R_l and S_l in (4.10), the policy iteration (PI) [43] is used to solve (4.14), (4.12) and (4.13) for optimal control problem to the learner. The PI for optimal control learning is seen as the inner loop of Algorithm 4.1. If the current estimate Q_l is not the equivalent weight to Q_e, then Q_l needs to be revised, as seen in Section 4.3.2.

4.3.2 State-penalty Weight Q_l Revision Based on Inverse Optimal Control

In this subsection, given expert's demonstration u_e^* in (4.4), the current Q_l is revised to be an equivalent weight to Q_e based on inverse optimal control.

Given the demonstrated u_e^* and the solutions of optimal control learning, the learner's state-penalty weight $\bar{Q}_l$ is revised based on inverse optimal control [31] as

$$q^T(x_l)\bar{Q}_l q(x_l) = u_e^{*T} R_l u_e^* - 2u_e^{*T} R_l u_l^* + v_l^{*T} S_l v_l^* - \nabla V_l^{*T}(x_l)(f(x_l) + g(x_l)u_l^* + h(x_l)v_l^*). \tag{4.15}$$

Then, the learner uses this revised $\bar{Q}_l$ in its HJI equation (4.14) to update u_l^*, v_l^* and $V_l^*(x_l)$. Repeat PI and inverse optimal control learning until Q_l converges to an equivalent weight to Q_e. As a result, (4.14) gives the same solution $\nabla V_e(x_e)$ as that by expert's HJI (4.6). The learner thus obtains the behavior (x_e, u_e^*). The iterative form of this inverse optimal control learning is seen as

the outer iteration loop of Algorithm 4.1.

This work now presents iterative Algorithm 4.1 below to solve the Adversarial Apprentice Game. The schematic diagram of inverse RL for the Adversarial Apprentice Game is depicted in Figure 4.1.

Algorithm 4.1 Model-based Inverse RL Algorithm for Adversarial Apprentice Game

1: **Initialization**: select $Q_l^0 \geq 0$, $R_l > 0$, $S_l > 0$, initial stabilizing u_l^{00}, $v_l^{00} = 0$, and small thresholds ε_1, ε_2. Set $j = 0$.

2: **Outer j iteration loop based on inverse optimal control**

3: **Inner i iteration loop using optimal control:** given j, set $i = 0$, use initial stabilizing u_l^{j0}.

4: Policy evaluation for solving V_l^{ji}

$$\begin{aligned} 0 = {} & q^T(x_l)Q_l^j q(x_l) + (u_l^{ji})^T R_l u_l^{ji} - (v_l^{ji})^T S_l v_l^{ji} \\ & + \left(\nabla V_l^{ji}(x_l)\right)^T \left(f(x_l) + g(x_l)u_l^{ji} + h(x_l)v_l^{ji}\right). \end{aligned} \tag{4.16}$$

5: Policy improvement for solving $u_l^{j(i+1)}$ and $v_l^{j(i+1)}$

$$u_l^{j(i+1)} = -\frac{1}{2}R_l^{-1}g^T(x_l)\nabla V_l^{ji}(x_l), \tag{4.17}$$

$$v_l^{j(i+1)} = \frac{1}{2}S_l^{-1}h^T(x_l)\nabla V_l^{ji}(x_l). \tag{4.18}$$

6: Stop if $\|\nabla V_l^{ji} - \nabla V_l^{j(i-1)}\| \leq \varepsilon_1$, where $\|\cdot\|$ denotes 2-norm, then set $\nabla V_l^j(x_l) = \nabla V_l^{ji}(x_l)$, $u_l^j = u_l^{ji}$ and $v_l^j = v_l^{ji}$. Otherwise, set $i \leftarrow i + 1$ and go to Step 4.

7: **State-penalty weight Q_l^{j+1} update using the expert's demonstration u_e^***

$$\begin{aligned} q^T(x_l)Q_l^{j+1}q(x_l) = {} & u_e^{*T}R_l u_e^* - 2u_e^{*T}R_l u_l^j + (v_l^j)^T S_l v_l^j \\ & - (\nabla V_l^j(x_l))^T \left(f(x_l) + g(x_l)u_l^j + h(x_l)v_l^j\right). \end{aligned} \tag{4.19}$$

8: **Stop if** $\|Q_l^{j+1} - Q_l^j\| \leq \varepsilon_2$. Otherwise, set $u_l^{(j+1)0} = u_l^j$ and $j \leftarrow j + 1$, then go to Step 3.

Remark 4.2. *In Algorithm 4.1, the inner loop consists of two-player zero-sum games for adversarial equilibrium. That is, standard two-player zero-sum games are solved as subproblems in inverse RL for Adversarial Apprentice Game. Therefore, compared to [63, 64] where inverse RL is*

dealt as the two-player zero-sum game, the defined Adversarial Apprentice Game is not a standard two-player zero-sum game. Besides, note that the inner iteration loops at outer j loop provide initial stabilizing control input for outer $j+1$ loop, i.e., $u_l^{(j+1)0} = u_l^j$.

Remark 4.3. *It is seen that Algorithm 4.1 has less complexity than directly solving HJI equation (13). In inner loops of Algorithm 4.1, one solves the linear equations* (4.16)*,* (4.17) *and* (4.18) *repetitively. In outer loops, one solves linear coefficient Q_l. Moreover, compared to inverse RL [18, 63, 64, 104] that have stochastic and complex MDPs and require large data set, one has simpler deterministic algorithms and does not need large-scale data.*

4.3.3 Convergence and Stability Analysis

In this subsection, theorem results show the convergence properties and the stability of Algorithm 4.1.

Assumption 4.2. *Assume that the equivalent weight to Q_e in Definition 4.1 exists.*

Theorem 4.1. ***(Convergence properties of Inverse RL Algorithm 4.1)*** *Let Assumptions 4.1-4.2 hold. Select $R_l > 0$, $S_l > 0$, and small $Q_l^0 \geq 0$. Consider the Algorithm 4.1 for solving the Adversarial Apprentice Game. If Algorithm 4.1 is convergent, then the learner has $(x_l, u_l^j) = (x_e, u_e^*)$ for $j \to \infty$ with actual $v_l = v_e$ in* (4.1) *and* (4.9)*, where u_l^j and u_e^* are given by* (4.17) *and* (4.4)*, respectively. The state-penalty weight Q_l^j, $j = 0, 1, \cdots$ converges to the weight Q_l^∞ which is equivalent to Q_e.*

Proof. First, note that if the initial state-penalty weight Q_l^0 is semi-positive definite, then the nonlinear Lyapunov function (4.16) has a unique and positive-definite solution at step 4. With the step 5 for policy improvement, there exists a converged solution set (u_l^0, v_l^0, V_l^0). These two steps are optimal control learning (also called PI) and are proven to be convergent [5].

Then, at the step 7 in Algorithm 4.1, the state-penalty weight is revised as Q_l^1, i.e.,

$$q^T(x_l)Q_l^{j+1}q(x_l) = u_e^{*T}R_l u_e^* - 2u_e^{*T}R_l u_l^j + (v_l^j)^T S_l v_l^j - \left(\nabla V_l^j(x_l)\right)^T \left(f(x_l) + g(x_l)u_l^j + h(x_l)v_l^j\right). \tag{4.20}$$

Given the state-penalty weight Q_l^j, the optimal control learning (steps 4 and 5) gives

$$0 = q^T(x_l)Q_l^j q(x_l) + (u_l^j)^T R_l u_l^j - (v_l^j)^T S_l v_l^j + \left(\nabla V_l^j(x_l)\right)^T \left(f(x_l) + g(x_l)u_l^j + h(x_l)v_l^j\right). \tag{4.21}$$

Substituting (4.21) into (4.20) yields

$$\begin{aligned} q^T(x_l)Q_l^{j+1}q(x_l) &= u_e^{*T}R_l u_e^* - 2u_e^{*T}R_l u_l^j + (u_l^j)^T R_l u_l^j + Q_l^j(x_l) \\ &= (u_e^* - u_l^j)^T R_l (u_e^* - u_l^j) + q^T(x_l)Q_l^j q(x_l) \\ &\geq 0. \end{aligned} \tag{4.22}$$

This shows that at any outer j loop, Q_l^j is semi-positive definite, where $j = 0, 1, \cdots$. The optimal control learning has a convergent solution set (u_l^j, v_l^j, V_l^j). Furthermore, $\|Q_l^j\|$ increases as j increases.

Denote $\Delta_l^j \triangleq (u_e^* - u_l^j)^T R (u_e^* - u_l^j)$. It is seen from Definition 4.1 that there exist different (Q_l^∞, R_l) from (Q_e, R_e) that result in $u_l^\infty = u_e^*$. Also, it is known that u_l^j is uniquely determined by Q_l^j. If the algorithm is convergent, given a small initial Q_l^0 such that $Q_l^0 \leq Q_l^\infty$, Q_l^j increases to Q_l^∞ and u_l^k converges to u_e^*. Then, $\|\Delta_l^j\|$ converges to 0. This means that as $j \to \infty$, one has $\|\Delta_i^\infty(x)\| = 0$, $u_l^j \to u_e^*$, $Q_l^j \to Q_l^\infty$ and

$$u_l^\infty = u_e^*. \tag{4.23}$$

This should hold at the same time instant. It follows from Assumption 4.1 that two agents have the same initial states. Then, (4.23) can obtain $u_l^\infty(t_0) = u_e^*(t_0)$ if one uses the measurements

starting from the initial state. When two systems have the same control policy and state at initial time instant, one further concludes $x_l = x_e$ for all time with $v_l = v_e$.

It is seen that the convergent behaviors $(x_l, u_l^\infty) = (x_e, u_e^*)$ and $Q_l^j \to Q_l^\infty$ are achieved simultaneously. Then, one rewrites the learner's HJI equation (4.14) as

$$\begin{aligned} 0 = {} & q^T(x_e) Q_l^\infty q(x_e) - \frac{1}{4} \nabla V_l^{\infty T}(x_e) g(x_e) R_l^{-1} g^T(x_e) \nabla V_l^\infty(x_e) \\ & + \frac{1}{4} \nabla V_l^{\infty T}(x_e) h(x_e) S_l^{-1} h^T(x_e) \nabla V_l^\infty(x_e) + \nabla V_l^{\infty T}(x_e) f(x_e), \end{aligned} \tag{4.24}$$

which gives the converged $V_l^\infty(x_e)$ with Q_l^∞, such that

$$-\frac{1}{2} R_l^{-1} g^T(x_e) \nabla V_l^\infty(x_e) = -\frac{1}{2} R_e^{-1} g^T(x_e) \nabla V_e^*(x_e). \tag{4.25}$$

Therefore, one has the convergence $Q_l^j \to Q_l^\infty$ and $(x_l, u_l^j) \to (x_e, u_e^*)$. However, one cannot guarantee $Q_l^\infty = Q_e$ because R_l and S_l can be different from R_e and S_e, respectively. Q_l^∞ is then the equivalent weight to Q_e as shown in Definition 4.1. □

Theorem 4.2. ***(Non-uniqueness of** Q_l^∞**)*** *Let Q_l^j and ∇V_l^j in Algorithm 4.1 converge to Q_l^∞ and ∇V_l^∞, respectively. Then, Q_l^∞ satisfies*

$$\begin{aligned} & q^T(x_e)(Q_l^\infty - Q_e) q(x_e) \\ & = \frac{1}{4} \nabla V_l^{\infty T}(x_e) g(x_e) R_l^{-1} (R_l - R_e) R_l^{-1} g^T(x_e) \nabla V_l^\infty(x_e) \\ & \quad + \frac{1}{4} \nabla V_e^{*T}(x_e) h(x_e) S_e^{-1} h^T(x_e) \nabla V_e^*(x_e) - \frac{1}{4} \nabla V_l^{\infty T}(x_e) h(x_e) S_l^{-1} h^T(x_e) \nabla V_l^\infty(x_e) \\ & \quad + (\nabla V_e^{*T}(x_e) - \nabla V_l^{\infty T}(x_e)) f(x_e), \end{aligned} \tag{4.26}$$

where ∇V_e^* is uniquely solved by (4.6)*, and $\nabla V_l^\infty(x_e)$ satisfies*

$$g^T(x_e) \nabla V_l^\infty(x_e) = R_l R_e^{-1} g^T(x_e) \nabla V_e^*(x_e), \tag{4.27}$$

which implies that Q_l^∞ may not be unique.

Proof. As shown in Theorem 4.1, when the learner obtains the convergence $(x_l, u_l^\infty) = (x_e, u_e^*)$, one follows from (4.25) and has

$$g^T(x_e)\nabla V_l^\infty(x_e) = R_l R_e^{-1} g^T(x_e)\nabla V_e^*(x_e). \tag{4.28}$$

Then, subtracting (4.24) from (4.6) yields

$$\begin{aligned}
&q^T(x_e)(Q_l^\infty - Q_e)q(x_e) \\
&= \frac{1}{4}\nabla V_l^{\infty T}(x_e) g(x_e) R_l^{-1}(R_l - R_e)R_l^{-1} g^T(x_e)\nabla V_l^\infty(x_e) \\
&\quad + \frac{1}{4}\nabla V_e^{*T}(x_e) h(x_e) S_e^{-1} h^T(x_e)\nabla V_e^*(x_e) - \frac{1}{4}\nabla V_l^{\infty T}(x_e) h(x_e) S_l^{-1} h^T(x_e)\nabla V_l^\infty(x_e) \\
&\quad + (\nabla V_e^{*T}(x_e) - \nabla V_l^{\infty T}(x_e)) f(x_e).
\end{aligned} \tag{4.29}$$

In (4.28), if one lets $g^T(x_e)X = R_l R_e^{-1} g^T(x_e)\nabla V_e^*(x_e)$, there will be infinite number of solutions for X unless rank$(g(x_e)) = n$. This means that one may find many solutions of $\nabla V_l^\infty(x_e)$ that make (4.28) hold. However, one cannot guarantee rank$(g(x_e)) = n$. Thus, one may obtain non-unique $\nabla V_l^\infty(x_e)$ that is different from $\nabla V_e^*(x_e)$. Besides, as stated in Remark 4.1, R_l can be different from R_e and S_l can be different from S_e. From (4.29), the equivalent weight $Q_l^\infty - Q_e$ may be nonzero and Q_l^∞ may be different from Q_e. There may be infinite number of solutions for Q_l. All possible and non-unique solutions for Q_l^∞ satisfy (4.29) with ∇V_l^∞ satisfying (4.28). □

Theorem 4.3. ***(Stability of the learner dynamics using Algorithm 4.1).*** *Give the learner dynamics* (4.9) *with the cost function* (4.10). *Give small $Q_l^0 \geq 0$, $R_l > 0$ and $S_l > 0$, and use Algorithm 4.1 for the Adversarial Apprentice Game. Then, at each iteration of Algorithm 4.1, the learner dynamics* (4.9) *is asymptotically stable when the adversarial input $v_l = 0$ in* (4.9).

Proof. To prove the stability of (4.9) with $v_l = 0$, one proves that $\dot{V}_l^j(x_l) \leq 0$ for any $j \geq 0$ with $v_l = 0$.

It follows from (4.14) that at outer j loop, one has

$$
\begin{aligned}
0 &= q^T(x_l)Q_l^j q(x_l) - \frac{1}{4}(\nabla V_l^j(x_l))^T g(x_l)R_l^{-1}g^T(x_l)\nabla V_l^j(x_l) \\
&\quad + \frac{1}{4}(\nabla V_l^j(x_l))^T h(x_l)S_l^{-1}h^T(x_l)\nabla V_l^j(x_l) + (\nabla V_l^j(x_l))^T f(x_l) \\
&= q^T(x_l)Q_l^{j-1}q(x_l) + (u_e^* - u_l^{j-1})^T R_l(u_e^* - u_l^{j-1}) \\
&\quad + \frac{1}{4}(\nabla V_l^j)^T(x_l)g(x_l)R_l^{-1}g^T(x_l)\nabla V_l^j(x_l) + \frac{1}{4}(\nabla V_l^j(x_l))^T \times h(x_l)S_l^{-1}h^T(x_l)\nabla V_l^j(x_l) \\
&\quad + (\nabla V_l^j)^T(x_l)(f(x_l) - \frac{1}{2}R_l^{-1}g^T(x_l)\nabla V_f l^j(x_l)) \\
&= q^T(x_l)Q_l^{j-1}q(x_l) + (u_e^* - u_l^{j-1})^T R_l(u_e^* - u_l^{j-1}) + (u_l^j)^T R_l u_l^j + (v_l^j)^T S_l v_l^j + \dot{V}_l^j(x_l). \quad (4.30)
\end{aligned}
$$

With $Q_l^{j-1} \geq 0$ proven in Theorem 4.1, it follows from (4.30) that $\dot{V}_l^j(x_l) \leq 0$. In addition, note that $\dot{V}_l^j(x_l) = 0$ only when $x_l = 0$. Thus, the learner is asymptotically stable with $v_l^j = 0$ during iterations of RL Algorithm 4.1. □

Remark 4.4. *Note that Algorithm 4.1 only learns the equivalent weight to Q_e without learning the control input weight of the expert. As shown in Assumption 4.1, $R_l > 0$ is arbitrarily selected by the learner and can be different from $R_e > 0$. With the arbitrarily selected $R_l > 0$, Theorem 4.2 shows that the learner still obtains the expert control policy in (20) and Q_l^∞ is characterized by (19). Also, the convergence of Algorithm 4.1 is then guaranteed by Theorem 4.1.*

Remark 4.5. *If the convergence (x_l, u_l^*) = (x_e, u_e^*) is obtained, then* (4.15) *and the corresponding* (4.9)-(4.14) *are standard inverse optimal control [31]. The learner obtains the inverse optimality.*

4.3.4 Implementing Model-based Inverse RL Algorithm 4.1 via NNs

This subsection now develops a NN-based method to compute inverse RL Algorithm 4.1 online. First, the learner approximates the performance cost function V_l^{ji} at step 4 of Algorithm 4.1 given the current Q_l^j, u_l^{ji} and v_l^{ji}. Then, $u_l^{j(i+1)}$ and $v_l^{j(i+1)}$ update using V_l^{ji} at step 5. Third, the learner updates Q_l^{j+1} at step 7 using the expert's control input u_e^* and the converged solutions from inner i-th iteration loop, i.e., u_l^j, v_l^j and V_l^j.

According to Weierstrass approximation Theorem [116] using polynomial approximation, there exists an activation vector function $\varphi(x_l) = [\varphi_1(x_l)\ \varphi_2(x_l)\ \cdots\ \varphi_{N_1}(x_l)]^T$ with N_1 hidden-layer

neurons, such that the cost function $V_l^{ji}(x_l)$ and its gradient can be uniformly approximated as

$$\hat{V}_l^{ji}(x_l) = (C_l^{ji})^T \varphi(x_l), \tag{4.31}$$

$$\nabla \hat{V}_l^{ji}(x_l) = \nabla \varphi^T(x_l) C_l^{ji}, \tag{4.32}$$

with $C_l^{ji} = \begin{bmatrix} C_{l1}^{ji} & C_{l2}^{ji} & \cdots & C_{lN_1}^{ji} \end{bmatrix}^T \in \mathbb{R}^{N_1}$ the weight vector and $\nabla\varphi(x_l) = [\nabla\varphi_1(x_l)\ \nabla\varphi_2(x_l)$ $\cdots\ \nabla\varphi_{N_1}(x_l)]^T$ the Jacobian of $\varphi(x_l)$. Given NNs (4.31) and (4.32), (4.16) is rewritten as

$$\begin{aligned} e_l^{ji}(t) = {} & q^T(x_l)\hat{Q}_l^j q(x_l) + (\hat{u}_l^{ji})^T R_l \hat{u}_l^{ji} - (\hat{v}_l^{ji})^T S_l \hat{v}_l^{ji} \\ & + (C_l^{ji})^T \nabla\varphi(x_l) \left(f(x_l) + g(x_l)\hat{u}_l^{ji} + h(x_l)\hat{v}_l^{ji} \right), \end{aligned} \tag{4.33}$$

where $e_l^{ji}(t)$ is the approximation error and is forced to be zero in average sense [5, 68] to find C_l^{ji}; $\hat{Q}_l^j$, $\hat{u}_l^{ji}$ and $\hat{v}_l^{ji}$ are approximation values using NN (4.31). It follows from (4.31) that the NN weight C_l^{ji} has N_1 unknown constants. In order to solve the unique C_l^{ji}, the batch least square (BLS) method is used to construct $\bar{N}_1 \geq N_1$ equations for (4.33) from $\bar{N}_1$ different time points. The continuous-time data can be well approximated by discretization with a small interval $T > 0$. Define

$$\Phi_l^{ji} = \begin{bmatrix} \alpha(x_l)|_T \\ \alpha(x_l)|_{2T} \\ \vdots \\ \alpha(x_l)|_{\bar{N}_1 T} \end{bmatrix}, \Psi_l^{ji} = \begin{bmatrix} \psi(x_l)|_T \\ \psi(x_l)|_{2T} \\ \vdots \\ \psi(x_l)|_{\bar{N}_1 T} \end{bmatrix}, \tag{4.34}$$

where

$$\begin{aligned} \alpha(x_l)|_{s_1 T} &= \nabla\varphi(x_l) \left(f(x_l) + g(x_l)\hat{u}_l^{ji} + h(x_l)\hat{v}_l^{ji} \right) \big|_{s_1 T}, \\ \psi(x_l)|_{s_1 T} &= \left(-q^T(x_l) Q_l^j q^T(x_l) - (\hat{u}_l^{ji})^T R_l \hat{u}_l^{ji} + (\hat{v}_l^{ji})^T S_l \hat{v}_l^{ji} \right) \big|_{s_1 T}, \\ & s_1 \in \{1, 2, \cdots, \bar{N}_1\}. \end{aligned}$$

Thus, C_l^{ji} is uniquely solved by

$$C_l^{ji} = \left((\Phi_l^{ji})^T \Phi_l^{ji}\right)^{-1} (\Phi_l^{ji})^T \Psi_l^{ji}. \tag{4.35}$$

The optimal control input (4.17) and the worst-case adversarial input (4.18) are then respectively approximated as

$$\hat{u}_l^{j(i+1)} = -\frac{1}{2} R_l^{-1} g^T(x_l) \nabla\varphi(x_l) C_l^{ji}, \tag{4.36}$$

$$\hat{v}_l^{j(i+1)} = \frac{1}{2} S_l^{-1} h^T(x_l) \nabla\varphi(x_l) C_l^{ji}. \tag{4.37}$$

When the solution set (C_l^{ji} $\hat{u}_l^{j(i+1)}$ $\hat{v}_l^{j(i+1)}$) converges to a set (C_l^{j} $\hat{u}_l^{j}$ $\hat{v}_l^{j}$), according to (4.15), the estimated state-penalty weight $\hat{Q}_l^{j+1}$ is then updated as

$$\begin{aligned} q^T(x_l)\hat{Q}_l^{j+1} q(x_l) = {} & u_e^{*T} R_l u_e^* - 2u_e^{*T} R_l \hat{u}_l^j + (\hat{v}_l^j)^T S_l \hat{v}_l^j \\ & - (C_l^j)^T \nabla\varphi(x_l) \left(f(x_l) + g(x_l)\hat{u}_l^j + h(x_l)\hat{v}_l^j\right). \end{aligned} \tag{4.38}$$

The matrix $\hat{Q}_l^{j+1}$ can be computed by (4.38) using BLS. $\hat{Q}_l^{j+1}$ has $(n+1)n/2$ unknown parameters. $\hat{Q}_l^{j+1}$ is uniquely solved by constructing $\bar{N}_2 \geq (n+1)n/2$ equations. Define

$$\Gamma_l = \begin{bmatrix} \text{vecv}(q(x_l) \otimes q(x_l))^T|_T \\ \text{vecv}(q(x_l) \otimes q(x_l))^T|_{2T} \\ \vdots \\ \text{vecv}(q(x_l) \otimes q(x_l))^T|_{\bar{N}_2 T} \end{bmatrix}, \; \Delta_l^j = \begin{bmatrix} \delta_l^j|_T \\ \delta_l^j|_{2T} \\ \vdots \\ \delta_l^j|_{\bar{N}_2 T} \end{bmatrix}, \tag{4.39}$$

where

$$\begin{aligned}
\text{vecv}(s\otimes s) &\triangleq [s_1^2\ s_1s_2\ \cdots\ s_1s_n\ s_2^2\ \cdots\ s_{n-1}s_n\ s_n^2]^T, s\in\mathbb{R}^n,\\
\delta_l^j &= \Big(-(C_l^j)^T\nabla\varphi(x_l)\big(f(x_l)+g(x_l)\hat{u}_l^j\\
&\qquad +h(x_l)\hat{v}_l^j\big)+u_e^{*T}R_lu_e^*-2u_e^{*T}R_l\hat{u}_l^j+(\hat{v}_l^j)^TS_l\hat{v}_l^j\Big)|_{s_2T},\\
s_2 &\in\{1,2,\cdots,\bar{N}_2\}.
\end{aligned}$$

Then, $\hat{Q}_l^{j+1}$ is uniquely solved by

$$\text{vecm}(\hat{Q}_l^{j+1})=\left(\Gamma_l^T\Gamma_l\right)^{-1}\Gamma_l^T\Delta_l^j, \tag{4.40}$$

where $\text{vecm}(\hat{Q}_l^{j+1})\triangleq[\hat{Q}_{11}^{j+1}\ 2\hat{Q}_{12}^{j+1}\ \cdots\ 2\hat{Q}_{1n}^{j+1}\ \hat{Q}_{22}^{j+1}\ 2\hat{Q}_{23}^{j+1}\ \cdots 2\hat{Q}_{(n-1)n}^{j+1}\ \hat{Q}_{nn}^{j+1}]^T$.

Now, a NN-based inverse RL Algorithm 4.2 is summed up below to implement the inverse RL Algorithm 4.1.

Algorithm 4.2 Model-based Inverse RL Algorithm via NNs for Adversarial Apprentice Game

1: **Initialization**: select $\hat{Q}_l^0\geq 0$, $R_l>0$, $S_l>0$, initial stabilizing $\hat{u}_l^{00}$, and small thresholds ε_1, ε_2. Set $j=0$.

2: **Outer j iteration loop based on inverse optimal control**

3: **Inner i iteration loop using optimal control:** given j, set $i=0$, use initial stabilizing $\hat{u}_l^{j0}$.

4: Policy evaluation for solving C_l^{ji} by (4.35).

5: Policy improvement for solving $\hat{u}_l^{j(i+1)}$ by (4.36) and $\hat{v}_l^{j(i+1)}$ by (4.37).

6: Stop if $\|C_l^{ji}-C_l^{j(i-1)}\|\leq\varepsilon_1$, then set $C_l^j=C_l^{ji}$, $\hat{u}_l^j=\hat{u}_l^{ji}$ and $\hat{v}_l^j=\hat{v}_l^{ji}$. Otherwise, set $i\leftarrow i+1$ and go to Step 4.

7: **State-penalty weight $\hat{Q}_l^{j+1}$ update by** (4.40) **using the expert's demonstration** u_e^*.

8: **Stop if** $\|\hat{Q}_l^{j+1}-\hat{Q}_l^j\|\leq\varepsilon_2$. Otherwise, set $\hat{u}_l^{(j+1)0}=\hat{u}_l^j$ and $j\leftarrow j+1$, then go to Step 3.

The next result shows the same convergence between Algorithms 4.2 and 4.1.

Theorem 4.4. ***(Convergence of Algorithm 4.2).*** *Algorithm 4.2 converges to Algorithm 4.1 and obtains a convergence* $(x_l,\, \hat{u}_l^j) \to (x_e,\, u_e^*)$ *as* $j \to \infty$.

Proof. In inner iteration loop, given j, $\hat{Q}_l^j \geq 0, R_l > 0, S_l > 0$, the NN weight C_l^{ji} can be uniquely determined if rank$((\Phi_l^{ij})^T \Phi_l^{ij}) \geq N_i$. This can be satisfied by letting the number of collected data in (4.34) satisfy $\bar{N}_1 \geq N_1$. Thus, $\hat{V}_l^{ji}$ is solved by BLS (4.35) and converges to V_l^{ji} solved by (4.16). It has been proven in [5] that $\hat{V}_l^{ji}$ in (4.31) uniformly approximates to V_l^j. By repeating (4.35)-(4.37), when C_l^{ji} converges to C_l^j, the learner has the unique inputs $\hat{u}_l^j$ and $\hat{v}_l^j$. These approximated inputs uniformly converges to (4.17)-(4.18), i.e., $\hat{u}_l^{ji} \to u_l^{ji}$ and $\hat{v}_l^{ji} \to v_l^{ji}$.

In outer iteration loop, the learner can use (4.40) to uniquely solve the $\hat{Q}_l^{j+1}$ with the condition of rank$(\Gamma_l^T \Gamma_l) \geq (n+1)n/2$. This can be satisfied by letting the number of collected data have $\bar{N}_2 \geq (n+1)n/2$ in (4.39). Due to the convergence $\hat{V}_l^{ji} \to V_l^{ji}$, $\hat{u}_l^{ji} \to u_l^{ji}$ and $\hat{v}_l^{ji} \to v_l^{ji}$, one thus has that $\hat{Q}_l^{j+1}$ is uniquely solved by (4.40) and converges to Q_l^{j+1}.

It is seen that (4.35)-(4.37) and (4.38) in Algorithm 4.2 are rigorously derived from (4.16)-(4.19) of Algorithm 4.1 using NN (4.31). Algorithm 4.2 obtains unique solutions in (4.35)-(4.37) and (4.38). These solutions converge to the solutions of (4.16)-(4.19) in Algorithm 4.1. One concludes that Algorithm 4.2 converges to Algorithm 4.1. Thus, the learner obtains a convergence $(x_l,\, \hat{u}_l^j) \to (x_e,\, u_e^*)$ with $v_l = v_e$ as $j \to \infty$. □

4.4 Model-free Inverse RL

Inverse RL Algorithms 4.1 and 4.2 in Section 4.3 require the knowledge of system dynamics f, g and h to solve Adversarial Apprentice Games. This section develops an online model-free off-policy integral inverse RL algorithm without using f, g and h. Then, this model-free inverse RL algorithm is implemented using NNs in Section 4.4.2.

4.4.1 Model-free Integral Inverse RL Algorithm

First, one uses the off-policy integral RL [73, 120] in inner i iteration loop of Algorithm 4.1,

which allows to find model-free equations equivalent to (4.16)-(4.18). First, rewrite (4.9) as

$$\dot{x}_l = f(x_l) + g(x_l)u_l^{ji} + h(x_l)v_l^{ji} + g(x_l)(u_l - u_l^{ji}) + h(x_l)(v_l - v_l^{ji}), \tag{4.41}$$

where $u_l^{ji} \in \mathbb{R}^m$ and $v_l^{ji} \in \mathbb{R}^k$ are inputs updated at inner i iteration given j. Differentiating V_l^{ji} along with (4.41) yield

$$\begin{aligned}\dot{V}_l^{ji} &= (\nabla V_l^{ji})^T(f + gu_l^{ji} + hv_l^{ji}) + (\nabla V_l^{ji})^T g(x_l)(u_l - u_l^{ji}) + (\nabla V_l^{ji})^T h(x_l)(v_l - v_l^{ji}) \\ &= -q^T(x_l)Q_l^j q(x_l) - (u_l^{ji})^T R_l u_l^{ji} + (v_l^{ji})^T S_l v_l^{ji} \\ &\quad - 2(u_l^{j(i+1)})^T R_l(u_l - u_l^{ji}) + 2(v_l^{j(i+1)})^T S_l(v_l - v_l^{ji}).\end{aligned} \tag{4.42}$$

Then, integrating both sides of (4.42) from t to $t+T$ yields the off-policy Bellman equation (4.43) (to be presented in Algorithm 4.3), which solves the converged V_l^l, u_l^j and v_l^j given Q_l^j.

Next, the integral RL technique is used in outer j iteration loop of Algorithm 4.1 to find a model-free equation equivalent to (4.19). To update the Q_l^{j+1}, one integrates both sides of (4.19) from t to $t+T$ to yield (4.44) (to be presented in Algorithm 4.3).

The integral inverse RL Algorithm 4.3 is summed up below.

Algorithm 4.3 Model-free Integral Inverse RL Algorithm for Adversarial Apprentice Game

1: **Initialization**: select $Q_l^0 \geq 0$, $R_l > 0$, $S_l > 0$, initial stabilizing u_l^{00}, and small thresholds ε_1, ε_2. Set $j = 0$. Apply stabilizing u_l to the learner dynamics (4.41).

2: **Outer j iteration loop based on inverse optimal control**

3: **Inner i iteration loop using optimal control:** given j, set $i = 0$, use initial stabilizing u_l^{j0}.

4: Off-policy Integral RL for solving V_l^j, u_l^j and v_l^j

$$\begin{aligned} & V_l^{ji}(x_l(t+T)) - V_l^{ji}(x_l(t)) \\ & \quad - \int_t^{t+T} 2\Big(\big(u_l^{j(i+1)}\big)^T R_l \big(u_l - u_l^{ji}\big) - \big(v_l^{j(i+1)}\big)^T S_l \big(v_l - v_l^{ji}\big) \Big) d\tau \\ & = \int_t^{t+T} \Big(- q^T(x_l) Q_l^j q(x_l) - (u_l^{ji})^T R_l u_l^{ji} + (v_l^{ji})^T S_l v_l^{ji} \Big) d\tau. \end{aligned} \tag{4.43}$$

5: Stop if $\|V_l^{ji} - V_l^{j(i-1)}\| \leq \varepsilon_1$, then set $V_l^j = V_l^{ji}$, $u_l^j = u_l^{ji}$ and $v_l^j = v_l^{ji}$. Otherwise, set $i \leftarrow i + 1$ and go to Step 4.

6: **State-penalty weight Q_l^{j+1} update using the expert's demonstration u_e^***

$$\begin{aligned} & \int_t^{t+T} q^T(x_l) Q_l^{j+1} q(x_l) d\tau \\ & = \int_t^{t+T} \Big(u_e^{*T} R_l u_e^* - 2 u_e^{*T} R_l u_l^j + (v_l^j)^T S_l v_l^j \Big) d\tau - V_l^j(x_l(t+T)) + V_l^j(x_l(t)). \end{aligned} \tag{4.44}$$

7: **Stop if** $\|Q_l^{j+1} - Q_l^j\| \leq \varepsilon_2$. Otherwise, set $u_l^{(j+1)0} = u_l^j$ and $j \leftarrow j + 1$, then go to Step 3.

The next theorem shows the convergence of Algorithm 4.3 to Algorithm 4.1.

Theorem 4.5. ***(Convergence of Algorithm 4.3)*** *Algorithm 4.3 converges to Algorithm 4.1 such that the learner achieves a convergence (x_l, u_l^j) $\to$ (x_e, u_e^*) as $j \to \infty$.*

Proof. First, given j, $j = 0, 1, \cdots$, and Q_l^j, one divides both sides of (4.43) by T and takes the

limit to be

$$\lim_{T\to 0}\frac{V_l^{ji}(x_l(t+T))-V_l^{ji}(x_l(t))}{T}-\lim_{T\to 0}\frac{2\int_t^{t+T}\big(u_l^{j(i+1)}\big)^T R_l\big(u_l-u_l^{ji}\big)d\tau}{T}$$
$$+\lim_{T\to 0}\frac{2\int_t^{t+T}\big(v_l^{j(i+1)}\big)^T S_l\big(v_l-v_l^{ji}\big)d\tau}{T}$$
$$+\lim_{T\to 0}\frac{\int_t^{t+T}\big(q^T(x_l)Q_l^j q(x_l)+(u_l^{ji})^T R_l u_l^{ji}-(v_l^{ji})^T S_l v_l^{ji}\big)d\tau}{T}$$
$$=0. \tag{4.45}$$

By LâHopitalâs rule, (4.45) becomes

$$(\nabla V_l^{ji})^T\big(f+gu_l^{ji}+hv_l^{ji}+g(x_l)(u_l-u_l^{ji})+h(x_l)(v_l-v_l^{ji})\big)$$
$$-\big(u_l^{j(i+1)}\big)^T R_l\big(u_l-u_l^{ji}\big)+\big(v_l^{j(i+1)}\big)^T S_l\big(v_l-v_l^{ji}\big)+q^T(x_l)Q_l^j q(x_l)+(u_l^{ji})^T R_l u_l^{ji}-(v_l^{ji})^T S_l v_l^{ji}$$
$$=0. \tag{4.46}$$

Then, submitting $u_l^{j(i+1)}$ in (4.17) and $v_l^{j(i+1)}$ in (4.18) into (4.46) yields (4.16). This implies that (4.43) gives the same solution as the Lyapunov function (4.16) with the inputs (4.17) and (4.18).

Similarly, one divides both the sides of (4.44) by T and takes the limit operation. This yields (4.19) and shows that (4.44) gives the same solution as (4.19). Thus, the learner obtains a convergence $(x_l,\, u_l^j)\to(x_e,\, u_e^*)$ with $v_l=v_e$ as $j\to\infty$. □

Remark 4.6. *For the learner, Algorithm 4.3 is completely model-free in both outer j iteration loop and inner i iteration loop, which is compared with [17, 83] that require the knowledge of system dynamics for reconstructing cost functions. This is also in contrast to the Algorithms 4.1 and 4.2 that require the full knowledge of system dynamics in both iteration loops.*

4.4.2 Implementing Model-free Integral Inverse RL Algorithm 4.3 via NNs

This subsection introduces a NN-based model-free off-policy inverse RL algorithm for implementing the Algorithm 4.3 using online data without knowing any knowledge of system dynam-

ics. Three NN-based approximators are designed for the learner's value function V_l^{ji}, the updated control input $u_l^{j(i+1)}$ and the updated adversarial input $v_l^{j(i+1)}$ in the Bellman equation (4.43) in Algorithm 4.3. Three approximators are

$$\hat{V}_l^{ji} = (C_l^{ji})^T \varphi(x_l), \tag{4.47}$$

$$\bar{u}_l^{j(i+1)} = (W_l^{ji})^T \phi(x_l), \tag{4.48}$$

$$\bar{v}_l^{j(i+1)} = (H_l^{ji})^T \rho(x_l), \tag{4.49}$$

where $\varphi(x_l)$ in (4.31), $\phi(x_l) = [\phi_1(x_l)\ \phi_2(x_l)\ \cdots\ \phi_{N_2}(x_l)]^T$ and $\rho(x_l) = \left[\rho_1(x_l)\ \rho_2(x_l)\ \cdots\ \rho_{N_3}(x_l)\right]^T$ are activation vector functions of three NNs, respectively. Moreover, $C_l^{ji} \in \mathbb{R}^{N_1}$, $W_l^{ji} \in \mathbb{R}^{m\times N_2}$ and $H_l^{ji} \in \mathbb{R}^{k\times N_3}$. N_1 in (4.31), N_2 and N_3 are hidden-layer neuron numbers of three NNs, respectively.

Define $u_l - \bar{u}_l^{ji} \triangleq \left[\tilde{u}_{l,1}^{ji}\ \tilde{u}_{l,2}^{ji}\ \cdots\ \tilde{u}_{l,m}^{ji}\right]^T$ and $v_l - \bar{v}_l^{ji} \triangleq \left[\tilde{v}_{l,1}^{ji}\ \tilde{v}_{l,2}^{ji}\ \cdots\ \tilde{v}_{l,k}^{ji}\right]^T$. Assume that weights R_l and S_l are given in the form of $R_l = \text{diag}\{r_1,\ r_2,\ \cdots,\ r_m\}^T$ and $S_l = \text{diag}\{s_1,\ s_2,\ \cdots,\ s_k\}^T$, respectively. Then, together with (4.47)-(4.49), (4.43) is expressed as

$$\begin{aligned}
&(C_l^{ji})^T \left[\varphi(x_l(t+T) - \varphi(x_l(t))\right] \\
&\quad + 2\sum_{h=1}^{m} r_h (W_{l,h}^{ji})^T \int_t^{t+T} \phi(x_l)\tilde{u}_{l,h}^{ji} d\tau - 2\sum_{p=1}^{k} s_p (H_{l,p}^{ji})^T \int_t^{t+T} \rho(x_l)\tilde{v}_{1,p}^{ji} d\tau \\
&= \bar{e}_l^{ji}(t) - \int_t^{t+T} \Big(q^T(x_l)\hat{Q}_l^j q(x_l) + (\bar{u}_l^{ji})^T R_l \bar{u}_l^{ji} - (\bar{v}_l^{ji})^T S_l \bar{v}_l^{ji}\Big)\, d\tau,
\end{aligned} \tag{4.50}$$

where $\bar{e}_l^{ji}(t)$ is the Bellman approximation error and is forced to be zero in some average sense to find C_l^{ji}, W_l^{ji} and H_l^{ji} [5, 68]. One uses BLS method to solve the unique solution set $\Big(C_l^{ji}\ W_{l,1}^{ji}\ \cdots\ W_{l,m}^{ji}\ H_{l,1}^{ji}\ \cdots\ H_{l,k}^{ji}\Big)$ given Q_l^j. This solution set provides information for the update of $\hat{Q}_l^{j+1}$

in (4.44). Thus, one defines

$$\Sigma_l^{ji} = \begin{bmatrix} \int_t^{t+T} \sigma_l^{ji} d\tau \\ \int_{t+T}^{t+2T} \sigma_l^{ji} d\tau \\ \vdots \\ \int_{t+(\iota-1)T}^{t+\iota T} \sigma_l^{ji} d\tau \end{bmatrix}, \Pi_l^{ij} = \begin{bmatrix} \pi_{x_l}^1 & \pi_{u_l}^1 & \pi_{v_l}^1 \\ \pi_{x_l}^2 & \pi_{u_l}^2 & \pi_{v_l}^2 \\ \vdots & \vdots & \vdots \\ \pi_{x_l}^\iota & \pi_{u_l}^\iota & \pi_{v_l}^\iota \end{bmatrix}, \tag{4.51}$$

where

$$\begin{aligned}
\sigma_l^{ji} &= -q^T(x_l)\hat{Q}_l^j q^T(x_l) - (\bar{u}_l^{ji})^T R_l \bar{u}_l^{ji} + (\bar{v}_l^{ji})^T S_l \bar{v}_l^{ji}, \\
\pi_{x_l}^\iota &= \varphi^T(x_l(t+\iota T)) - \varphi^T(x_l(t+\iota T - T)), \\
\pi_{u_l}^\iota &= [r_1 \int_{t+\iota T-T}^{t+\iota T} \phi^T(x_l)\tilde{u}_{l,1}^{ji} d\tau \ \cdots \ r_m \int_{t+\iota T-T}^{t+\iota T} \phi^T(x_l)\tilde{u}_{l,m}^{ji} d\tau], \\
\pi_{v_l}^\iota &= [s_1 \int_{t+\iota T-T}^{t+\iota T} \rho^T(x_l)\tilde{v}_{l,1}^{ji} d\tau \ \cdots \ s_m \int_{t+\iota T-T}^{t+\iota T} \rho^T(x_l)\tilde{v}_{l,k}^{ji} d\tau].
\end{aligned}$$

The unknown parameters in (4.50) can be uniquely solved by using BLS when $(\Pi_l^{ji})^T \Pi_l^{ji}$ has full rank. It is required to satisfy a persistent excitation (PE) condition which needs ι groups of data collection. The positive integer ι is no less than the number of unknown parameters in (4.50), i.e., $\iota \geq N_1 + m \times N_2 + k \times N_3$. Then, the unknown weights C_l^{ji}, W_l^{ji} and H_l^{ji} in (4.50) are uniquely solved by

$$\left[(C_l^{ji})^T \ (W_{l,1}^{ji})^T \ \cdots \ (W_{l,m}^{ji})^T \ (H_{l,1}^{ji})^T \ \cdots \ (H_{l,k}^{ji})^T\right]^T = ((\Pi_l^{ji})^T \Pi_l^{ji})^{-1} (\Pi_l^{ji})^T \Sigma_l^{ji}. \tag{4.52}$$

When $\left(C_l^{ji} \ W_l^{ji} \ H_l^{ji}\right)$ converges to $\left(C_l^j \ W_l^j \ H_l^j\right)$, given the expert's control input u_e^* and using this convergent solution set, the learner solves $\hat{Q}_l^{j+1}$ by

$$\begin{aligned}
\int_t^{t+T} q^T(x_l)\hat{Q}_l^{j+1} q(x_l) = \int_t^{t+T} &\Big(u_e^{*T} R_l u_e^* - 2u_e^{*T} R_l W_l^j \phi + \rho^T H_l^j S_l (H_l^j)^T \rho\Big) d\tau \\
&- (C_l^j)^T \Big(\varphi(x_l(t+T) - \varphi(x_l(t))\Big),
\end{aligned} \tag{4.53}$$

where the unique $\hat{Q}_l^{j+1}$ is computed by constructing $\bar{N}_2 \geq (n+1)n/2$ equations. Define

$$(Y_l^j)^T = \begin{bmatrix} y_l^j(T) & y_l^j(2T) & \cdots & y_l^j(\bar{N}_2 T) \end{bmatrix}, \tag{4.54}$$
$$(\bar{\Gamma}_l)^T = \begin{bmatrix} \int_t^{t+T} \gamma_l d\tau & \int_{t+T}^{t+2T} \gamma_l d\tau & \cdots & \int_{t+(\bar{N}_2-1)T}^{t+\bar{N}_2 T} \gamma_l d\tau \end{bmatrix},$$

where

$$\begin{aligned} y_l^j(s_2 T) = & \int_{t+(s_2-1)T}^{t+s_2 T} \big(u_e^{*T} R_l u_e^* - 2u_e^{*T} R_l W_l^j \phi + \rho^T H_l^j S_l (H_l^j)^T \rho\big) d\tau \\ & - (C_l^j)^T \big(\varphi(x_l(t+s_2 T)) - \varphi(x_l(t+(s_2-1)T))\big), \\ \gamma_l = & \ \text{vecv}(q(x_l) \otimes q(x_l))^T, \\ s_2 \in & \{1, 2, \cdots, \bar{N}_2\} \end{aligned}$$

Then, $\hat{Q}_l^{j+1}$ is solved by

$$\text{vecm}(\hat{Q}_l^{j+1}) = \left(\bar{\Gamma}_l^T \bar{\Gamma}_l\right)^{-1} \bar{\Gamma}_l^T Y_l^j. \tag{4.55}$$

Based on above derivation, an online NN-based model-free integral inverse RL Algorithm 4.4 is summed up as follows.

Algorithm 4.4 Model-free Integral Inverse RL Algorithm via NNs for Adversarial Apprentice Game

1: **Initialization**: select $\hat{Q}_l^0 \geq 0$, stabilizing $\bar{u}_l^{00}$, and small thresholds ε_1, ε_2. Apply stabilizing u_l to (4.41). Set $j = 0$.

2: **Outer j iteration loop based on inverse optimal control**

3: **Inner i iteration loop using optimal control:** given j, set $i = 0$, use initial stabilizing $\bar{u}_l^{j0}$.

4: Off-policy integral RL for solving C_l^{ji}, W_l^{ji} and H_l^{ji} by (4.52).

5: Stop if $\|\hat{V}_l^{ji} - \hat{V}_l^{j(i-1)}\| \leq \varepsilon_1$, then set $(C_l^j\ W_l^j\ H_l^j)$ =$(C_l^{ji}\ W_l^{ji}\ H_l^{ji})$. Otherwise, set $i \leftarrow i+1$ and go to Step 4.

6: **State-penalty weight $\hat{Q}_l^{j+1}$ update by** (5.44) **using the expert's demonstration** u_e^*.

7: **Stop if** $\|\hat{Q}_l^{j+1} - \hat{Q}_l^j\| \leq \varepsilon_2$. Otherwise, set $\bar{u}_l^{(j+1)0} = \bar{u}_l^j$ and $j \leftarrow j+1$ and go to Step 3.

Remark 4.7. *The convergence of Algorithm 4.4 to Algorithm 4.1 can be obtained by proving the convergence of Algorithm 4.4 to Algorithm 4.3. This can be referred to the Theorem 4.5. Then, it follows from Theorem 4.5 that Algorithm 4.3 convergences to Algorithm 4.1. Thus, one concludes that Algorithm 4.4 convergences to Algorithm 4.1.*

4.5 Simulation Studies

In this section, two examples, i.e., an example of linear expert-learner system and an example of nonlinear expert-learner system are studied to verify the effectiveness of the proposed inverse RL methods. In addition, the proposed methods are compared with the off-policy tracking method in [73] for the imitation performance.

4.5.1 Example 1: Linear Systems

Consider the linear learner and expert systems with identical system dynamics as

$$\dot{x}_l = Ax_l + Bu_l + Dv_l,$$
$$\dot{x}_e = Ax_e + Bu_e + Dv_e,$$

where the system dynamics matrices A, B and D are given by

$$A = \begin{bmatrix} -3 & -2 \\ 2 & -3 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \quad D = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Table 4.1 gives the parameters of the learner and expert systems in an Adversarial Apprentice Game. In the table, I_n denotes the n-dimensional identity matrix.

Given the expert system and other parameters in Table 4.1, one refers to the optimal control technique [54] and derives the expert's optimal feedback control input as

$$u_e^* \triangleq -K_e x_e = \begin{bmatrix} 0.2258 & -2.0640 \end{bmatrix} x_e,$$

Table 4.1: Parameter Setting

Parameters	Expert	Learner
State-Penalty Weight	$Q_e = 10I_2$	$Q_l^0 = I_2$
Actual Adversarial input	$v_e = 0.01\text{sin}(x_e)$	$v_l = 0.01\text{sin}(x_l)$
Control Input Weight	$R_e = 1$	$R_l = 2$
Adversarial Input Weight	$S_e = 25$	$S_l = 64$
Initial Condition	$x_e(0) = [3, \ -1]^T$	$x_l(0) = [3, \ -1]^T$

where $K_e \triangleq R_e^{-1}B^TP_e$ and P_e is the solution of Riccati equation

$$A^TP_e + P_eA + S_e - P_eBR_e^{-1}B^TP_e + \frac{1}{\gamma_e^2}P_eDD^TP_e = 0.$$

Note that the trajectories of expert optimal control input are given for the learner instead of the policy.

For the learner, consider the performance cost function as the quadratic form $V_l = x_l^TP_l^jx_l$ and the state penalty as $Q_l(x_l) = x_l^TQ_l^jx_l$ with the corresponding state-penalty weight Q_l^j. The optimal control input is $u_l^j = -K_l^jx_l$ with feedback gain K_l^j. One selects the activation function for the performance cost function as $\varphi(x_l) = [x_{l1}^2 \ x_{l2}^2 \ 2x_{l1}x_{l2}]^T$ in Algorithm 4.2.

Now, given the parameters in Table 4.1, one shows simulation results using Algorithm 4.2. The small reinforcement interval is $T = 0.005$. Figure 4.2 shows that the learner learns and mimics the expert's behavior trajectories (x_e, u_e^*) within a short time. In Figure 4.2, x_{ln} and x_{en} denotes the n-th state of the leaner and the expert, respectively. Figure 4.3 shows the learning of the state penalty weight Q_l^j and the feedback gain K_l^j. Observe that Q_l^j converges to Q_l^∞ which is equivalent to Q_e, and K_l^j converges to K_e. Eventually, Q_l^j, K_l^j and C_l^j respectively converge to the values

$$Q_l^\infty = \begin{bmatrix} 3.4739 & -2.1831 \\ -2.1831 & 9.9974 \end{bmatrix}, \quad K_l^\infty = \begin{bmatrix} -0.2216 & 2.0582 \end{bmatrix},$$
$$C_l^\infty = \begin{bmatrix} 2.5132 & 2.0580 & -0.2216 \end{bmatrix}^T.$$

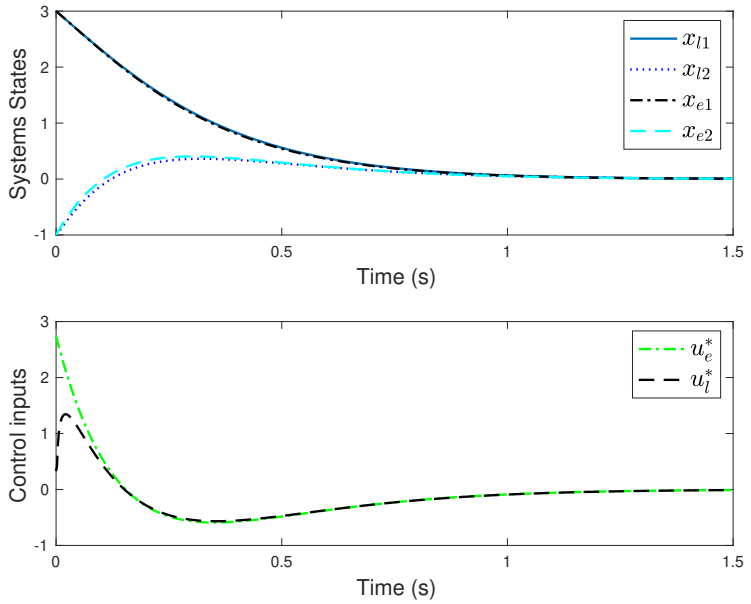


Figure 4.2: Trajectories of the linear learner using Algorithm 4.2

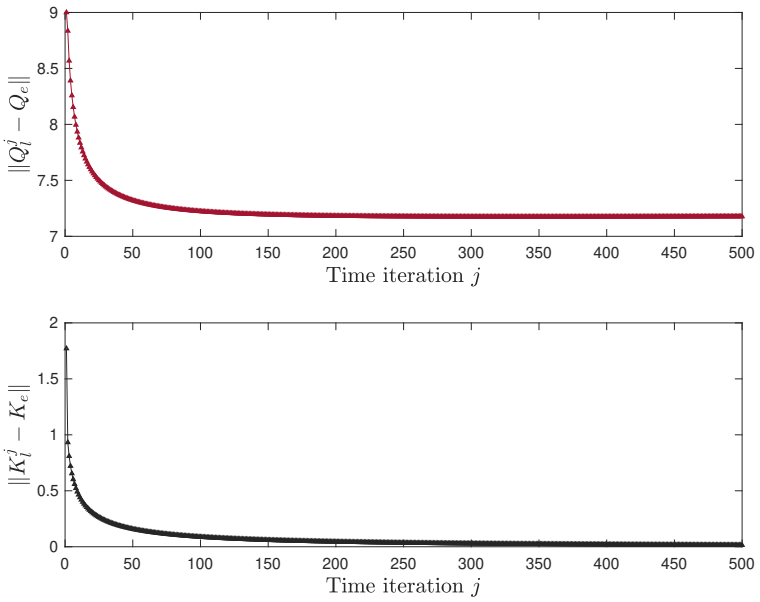


Figure 4.3: Convergence of the linear learner's state-penalty weight Q_l and feedback gain K_l using Algorithm 4.2.

To implement Algorithm 4.4, select the activation functions for V_l, u_l and v_l as $\varphi(x_l) = [x_{l1}^2 \; x_{l2}^2 \; 2x_{l1}x_{l2}]^T$, $\phi(x_l) = [-x_{l1} \; -x_{l2} \; -x_{l1}x_{l2}]^T$ and $\rho(x_l) = [x_{l1} \; x_{l2} \; x_{l1}x_{l2}]^T$, respectively.

Now, considering the same parameters in Table 4.1, one shows the simulation results using Algorithm 4.4. Figure 4.4 shows that the learner learns and performs the expert's behavior trajectories (x_e, u_e^*) within a short period. Figure 4.5 shows the convergence of Q_l^j and K_l^j. Note that Q_l^j converges to a weight which is equivalent to Q_e according to Theorem 4.1, and K_l^j converges to K_e. Finally, Q_l^j, K_l^j, C_l^j, W_l^j and H_l^j respectively converge to the values

$$
\begin{aligned}
Q_l^\infty &= \begin{bmatrix} 3.4539 & -2.2134 \\ -2.2134 & 9.9626 \end{bmatrix}, \; K_l^\infty = \begin{bmatrix} -0.2744 & 2.0723 \end{bmatrix}, \\
C_l^\infty &= \begin{bmatrix} 2.5355 & 2.0635 & -0.2258 \end{bmatrix}^T, W_l^\infty = \begin{bmatrix} -0.2258 & 2.0635 & 0.0004 \end{bmatrix}^T, \\
H_l^\infty &= \begin{bmatrix} 0.0209 & -0.0021 & 0.0005 \end{bmatrix}^T.
\end{aligned}
$$

It should be noted that in Figures 4.3 and 4.5, Q_l^j does not converge to Q_e, while K_l^j converges to K_e. The reason is that Q_l^j converges to an equivalent weight Q_l^∞ to Q_e (see Definition 4.1). This equivalent weight defines the learner the same optimal behavior as the expert. The existence and the convergence of the equivalent weight are shown in Theorems 4.2 and 4.1. Furthermore, in the linear example, when the learner has the convergence $(x, u_l^\infty) = (x, u_e^*)$, one has $-K_l^\infty x_e = -K_e x_e$. This implies $K_l^\infty = K_e$.

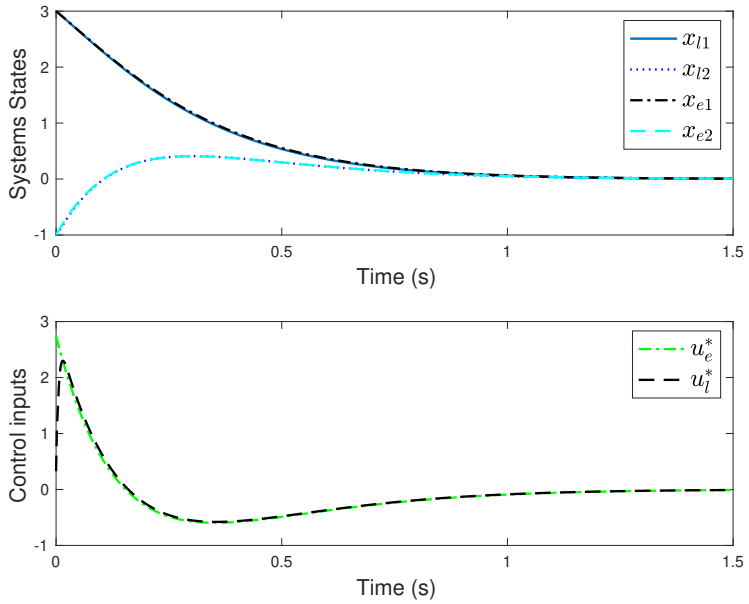


Figure 4.4: Trajectories of the linear learner using Algorithm 4.4.

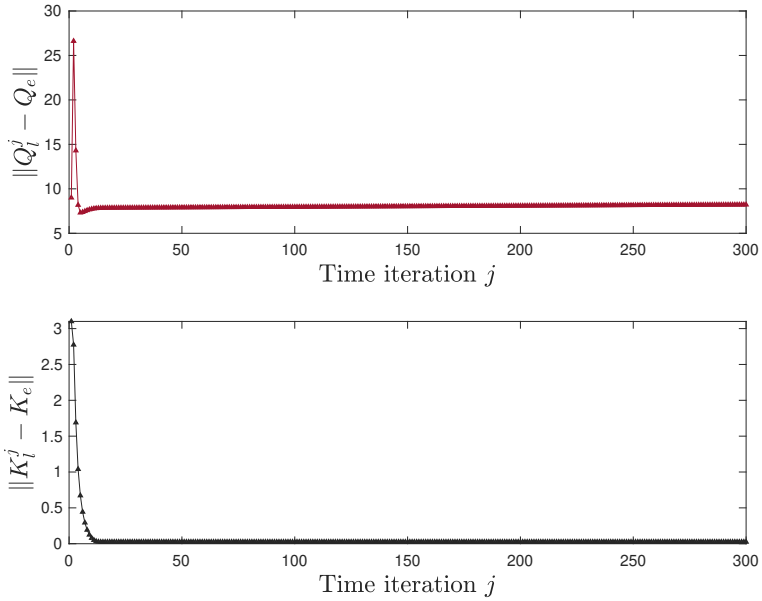


Figure 4.5: Convergence of the linear learner's state-penalty weight Q_l and feedback gain K_l using Algorithm 4.4.

4.5.2 Example 2: Nonlinear Systems

Consider that the draft function $f(\cdot)$, control input function $g(\cdot)$ and adversarial input function

$h(\cdot)$ for two agents are given by

$$f(s) = \begin{bmatrix} -s_1 + x_2 \\ -s_1^3 \end{bmatrix}, \ g(s) = \begin{bmatrix} 0 \\ s_2 \end{bmatrix}, \ h(x) = \begin{bmatrix} 0 \\ s_1 \end{bmatrix}.$$

where s represents x_l or x_e, s_n denotes the n-th element of s.

Suppose in this nonlinear example, the learner has the same actual adversarial input weights as that in Example 1, and so does the expert. The initial states are $x_l(0) = x_e(0) = [3, \ -3]^T$.

Consider both the learner and expert with the state penalty in the form of

$$Q(x) = \begin{bmatrix} x_1^2 & x_2^2 \end{bmatrix} Q \begin{bmatrix} x_1^2 & x_2^2 \end{bmatrix}^T,$$

where Q represents Q_l or Q_e. Then, select state-penalty weights Q_l^0 and Q_e as

$$Q_l^0 = \begin{bmatrix} 1 & -0.0035 \\ -0.0035 & 0.25 \end{bmatrix}, \ Q_e = \begin{bmatrix} 2 & -0.0078 \\ -0.0078 & 1 \end{bmatrix}$$

and control input weights are $R_l = R_e = 1$. Based on the converse Hamilton-Jacobi-Bellman approach in [78], the learner's optimal value function is $V_l^* = \frac{1}{4}x_{l1}^4 + \frac{1}{2}x_{l2}^2$ and the expert's optimal value function is $V_e^* = \frac{1}{2}x_{e1}^4 + x_{e2}^2$. Select the activation function for the learner as $\varphi(x_l) = [x_{l2}^2 \ x_{l1}^4 \ x_{l2}^4]^T$.

Given the above parameters, this work then shows simulation results using the model-based inverse RL Algorithm 4.2 for nonlinear system example. Set $T = 0.004$. Figure 4.6 shows that the learner mimics the expert's behavior (x_e, u_e^*). Figure 4.7 shows the convergence of the state penalty weight Q_l^j. It is observed that Q_l^j converges to Q_l^∞ which is equivalent to Q_e. They are

$$Q_l^\infty = \begin{bmatrix} 1.7625 & -0.5819 \\ -0.5819 & 1.0333 \end{bmatrix}, C_l^\infty = \begin{bmatrix} 0.8800 & 0.4154 & 0.0006 \end{bmatrix}^T.$$

In order to use the model-free inverse RL Algorithm 4.4, this work selects the activation func-

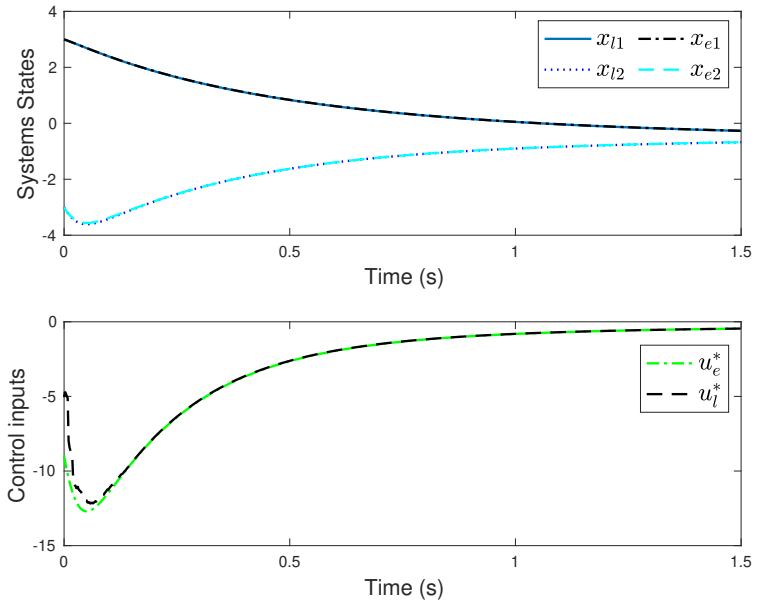


Figure 4.6: Trajectories of states and control inputs of the nonlinear learner using Algorithm 4.2

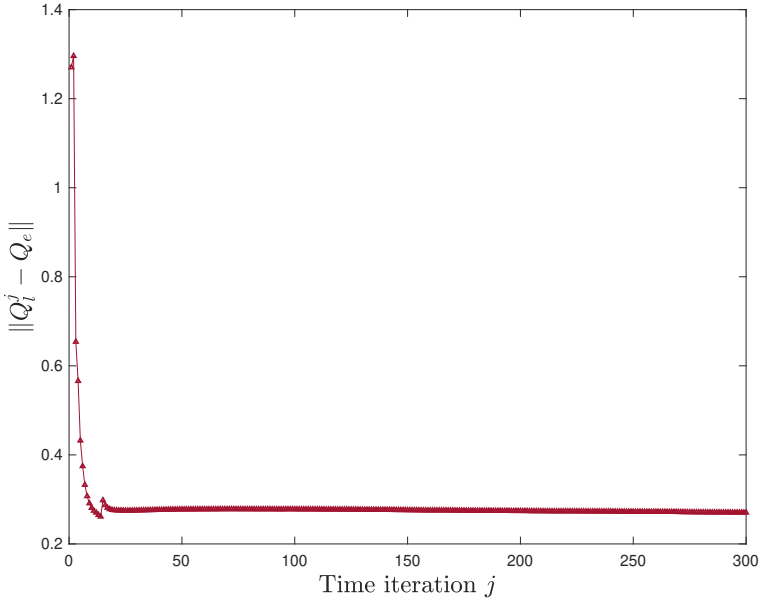


Figure 4.7: Convergence of the nonlinear learner's state-penalty weight Q_l using Algorithm 4.2.

tions for V_l, u_l and v_l as $\varphi(x_l) = [x_{l1}^2\ x_{l2}^2\ x_{l1}^4\ x_{l2}^4]^T$, $\phi(x_l) = [x_{l1}x_{l2}\ x_{l2}^2\ x_{l1}x_{l2}^3\ x_{l1}^3x_{l2}\ x_{l1}^4\ x_{l2}^4]^T$ and $\rho(x_l) = [x_{l1}x_{l2}\ x_{l2}^2\ x_{l1}x_{l2}^3\ x_{l1}^3x_{l2}\ x_{l1}^4\ x_{l2}^4]^T$, respectively.

Given the above parameters, this work now shows results using Algorithm 4.4. Figure 4.8 shows that the learner obtains a convergence behavior $(x_l, u_l^*) = (x_e, u_e^*)$. Figure 4.9 shows the convergence of state-penalty weight Q_l^j. One obtains the converged C_l^∞, W_l^∞, H_l^∞ and Q_l^∞ as

$$Q_l^\infty = \begin{bmatrix} 1.7812 & -0.5753 \\ -0.5753 & 1.0304 \end{bmatrix}, C_l^\infty = \begin{bmatrix} 0.0008 & 0.8110 & 0.4157 & 0.0005 \end{bmatrix}^T,$$
$$W_l^\infty = \begin{bmatrix} 0.0002 & -0.8120 & 0.0001 & 0 & 0.0003 & -0.0011 \end{bmatrix}^T,$$
$$H_l^\infty = \begin{bmatrix} 0.8155 & 0.0001 & 0 & 0.0012 & 0.0001 & 0 \end{bmatrix}^T.$$

Note that Q_l^j in Figures 4.7 and 4.9 does not converge to Q_e because Q_l^j converges to an equivalent weight Q_l^∞ that is not equal to Q_e as shown in Definition 4.1 and Theorem 4.2. This equivalent weight can define the learner the same optimal behavior as the expert. The existence and the convergence of the equivalent weight are shown in Theorems 4.2 and 4.1.

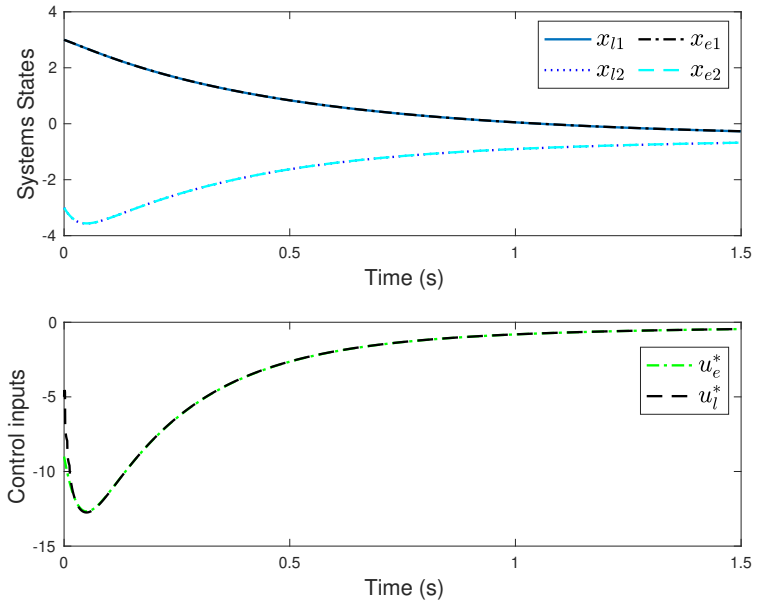


Figure 4.8: Trajectories of states and control inputs of the nonlinear learner using Algorithm 4.4

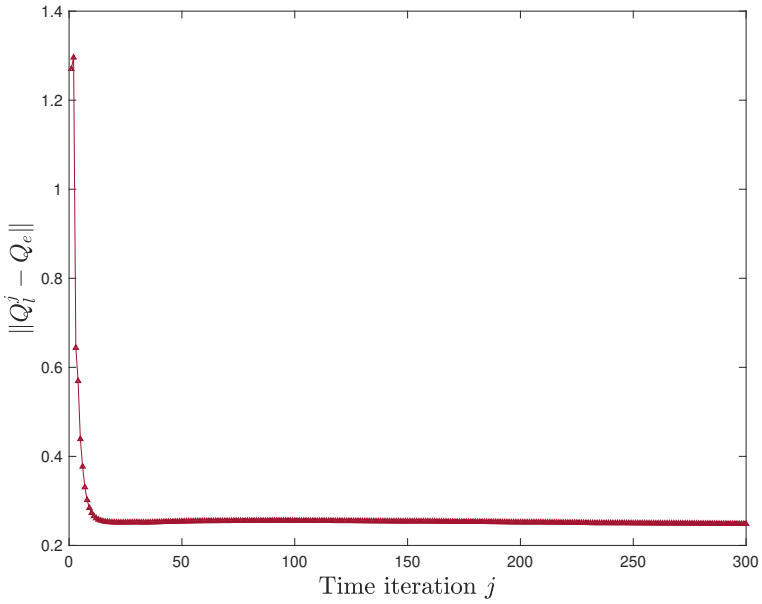


Figure 4.9: Convergence of the nonlinear learner's state-penalty weight Q_l using Algorithm 4.4

4.5.3 Comparison to Existing Optimal Tracking by RL Technique in [73]

This work now compares the proposed inverse RL imitation learning methods with the off-policy optimal tracking method in [73]. [73] proposes off-policy tracking control algorithm given manually specified cost function, whereas this work reconstructs unknown cost function from demonstrations for the imitation. The tracking performance by [73] is then compared with the imitation performance of the examples in Section 4.5.1 and Section 4.5.2.

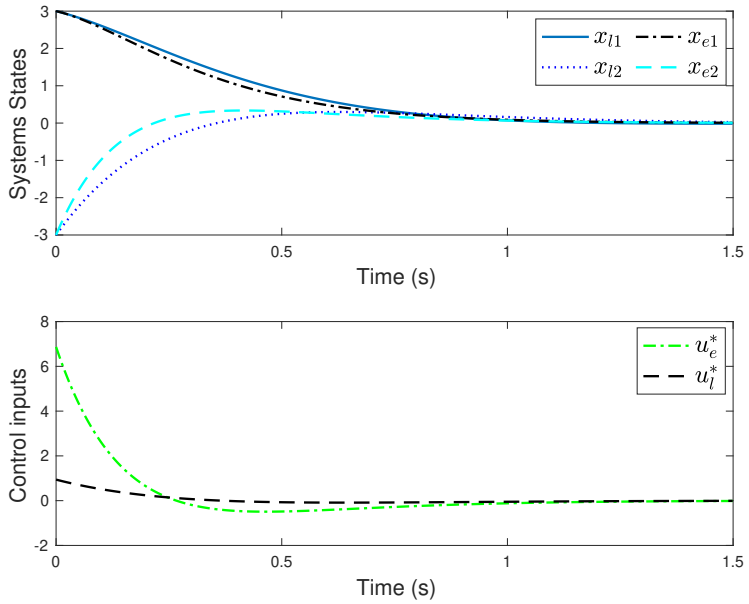


Figure 4.10: Trajectories of the linear learner using off-policy algorithm [73]

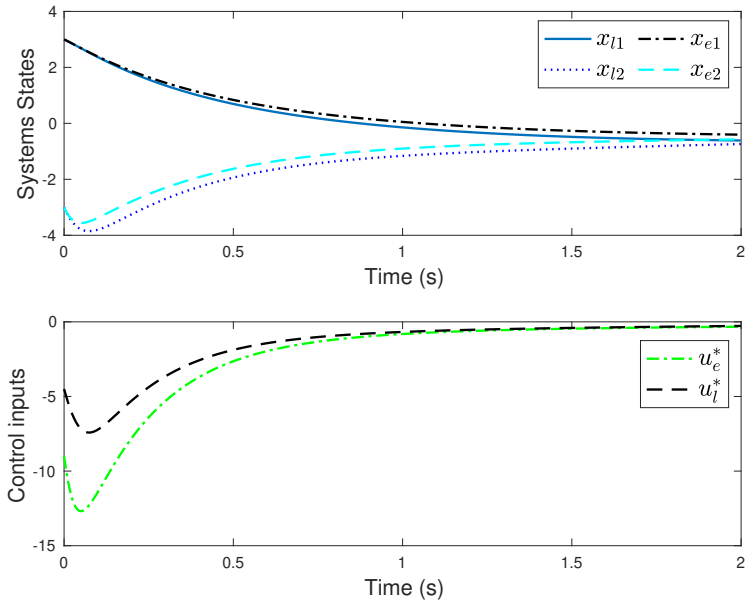


Figure 4.11: Trajectories of the nonlinear learner using off-policy algorithm [73]

The off-policy tracking algorithm in [73] is implemented in both two examples that are given in Section 4.5.1 and Section 4.5.2. Assume that the state-penalty weights of the learner in two examples for [73] are manually selected as initial ones, i.e., $Q_l^0 = I_2$ for the linear example in

Section 4.5.1 and $Q_l^0 = \begin{bmatrix} 1 & -0.0035 \\ -0.0035 & 0.25 \end{bmatrix}$ for the nonlinear example in Section 4.5.2. The other parameters are the same as those in two examples.

Figure 4.10 and Figure 4.11 show the trajectories of the linear and nonlinear learners, respectively, by using the off-policy algorithm in [73]. Comparing Figure 4.4 and Figure 4.10, and Figure 4.8 and Figure 4.11, one notices that the proposed inverse RL methods have better imitation performance. This shows that it is hard to manually specify a cost function for the learner such that it has the same behavior trajectories with the expert all the time. However, the inverse RL methods of reconstructing cost functions from demonstrations can obtain the same policy as the expert, thus leading to better imitation performance.

4.6 Conclusion

This chapter proposes novel inverse RL control methods to solve the Adversarial Apprentice Game by learning the objective weights of the expert, such that the learner performed the expert's behavior. Both model-based and model-free inverse RL algorithms are developed to solve the game, and then implemented by using neural networks. Given the expert's demonstrations, the learner learns the expert's unknown performance weights which are not unique. The convergence and stability of the algorithms are guaranteed.

Chapter 5: INVERSE RL FOR MULTI-PLAYER NONCOOPERATIVE APPRENTICE GAMES

5.1 Introduction

This chapter generalizes the inverse RL algorithms to multi-player games for imitation learning, which we call Multi-player Noncooperative Apprentice Games. In these games, both the expert and the learner have N-player control inputs. Inverse RL algorithms are proposed to solve the games for nonlinear continuous-time systems described by differential dynamic equations. The games are solved by reconstructing the unknown performance reward function of each player in the expert system by each player in the learner system. The learner observes the demonstrated expert players' behaviors, such that the learner mimics the expert's behaviors, i.e., states and control inputs of each player.

First, this chapter develops a model-based inverse RL algorithm that involves two learning stages: an optimal control learning stage and a second stage based on inverse optimal control. Second, a completely model-free off-policy integral inverse RL algorithm is developed for the expert and the learner with homogeneous control inputs. Third, another partially model-free integral inverse RL algorithm is proposed for heterogeneous control inputs. These two integral inverse RL algorithms are further implemented via neural networks (NNs). Finally, simulations verify the effectiveness of the proposed algorithms.

The rest of the chapter is organized as follows. Section 5.2 introduces multi-player non-zero-sum game expert and learner systems, and formalizes inverse RL problems. Section 5.3 develops model-based inverse RL algorithms for multiplayer noncooperative apprentice games. Section 5.4 proposes a completely model-free inverse RL algorithm for the case of homogeneous control inputs. Section 5.5 proposes a partially model-free inverse RL algorithm for the case of heterogeneous control inputs. In Section 5.6, simulation examples are studied to verify the proposed methods. Section 5.7 concludes the chapter.

5.2 Problem Formulation

This section introduces Multi-player Noncooperative Apprentice Games, multi-player non-zero-sum game expert system and N-player learner system.

5.2.1 Multi-player Noncooperative Apprentice Games

This chapter considers a noncooperative game as the setting, where both the learner and the expert systems consist of multiple players. One calls the noncooperative games based on apprentice learning with multiple players **Multi-player Noncooperative Apprentice Games**, in which the learner aims to mimic the expert's trajectories but does not know the performance reward functions of expert players. First, the expert computes desired trajectories, i.e., optimal states and control inputs of all players with desired performance weights. Then, given the expert's trajectories, the learner reconstructs the reward functions of expert players to mimic expert's trajectories. In the autonomous driving case, an expert driver solves the noncooperative game on states and players (acceleration, braking, turning, etc) based on rewards. The expert driver shows optimal trajectories to the learner vehicle which also has the noncooperative game to be solved and then mimics the trajectories by learning the expert game rewards based on IOC.

Specifically, a Multi-player Noncooperative Apprentice Game is defined as a tuple $(N, \mathcal{S}, \mathcal{A}, Q)$, where N, $\mathcal{S}$ and $\mathcal{A}$ are player, state and action sets, respectively. $Q : \mathcal{S} \to \mathbb{R}$ denotes a quadratic function set of system states.

5.2.2 Multi-player non-zero-sum game expert system

Consider the N-player non-zero-sum expert system as

$$\dot{x}_e = f(x_e) + \sum_{i=1}^{N} g_i(x_e) u_{ie}, \tag{5.1}$$

where $x_e \in \mathbb{R}^n$, $u_{ie} \in \mathbb{R}^{m_i}$, $f(x_e)$ and $g_i(x_e)$ denote expert's state, control inputs of player i, $i \in N$, state dynamics and control input dynamics, respectively. Assume that $f + \sum_{i=1}^{N} g_i u_{ie}$ is Lipschitz

continuous on a set $\Pi \in \mathbb{R}^n$ containing the origin such that there exist continuous controls on Π that stabilize (5.1).

Definition 5.1. *If $g_i \neq g_j$ for some $i \neq j$ in* (5.1)*, where $i, j \in N$, then the control inputs in* (5.1) *are said to be heterogeneous. If $g_i = g$, $\forall i \in N$, then the control inputs in* (5.1) *are said to be homogeneous.*

The performance reward function of each expert player i is defined as

$$\begin{aligned} &V_{ie}(x_e(t_0), u_{1e}, \cdots, u_{Ne}) \\ &= \int_{t_0}^{\infty} \Big(q^T(x_e) Q_{ie} q(x_e) + \sum_{j=1}^{N} u_{je}^T R_{je} u_{je} \Big) d\tau, \end{aligned} \tag{5.2}$$

where $q(x_e) = [x_{e_1}^s \cdots x_{e_N}^s]^T \in \mathbb{R}^n$ is the state function with the power of s; $Q_{ie} = Q_{ie}^T \in \mathbb{R}^{n \times n} > 0$ and $R_{je} = R_{je}^T \in \mathbb{R}^{m \times m} > 0$ for $i, j \in N$ are performance weights.

Definition 5.2. *Control policy profiles (u_{ie}^*, u_{-ie}^*) are said to form a Nash equilibrium set in the N-player noncooperative game* (5.2) *if $V_{ie}^* = V_{ie}(u_{ie}^*, u_{-ie}^*) \leq V_{ie}(u_{ie}, u_{-ie}^*)$ holds for all, where $-i \triangleq \{1, \cdots, i-1, i+1, \cdots N\}$.*

Given performance reward functions (5.2) and based on optimal control for multiplayer games [10, 54], (5.1) has optimal controls u_{ie}^* as

$$u_{ie}^* = -\frac{1}{2} R_{ie}^{-1} g_i^T(x_e) \nabla V_{ie}(x_e), \ i \in N, \tag{5.3}$$

associated with expert's N-coupled HJ equations

$$\begin{aligned} 0 = q^T(x_e) Q_{ie} q(x_e) + \frac{1}{4} \sum_{j=1}^{N} \nabla V_{je}^T(x_e) g_j(x_e) R_{je}^{-1} g_j^T(x_e) \nabla V_{je}(x_e) \\ + \nabla V_{ie}^T(x_e) \Big(f(x_e) - \frac{1}{2} \sum_{j=1}^{N} g_j(x_e) R_{je}^{-1} g_j^T(x_e) \nabla V_{je}(x_e) \Big), \ i \in N. \end{aligned} \tag{5.4}$$

5.2.3 N-player game learner system

Consider the N-player learner system that has the same dynamic functions as the expert system (5.1)

$$\dot{x} = f(x) + \sum_{i=1}^{N} g_i(x)u_i, \tag{5.5}$$

where the learner has the state $x \in \mathbb{R}^n$ and player control inputs $u_i \in \mathbb{R}^{m_i}$, $i \in N$. The performance reward function of player i is defined as

$$V_i(x(t_0), u_i, u_{-i}) = \int_{t_0}^{\infty} \Big(q^T(x) Q_i q(x) + \sum_{j=1}^{N} u_j^T R_j u_j \Big) d\tau \tag{5.6}$$

with the same state function $q(\cdot) \in \mathbb{R}^n$ as that in (5.2), and performance weights $Q_i = Q_i^T \in \mathbb{R}^{n\times n} > 0$ and $R_j = \text{diag}\{r_{j_1}, r_{j_2}, \cdots, r_{j_m}\} \in \mathbb{R}^{m\times m} > 0$, for all $i, j \in N$.

The next assumptions and inverse RL problems are given.

Assumption 5.1. *The learner* (5.5) *knows* $q(\cdot)$, Q_i, $\forall i \in N$, *and* $R_1, \cdots, R_N$ *in* (5.6). *However, it does not know the expert's performance weights, i.e.,* Q_{ie}, $R_{1e}, \cdots, R_{Ne}$ *in* (5.2). *Moreover,* R_j *can be different from* R_{je}.

Assumption 5.2. *Each learner player* i *can observe the expert's control input* u_{ie}^*, *where* $i \in N$. *Both systems have the same initials, i.e.,* $x(t_0) = x_e(t_0)$.

Multi-player Noncooperative Apprentice Game Problem. Given Assumptions 5.1-5.2 and R_j for all $j \in N$, the learner should infer the unknown performance weight Q_{ie} for (5.6) for all $i \in N$ to mimic expert's trajectories, i.e., (x, u_1^*, u_2^*, $\cdots$, u_N^*) = (x_e, u_{1e}^*, u_{2e}^*, $\cdots$, u_{Ne}^*).

5.3 Model-based Inverse RL for Multi-player Noncooperative Apprentice Games

To solve the above problem, this section develops a model-based inverse RL algorithm that

combines a first optimal control learning stage and a second IOC-based learning stage. The algorithm infers the unknown expert player i's performance weight Q_{ie} in (5.2) for learner player i's reward function (5.6) by observing the expert player i's control input u_{ie}^*. Then, the learner system aims to obtain a convergence $(x, u_1^*, u_2^*, \cdots, u_N^*) = (x_e, u_{1e}^*, u_{2e}^*, \cdots, u_{Ne}^*)$.

5.3.1 Optimal Control Learning of Learner System

The first stage of inverse RL Algorithm 5.1 is based on optimal control. Similar to (5.1)-(5.4), the learner has optimal controls

$$u_i^* = -\frac{1}{2} R_i^{-1} g_i^T(x) \nabla V_i(x),\ i \in N \tag{5.7}$$

and N-coupled HJ equations

$$\begin{aligned} 0 = {} & q^T(x) Q_i q(x) + \frac{1}{4} \sum_{j=1}^{N} \nabla V_j^T(x) g_j(x) R_j^{-1} g_j^T(x) \nabla V_j(x) \\ & + \nabla V_i^T(x) \Big(f(x) - \frac{1}{2} \sum_{j=1}^{N} g_j(x) R_j^{-1} g_j^T(x) \nabla V_j(x) \Big),\ i \in N. \end{aligned} \tag{5.8}$$

For player i, given a current estimate Q_i of Q_{ie}, the learner uses optimal control learning to solve (5.7)-(5.8) and obtains the converged optimal solution (u_i^*, V_i^*). The iteration form of this stage is to be seen as (5.10) and (5.11) in Algorithm 5.1.

5.3.2 IOC Learning for Performance Weight Q_i

The second learning stage of inverse RL Algorithm 5.1 is based on IOC [31]. The learner improves the estimate Q_i of Q_{ie} for the player i using expert player i's u_{ie}^* in (5.3). Repeating these two stages for all players, the learner eventually obtains $(x, u_1^* \cdots, u_N^*) \to (x_e, u_{1e}^*, \cdots, u_{Ne}^*)$.

Given u_{ie}^* and (u_i^*, V_i^*) from the first learning stage, the learner revises player i's performance

weight Q_i by

$$q(x)^T Q_i q(x) = u_{ie}^{*T} R_i u_{ie}^* - 2u_{ie}^{*T} R_i u_i^* - \sum_{j\in -i} u_j^{*T} R_j u_j^* - \nabla V_i^{*T}(x)\Big(f(x) + \sum_{j=1}^{N} g_j(x) u_j^*\Big). \quad (5.9)$$

The iteration form of (5.9) using IOC learning is given by (5.12) to be presented in Algorithm 5.1. Now give a model-based inverse RL Algorithm 5.1 based on (5.7)-(5.9).

Algorithm 5.1 Model-based Inverse RL Algorithm

1: Select $Q_i^0 > 0$, stabilizing u_i^{00}, and small thresholds ε_i and e_i, $\forall i \in N$. Set $k = 0$;

2: **Outer k iteration loop based on IOC**

3: **Inner l iteration loop using optimal control** given k, set $l = 0$;

4: Solve the N-tuple of costs using

$$0 = q^T(x) Q_i^k q(x) + \sum_{j=1}^{N} (u_j^{kl})^T R_j u_j^{kl} + \left(\nabla V_i^{kl}(x)\right)^T \Big(f(x) + \sum_{j=1}^{N} g_j(x) u_j^{kl}\Big), i \in N; \quad (5.10)$$

5: Update the N-tuple of control policies using

$$u_i^{k(l+1)} = -\frac{1}{2} R_i^{-1} g_i^T(x) \nabla V_i^{kl}(x),\ i \in N; \quad (5.11)$$

6: Stop if $\|V_i^{kl} - V_i^{k(l-1)}\| \le \varepsilon_i$, then set $V_i^k = V_i^{kl}$ and $u_i^k = u_i^{kl}$, otherwise set $l \leftarrow l + 1$ and go to Step 4;

7: **Outer k iteration loop based on IOC:** update the performance weight Q_i^{k+1} using u_{ie}^* (5.3) by

$$\begin{aligned} &q^T(x) Q_i^{k+1} q(x) \quad (5.12)\\ &= u_{ie}^{*T} R_i u_{ie}^* - 2u_{ie}^{*T} R_i u_i^k - \sum_{j\in -i} (u_j^k)^T R_j u_j^k - \left(\nabla V_i^k(x)\right)^T \Big(f(x) + \sum_{j=1}^{N} g_j(x) u_j^k\Big),\ i \in N; \end{aligned}$$

8: Stop if $\|u_i^k - u_{ie}^*\| \le e_i$, otherwise set $u_i^{(k+1)0} = u_i^k$, $k \leftarrow k + 1$, and go to Step 3.

Remark 5.1. *The importance of Algorithm 5.1 are mainly three aspects. Firstly, compared with multiplayer inverse RL for MDPs [75, 77, 123], this work studies multiplayer apprentice games described by differential dynamic equations. Secondly, based on Algorithm 5.1, this work enables to derive model-free versions to reconstruct reward functions compared with IOC works [39, 74] that use system dynamics. Thirdly, Algorithm 5.1 applies to both systems that have either homogeneous or heterogeneous control inputs.*

5.3.3 Convergence and Stability Analysis

This subsection shows 1) the convergence of Algorithm 5.1, 2) the non-uniqueness of performance weights, and 3) the stability of learner dynamics.

Lemma 5.1. *[113] Given $Q_i^k > 0$ and admissible u_i^{k0}, the nonlinear Lyapunov equations* (5.10) *have solutions $V_i^k \geq 0$ for all $i \in N$ and $k = 0, 1, \cdots$.*

Definition 5.3. ***(Equivalent weight).*** *The matrices Q_{ie}, $R_{1e}, \cdots, R_{Ne}$ in* (5.4) *define x_e and u_{ie}^* in* (5.3)*. One may find a different $\tilde{Q}_i$ given $R_1, \cdots, R_N$ in* (5.8)*, such that $x = x_e$ and $u_i^* = u_{ie}^*$, with u_i^* defined in* (5.7)*. Then, $\tilde{Q}_i$ is called to be equivalent to Q_{ie}, where $i \in N$.*

Definition 5.4. *The control input u_i is said to be an uniformly approximate solution of u_{ie}^* if there exists a small threshold e_i such that $\|u_i - u_{ie}^*\| \leq e_i$ holds for each $i \in N$, where u_{ie}^* is given in* (5.2)*.*

Assumption 5.3. *Suppose there exists at least one equivalent weight to Q_{ie} in Definition 5.3 given functions $q(\cdot)$, $f(\cdot)$ and $g(\cdot)$ with the chosen R_j, $j \in N$ in Algorithm 5.1.*

Theorem 5.1. *Let Assumptions 5.1-5.3 hold. Use Algorithm 5.1 to solve the multi-player apprentice problem. Suppose the equivalent weight in Definition 5.3 is nonunique. Then, the algorithm will terminate at a limited iteration step. The learner obtains uniformly approximate solutions of $\tilde{Q}_i$ and $(x_e, u_{1e}^*, \cdots, u_{Ne}^*)$, where $\tilde{Q}_i$ is equivalent to Q_{ie}, $\forall i \in N$.*

Proof. One will prove the convergence in both inner iteration loops and outer iteration loops.

First, one shows the convergence of inner iteration loops at any outer k iteration loop. As shown in [105] and given Lemma 5.1, at outer k iteration loop, if given initial stabilizing control inputs u_i^{k0} and $Q_i^k > 0$, $R_j > 0$, $j \in N$, inner iteration steps 4 and 5 give the convergent solutions u_i^k and V_i^k as $l \to \infty$, where V_i^k is the optimal value. Note that u_i^{k0} is obtained from Step 8. The positive definiteness of Q_i^k is proved as below.

Second, one proves the convergence of outer iteration loops by finding the limited maximum iteration steps, which is similar to that in [3].

Given the current Q_i^k, inner iteration loop gives

$$0 = q^T(x)Q_i^k q(x) + \sum_{j=1}^{N}(u_j^k)^T R_j u_j^k + \left(\nabla V_i^k(x)\right)^T \left(f(x) + \sum_{j=1}^{N} g(x)u_j^k\right),\ i \in N. \tag{5.13}$$

Adding (5.13) to the right side of (5.12) yields

$$\begin{aligned} q^T(x)Q_i^{k+1}q(x) &= u_{ie}^{*T}R_i u_{ie}^* - 2u_{ie}^{*T}R_i u_i^k + (u_i^k)^T R_i u_i^k + q^T(x)Q_i^k q(x) \\ &= (u_{ie}^* - u_i^k)^T R_i (u_{ie}^* - u_i^k) + q^T(x)Q_i^k q(x) \end{aligned} \tag{5.14}$$

which implies that given initial $Q_i^0 > 0$, one has

$$0 < Q_i^0 \le \cdots \le Q_i^k \le Q_i^{k+1}, \quad \forall i, k. \tag{5.15}$$

Define $\Delta_i^k(x) \triangleq (u_{ie}^* - u_i^k)^T R_i (u_{ie}^* - u_i^k)$. Based on (5.14), one has

$$q^T(x)Q_i^k q(x) = \Delta_i^{k-1}(x) + \Delta_i^{k-2}(x) + \cdots + \Delta_i^0(x) + q^T(x)Q_i^0 q(x). \tag{5.16}$$

Suppose there are infinitely many equivalent weights $\tilde{Q}_i$. This will be proved in Theorem 5.2. There would exist at least one $\tilde{Q}_i$ such that $\tilde{Q}_i \ge Q_i^0$ holds and Q_i^k can increase to the neighbor of it. That is, there exists a small threshold α, such that $\|Q_i^k - \tilde{Q}_i\| \le \alpha$ holds. Then, the inner loops have the uniformly approximate solution (see Definition 5.4) of u_{ie}^*, i.e., $\|u_i^k - u_{ie}^*\| \le e_i$. Here, the

small threshold e_i is used to stop the computation at k without affecting the positive definiteness of Q_i^k. Thus, Algorithm 5.1 is stopped at k and one has

$$\begin{aligned}\|q^T(x)\tilde{Q}_i q(x)\| &= \|\Delta_i^k(x) + \Delta_i^{k-1}(x) + \cdots + \Delta_i^0(x) + q^T(x)Q_i^0 q(x)\| \\ &\geq (k+1)e_i^2\lambda_{min}(R_i) + \|q^2\|\|Q_i^0\| \end{aligned} \tag{5.17}$$

which implies

$$k \leq \frac{\|q\|^2\|\tilde{Q}_i\| - \|q\|^2\|Q_i^0\|}{e_i^2\lambda_{min}(R_i)} \equiv k_0. \tag{5.18}$$

This shows that Algorithm 5.1 will terminate after at most outer k_0 loops. Then, the learner obtains uniformly approximate solutions of $\tilde{Q}_i$ and $(x_e, u_{1e}^*, \cdots, u_{Ne}^*)$, where $\tilde{Q}_i$ is the equivalent weight to Q_{ie}.

Given the same initial states, i.e. $x(t_0) = x_e(t_0)$, one uses the uniformly approximate control policy $u_{ie}^*(t_0)$ from t_0. Then, the uniformly approximate solutions of $(x_e, u_{1e}^*, \cdots, u_{Ne}^*)$ are obtained for all time.

However, one cannot guarantee that $\tilde{Q}_i = Q_{ie}$ holds for any $i \in N$ because $\tilde{Q}_i$ is only guaranteed to be the equivalent weight to Q_{ie} as shown in Definition 5.3. □

Remark 5.2. *[40] shows that different performance functions can define the same optimal behaviors. This allows us to give Assumption 5.3 for nonlinear systems. Given Assumptions 5.1-5.3 and an initial $Q_i^0 > 0$, Theorem 5.1 ensures the existence and the convergence of solutions of* (5.10) *and $\tilde{Q}_i$.*

The next result shows the assumed nonuniqueness of $\tilde{Q}_i$ in Definition 5.3 and Theorem 5.1.

Theorem 5.2. *Suppose the learner obtains $\tilde{Q}_i$ and $\nabla\tilde{V}_i(x_e)$ with nonzero u_{ie}^*. Then, they may be nonunique. Besides, $\tilde{Q}_i$ can be different from Q_{ie}. Then, $\nabla\tilde{V}_i(x_e)$ satisfies*

$$g_i^T(x_e)\nabla\tilde{V}_i(x_e) = R_i R_{ie}^{-1} g_i^T(x_e)\nabla V_{ie}^*(x_e),\ i \in N \tag{5.19}$$

and $\tilde{Q}_i$ satisfies

$$
\begin{aligned}
&q^T(x_e)(Q_{ie} - \tilde{Q}_i)q(x_e) \qquad (5.20)\\
&= \frac{1}{4}\sum_{j=1}^{N} \nabla\tilde{V}_j^T(x_e) g_j(x_e) R_j^{-1}(R_j - R_{je})R_j^{-1} g_j^T(x_e)\nabla\tilde{V}_j(x_e)\\
&\quad + \big(\nabla\tilde{V}_i^T(x_e) - \nabla V_{ie}^{*T}(x_e)\big)\Big(f(x_e) - \frac{1}{2}\sum_{j=1}^{N} g_j(x_e)R_{je}^{-1}g_j^T(x_e)\nabla\tilde{V}_j(x_e)\Big),\ i \in N,
\end{aligned}
$$

with $\tilde{Q}_i$ solved with nonzero $q(x_e)$ and $\nabla V_{ie}^*(x_e)$.

Proof. Theorem 5.1 shows that the learner obtains the uniformly approximate solutions of $\tilde{Q}_i$ and $(x_e, u_{1e}^*, \cdots, u_{Ne}^*)$. Simultaneously, let $\tilde{V}_i(x_e)$ be associated with $\tilde{Q}_i$. Then, one has

$$
u_{ie}^* = -\frac{1}{2}R_i^{-1}g_i^T(x_e)\nabla\tilde{V}_i(x_e),\ i \in N. \qquad (5.21)
$$

where $\nabla\tilde{V}_i(x_e)$ satisfies

$$
\begin{aligned}
0 &= q^T(x_e)\tilde{Q}_i q(x_e) + \frac{1}{4}\sum_{j=1}^{N}\nabla\tilde{V}_j^T(x_e)g_j(x_e)R_{je}^{-1}g_j^T(x_e)\nabla\tilde{V}_j(x_e)\\
&\quad + \nabla\tilde{V}_i^T(x_e)\Big(f(x_e) - \frac{1}{2}\sum_{j=1}^{N} g_j(x_e)R_{je}^{-1}g_j^T(x_e)\nabla\tilde{V}_j(x_e)\Big), \qquad (5.22)
\end{aligned}
$$

It follows from Definition 5.3, (5.21) and (5.3) that one has (5.19). Subtracting (5.22) from (5.4) yields (5.20).

It is seen from (5.19) that if one lets $g^T(x_e)X_i = Y_i$, where $Y_i = R_i R_{ie}^{-1} g^T(x_e)\nabla V_{ie}^*(x_e)$ and $x_e \neq 0$, then there will be infinitely many solutions of X_i when rank$(g(x_e)) \neq n$. In fact, one cannot guarantee that rank$(g(x_e)) = n$ holds. Therefore, the value $\nabla\tilde{V}_i(x_e)$ that makes (5.22) hold is not unique. Then, it follows from (5.20) that $\tilde{Q}_i$ is not unique, where $q(x_e)$ is nonzero in (5.20). Otherwise, $\tilde{Q}_i$ is unique if rank$(g(x_e)) = n$. As a result, the equivalent weight $\tilde{Q}_i$ may not be unique but are all bounded. Note that as shown in Theorem 5.1, Algorithm 5.1 does not learn all the equivalent weights but converge and stop as long as one of them is learned. □

Lemma 5.2. *Consider the learner system* (5.5) *with performance reward functions* (5.6)*. Use Algorithm 5.1. Then,* (5.5) *is asymptotically stable with the equilibrium* u_i^{kl} *in* (5.11) *and* u_i^k *in* (5.12) *at each outer* k *and inner* l *loop, where* $k = 0, 1, \cdots$*,* $l = 0, 1, \cdots$ *and* $i \in N$*.*

Proof. The stability of the learner (5.5) are proved at both inner and outer loops of Algorithm 5.1. The learner should learn stabilizing control inputs and weights $Q_i^k > 0$ in inner loops and outer loops, respectively.

Select (5.6) as the learner player i's Lyapunov candidate for (5.5). First, given k, according to (5.10), one has $\dot{V}_i^{kl} = -q^T(x)Q_i^k q(x) - \sum_{j=1}^N (u_j^{kl})^T R_j u_j^{kl}$, which implies that $\dot{V}_i^{kl} \leq 0$, $\forall l = 0, 1, \cdots$. Then, one deduces that $V_i^{kl} \geq 0$. Also, note that $\dot{V}_i^{kl} = V_i^{kl} = 0$ holds only when $x = 0$. Thus, the learner system is asymptotically stable in inner iteration loops.

Second, it follows from Theorem 5.1 that (5.13) holds. Then, for outer k, $k = 0, 1, \cdots$, iteration loop, one has $\dot{V}_i^k = -q^T(x)Q_i^k q(x) - \sum_{j=1}^N (u_j^k)^T R_j u_j^k$. It follows from (5.14) that $Q_i^k > 0$. Then, $\dot{V}_i^k \leq 0$ holds. Also, (5.6) indicates $V_i^k \geq 0$. Thus, the learner system is asymptotically stable in outer iteration loops.

Therefore, the learner system (5.5) is asymptotically stable at both inner and outer loops in Algorithm 5.1. □

Remark 5.3. *This work learns the state-penalty weights and selects the control penalty weights. This is a partial inverse RL problem. Yet Theorems 5.1-5.2 show that it is equivalent to a full inverse RL problem because it reconstructs equivalent weights* $(\tilde{Q}_i, R_j), i, j \in N$*, that generate the same expert behaviors* $(x_e, u_{1e}^*, \cdots, u_{Ne}^*)$ *in apprentice learning problems.*

5.4 Completely Model-free Inverse RL for Homogeneous Control Inputs

Inverse RL Algorithm 5.1 needs the system dynamics f, g_1, $\cdots$, g_N to be known. In contrast, this section develops a completely model-free inverse RL algorithm for the case of homogeneous control inputs, i.e., $g_j = g$, $\forall j \in N$. The algorithm is then implemented via NNs.

5.4.1 Off-policy Inverse RL for Homogeneous Control Inputs

To develop a completely model-free inverse RL algorithm for (5.1) and (5.5) with homogeneous control inputs, integral RL techniques [72] are used in inner l iteration loop of Algorithm 5.1 with $g_j = g$ for all $j \in N$. This finds model-free equations that are equivalent to (5.10)-(5.11) with $g_j = g$. Towards this end, one first rewrites (5.5) with homogeneous control inputs as

$$\dot{x} = f(x) + \sum_{j=1}^{N} g(x)u_j^{kl} + \sum_{j=1}^{N} g(x)(u_j - u_j^{kl}), \tag{5.23}$$

where $u_j^{kl} \in \mathbb{R}^m$ is the update of learner player j at inner l loop and outer k loop given the performance weight Q_i^k in Algorithm 5.1.

One refers to the off-policy integral RL for multi-player systems [105] and obtains the off-policy integral RL Bellman equations for (5.10) and (5.11) as

$$\begin{aligned} &V_i^{kl}(x(t+T)) - V_i^{kl}(x(t)) \\ &= \int_t^{t+T} \Big(- q^T(x)Q_i^k q(x) - \sum_{j=1}^{N} (u_j^{kl})^T R_j u_j^{kl} \Big) d\tau - \int_t^{t+T} \Big(2(u_i^{k(l+1)})^T R_i \sum_{j=1}^{N} (u_j - u_j^{kl}) \Big) d\tau \end{aligned} \tag{5.24}$$

which solves the converged solution set $(V_i^{kl},\, u_i^{kl}) \to (V_i^k, u_i^k)$ for all $i \in N$ given the current estimate Q_i^k.

Next, one uses integral RL in outer k loop of Algorithm 5.1 again to find a model-free equation equivalent to (5.12). Given u_{ie}^* and the converged (V_i^k, u_i^k) from (5.24), one updates Q_i^{k+1} by

$$\begin{aligned} &\int_t^{t+T} q^T(x)Q_i^{k+1}q(x)d\tau \\ &= \int_t^{t+T} \Big(u_{ie}^{*T} R_i u_{ie}^* - 2u_{ie}^{*T} R_i u_i^k - \sum_{j \in -i} (u_j^k)^T R_j u_j^k \Big) d\tau - V_i^k(x(t+T)) + V_i^k(x(t)). \end{aligned} \tag{5.25}$$

The completely model-free off-policy integral inverse RL algorithm is given in Algorithm 5.2.

Algorithm 5.2 Completely Model-free Integral Inverse RL Algorithm for Homogeneous Control Inputs

1: Select $Q_i^0 > 0$, stabilizing u_i^{00}, and small thresholds ε_i, e_i, $\forall i \in N$. Set $k = 0$. Use stabilizing u_j in (5.23);

2: **Outer k iteration loop based on IOC**

3: **Inner l iteration loop using optimal control:** given k, set $l = 0$;

4: Solve the N-tuple of costs and control inputs using (5.24);

5: Stop if $\|V_i^{kl} - V_i^{k(l-1)}\| \leq \varepsilon_i$, then set $u_i^k = u_i^{kl}$ and $V_i^k = V_i^{kl}$, otherwise set $l \leftarrow l + 1$ and go to Step 4;

6: **Outer k iteration loop based on IOC:** update the performance weight Q_i^{k+1} using u_{ie}^* by (5.25);

7: Stop if $\|u_i^k - u_{ie}^*\| \leq e_i$, otherwise set $u_i^{(k+1)0} = u_i^k$, $k \leftarrow k + 1$ and go to Step 3.

The next theorem shows the same convergence properties between Algorithm 5.2 and Algorithm 5.1.

Theorem 5.3. *Algorithm 5.2 converges to Algorithm 5.1 and obtains the same convergence.*

Proof. First, given j, $j = 0, 1, \cdots$, and Q_l^j, one divides both the sides of (5.24) by T and takes the limit to be

$$\begin{aligned} &\lim_{T\to 0} \frac{V_i^{kl}(x(t+T)) - V_i^{kl}(x(t))}{T} \\ &+ \lim_{T\to 0} \frac{2\int_t^{t+T} \left(q^T(x) Q_i^k q(x) + \sum_{j=1}^N (u_j^{kl})^T R_j u_j^{kl}\right) d\tau}{T} \\ &+ \lim_{T\to 0} \frac{\int_t^{t+T} \left(2(u_i^{k(l+1)})^T R_i \sum_{j=1}^N (u_j - u_j^{kl}\right) d\tau}{T} = 0. \end{aligned} \tag{5.26}$$

By LâHopitalâs rule, (5.26) becomes

$$
\begin{aligned}
&(\nabla V_i^{kl})^T\Big(f(x)+\sum_{j=1}^{N} g(x)u_j^{kl}+\sum_{j=1}^{N} g(x)(u_j-u_j^{kl}))\Big)\\
&+q^T(x)Q_i^k q(x)+\sum_{j=1}^{N}(u_j^{kl})^T R_j u_j^{kl}\\
&+2(u_i^{k(l+1)})^T R_i\sum_{j=1}^{N}(u_j-u_j^{kl})=0.
\end{aligned}
\tag{5.27}
$$

Then, submitting $u_i^{k(l+1)}$ (5.11) into (5.27) yields (5.10). This implies that (5.9) gives the same solution as the Lyapunov function (5.10) with the inputs (5.11).

Similarly, one divides both the sides of (5.25) by T and takes the limit. This yields (5.12) and shows that (5.25) gives the same solution as (5.12). Thus, Algorithm 5.2 converges to Algorithm 5.1. Algorithm 5.2 obtains a convergence $(x, u_1^k, \cdots, u_N^k) \to (x, u_{1e}^*, \cdots, u_{Ne}^*)$. □

5.4.2 Implementation of Inverse RL Algorithm 5.2 via NNs

Now, one implements the model-free Algorithm 5.2 based on NNs. Two NN-based approximators are designed for V_i^{kl} and $u_i^{k(l+1)}$ in (5.24). According to approximation method in [116], two NNs are given by

$$\hat{V}_i^{kl}=(W_i^{kl})^T\phi_i(x), \tag{5.28}$$

$$\hat{u}_i^{k(l+1)}=(H_i^{kl})^T\varphi_i(x), \tag{5.29}$$

where $W_i^{kl}\in\mathbb{R}^{M_i}$ and $H_i^{kl}\in\mathbb{R}^{m\times J_i}$ are unknown NN weights of (5.28) and (5.29), respectively, $\phi_i(x)=[\phi_{i_1}(x)\ \phi_{i_2}(x)\ \cdots\ \phi_{i_{M_i}}(x)]^T$ and $\varphi_i(x)=[\varphi_{i_1}(x)\ \varphi_{i_2}(x)\ \cdots\ \varphi_{i_{J_i}}(x)]^T$ are two activation functions with hidden-layer neuron numbers M_i and J_i, respectively.

Let $u_i - \hat{u}_i^{kl} \triangleq \big[\tilde{u}_{i_1}^{kl},\ \tilde{u}_{i_2}^{kl},\ \cdots,\ \tilde{u}_{i_m}^{kl}\big]^T$. Then, one rewrites (5.24) with residual errors $\hat{\epsilon}_i^{kl}$ as

$$
\begin{aligned}
&(W_i^{kl})^T\big(\phi_i(x(t+T) - \phi_i(x(t))\big) \qquad (5.30)\\
&= -\int_t^{t+T}\Big(q^T(x)Q_i^k q(x) + \sum_{j=1}^{N}(\hat{u}_j^{kl})^T R_j \hat{u}_j^{kl}\Big)d\tau - 2\sum_{h=1}^{m}(H_{i_h}^{kl})^T r_{i_h}\int_t^{t+T}\varphi_i(x)\sum_{j=1}^{N}\tilde{u}_{j_h}^{kl}d\tau + \hat{\epsilon}_i^{kl}
\end{aligned}
$$

which is used to solve the unknown NN weight set $\hat{W}_i^{kl} \triangleq \big(W_i^{kl},\ H_{i_1}^{kl},\ \cdots,\ H_{i_m}^{kl}\big)$ for all $i \in N$ given Q_i^k by batch least squares (BLSs) method. Based on the method of weighted residuals [27], $\hat{W}_i^{kl}$ is determined by projecting $\hat{\epsilon}_i^{kl}$ onto $\mathrm{d}\hat{\epsilon}_i^{kl}/\mathrm{d}\hat{W}_i^{kl}$ and setting the inner product to zero, i.e., $\langle \frac{\mathrm{d}\hat{\epsilon}_i^{kl}}{\mathrm{d}\hat{W}_i^{kl}}, \hat{\epsilon}_i^{kl}\rangle \equiv 0$. That is, solve $\hat{W}_i^{kl}$ by letting $\hat{\epsilon}_i^{kl} \equiv 0$ in (5.30) (also see [68]). To use BLSs for (5.30), one defines

$$
\Psi_i^{kl} = \begin{bmatrix}\gamma_{i_1}\\ \gamma_{i_2}\\ \vdots\\ \gamma_{i_{\iota_i}}\end{bmatrix}, \Pi_i^{kl} = \begin{bmatrix}\pi_{i1}^{(x)} & \pi_{i1}^{(\hat{u}_1)} & \cdots & \pi_{i1}^{(\hat{u}_m)}\\ \pi_{i2}^{(x)} & \pi_{i2}^{(\hat{u}_1)} & \cdots & \pi_{i2}^{(\hat{u}_m)}\\ \vdots & \vdots & \ddots & \vdots\\ \pi_{i\iota_i}^{(x)} & \pi_{i\iota_i}^{(\hat{u}_1)} & \cdots & \pi_{i\iota_i}^{(\hat{u}_m)}\end{bmatrix}, \qquad (5.31)
$$

where

$$
\begin{aligned}
\gamma_{i_{\iota_i}} &= \int_{t+(\iota_i-1)T}^{t+\iota_i T}\Big(q^T(x)Q_i^k q(x) + \sum_{j=1}^{N}(\hat{u}_j^{kl})^T R_j \hat{u}_j^{kl}\Big)d\tau,\\
\pi_{i\iota_i}^{(x)} &= \phi_i^T(x(t+\iota_i T - T)) - \phi_i^T(x(t+\iota_i T)),\\
\pi_{i\iota_i}^{\hat{u}1} &= -r_{i_1}\int_{t+\iota T-T}^{t+\iota_i T}\varphi_i^T(x)\tilde{u}_{i_1}^{kl}d\tau,\\
\pi_{i\iota_i}^{\hat{u}m} &= -r_{i_m}\int_{t+\iota_i T-T}^{t+\iota_i T}\varphi_i^T(x)\tilde{u}_{i_m}^{kl}d\tau.
\end{aligned}
$$

Based on (5.30)-(5.31), one uses

$$
\Big[(W_i^{kl})^T\ (H_{i_1}^{kl})^T\ \cdots\ (H_{i_m}^{kl})^T\Big]^T = \Big((\Pi_i^{kl})^T\Pi_i^{kl}\Big)^{-1}(\Pi_i^{kl})^T\Psi_i^{kl} \qquad (5.32)
$$

to uniquely solve $\hat{W}_i^{kl} \to \hat{W}_i^k$. The uniqueness of $\hat{W}_i^k$ is guaranteed by collecting enough data in (5.31), such that $(\Pi_i^{kl})^T \Pi_i^{kl}$ in (5.32) has full rank, $\forall i \in N$. This is guaranteed by having $\iota_i \geq M_i + m_i \times J_i$ hold, where $M_i + m_i \times J_i$ is the number of unknown parameters in (5.30).

With the demonstration u_{ie}^* and the converged solution set $\hat{W}_i^k$ from (5.32), the learner updates Q_i^{k+1} by

$$\begin{aligned} &\int_t^{t+T} q^T(x) Q_i^{k+1} q(x) + (W_i^k)^T \pi_{i1}^x \\ &= \int_t^{t+T} \Big(u_{ie}^{*T} R_i u_{ie}^* - 2 u_{ie}^{*T} R_i W_i^k \varphi_i(x) - \sum_{j \in -i} \bar{\varphi}_j^k(x) \Big) d\tau \end{aligned} \tag{5.33}$$

with π_{i1}^x defined in (5.31) and $\bar{\varphi}_j^k(x) = \varphi_j^T H_j^k R_j (H_j^k)^T \varphi_j$.

Using Kronecker product, one rewrites (5.33) as

$$\int_t^{t+T} \text{vecv}(q(x))^T d\tau \ \text{vecs}(Q_i^{k+1}) = \delta_{i_1}^k \tag{5.34}$$

and defines

$$\Gamma_i = \begin{bmatrix} \int_t^{t+T} \text{vecv}(q(x))^T d\tau \\ \int_{t+T}^{t+2T} \text{vecv}(q(x))^T d\tau \\ \vdots \\ \int_{t+(\mu_i-1)T}^{t+\mu_i T} \text{vecv}(q(x))^T d\tau \end{bmatrix}, \eta_i^k = \begin{bmatrix} \delta_{i_1}^k \\ \delta_{i_2}^k \\ \vdots \\ \delta_{i_{\mu_i}}^k \end{bmatrix}, \tag{5.35}$$

where $\delta_{i_1}^k = \int_{t+(\mu_i-1)T}^{t+\mu_i T} (u_{ie}^{*T} R_i u_{ie}^* - 2 u_{ie}^{*T} R_i W_i^k \varphi_i(x) - \sum_{j \in i} \bar{\varphi}_j^k(x)) d\tau - (W_i^k)^T \pi_{i1}^x$.

Let $\mu_i \geq n+1$ hold, such that $\text{rank}(\Gamma_i^T \Gamma_i) = n+1$. One thus uniquely solves Q_i^{k+1} by

$$\text{vecs}(Q_i^{k+1}) = \left(\Gamma_i^T \Gamma_i\right)^{-1} \Gamma_i^T \eta_i^k. \tag{5.36}$$

Remark 5.4. *Referring to the convergence of NN approximation in [42, 68], one can infer that* $\hat{V}_i^{kl} \to V_i^k$ *and* $\hat{u}_i^{k(l+1)} \to u_i^k$ *as* $M_i \to \infty$ *and* $N_i \to \infty$ *in* (5.30)*. Then,* Q_i^{k+1} *in* (5.36) *uniformly*

converges to that in (5.12). *Thus, the solutions of implementations* (5.30) *and* (5.36) *converge to that of Algorithm 5.2 and Algorithm 5.1 based on Theorem 5.3. This also implies that the NN solutions are stabilizing [68]. Similarly, for the cases of heterogeneous control inputs in next section, this work can also obtain the uniformly convergence of the implementations.*

5.5 Partially Model-free Inverse RL for Heterogeneous Control Inputs

This section proposes a partially model-free inverse RL algorithm using NNs for the case of heterogeneous control inputs, that is, g_i can be different for different $i \in N$. The algorithm needs the control input dynamics $g_1, g_2, \cdots, g_N$ to be known but without knowing f.

5.5.1 Inverse RL for Heterogeneous Control Inputs

One rewrites the learner system (5.23) with heterogeneous control input dynamics g_j for all $j \in N$ as

$$\dot{x} = f(x) + \sum_{j=1}^{N} g_j(x)u_j^{kl} + \sum_{j=1}^{N} g_j(x)(u_j - u_j^{kl}). \tag{5.37}$$

Then, one refers to the off-policy integral RL for multi-player games in [105] and thus obtains

$$\begin{aligned} &V_i^{kl}(x(t+T)) - V_i^{kl}(x(t)) \\ &= \int_t^{t+T} \Big(- q^T(x)Q_i^k q(x) - \sum_{j=1}^{N} (u_j^{kl})^T R_j u_j^{kl} \Big) d\tau - \int_t^{t+T} \Big((\nabla V_i^{kl})^T \sum_{j=1}^{N} g_j(x)(u_j - u_j^{kl}) \Big) d\tau \end{aligned} \tag{5.38}$$

which is combined with (5.11) to obtain the converged (V_i^k, u_i^k) using PI given Q_i^k and g_j. Then, the learner updates Q_i^{k+1} using IOC by

$$\begin{aligned} &\int_t^{t+T} q^T(x)Q_i^{k+1}q(x)d\tau + V_i^k\big(x(t+T)\big) - V_i^k\big(x(t)\big) \\ &= \int_t^{t+T} \Big(u_{ie}^{*T} R_i u_{ie}^* - 2u_{ie}^{*T} R_i u_i^k - \sum_{j \in -i} (u_j^k)^T R_j u_j^k \Big) d\tau. \end{aligned} \tag{5.39}$$

The partially model-free integral inverse RL algorithm is given in Algorithm 5.3.

Algorithm 5.3 Partially Model-free Integral Inverse RL Algorithm for Heterogeneous Control Inputs

1: Select $Q_i^0 > 0$, stabilizing u_i^{00}, and small thresholds ε_i, e_i, $\forall i \in N$. Set $k = 0$. Use stabilizing u_j in (5.37);

2: **Outer k iteration loop based on IOC**

3: **Inner l iteration loop using optimal control:** given k, set $l = 0$;

4: Solve N-tuple of costs by (5.38) and update N-tuple of control inputs by (5.11);

5: Stop if $\|V_i^{kl} - V_i^{k(l-1)}\| \leq \varepsilon_i$, then set $u_i^k = u_i^{kl}$ and $V_i^k = V_i^{kl}$, otherwise set $l \leftarrow l + 1$ and go to Step 4;

6: **Outer k iteration loop based on IOC:** update the performance weight Q_i^{k+1} using u_{ie}^* by (5.39);

7: Stop if $\|u_i^k - u_{ie}^*\| \leq e_i$, otherwise set $u_i^{(k+1)0} = u_i^k$, $k \leftarrow k + 1$ and go to Step 3.

5.5.2 Implementation of Inverse RL Algorithm 5.3 via NNs

To implement Algorithm 5.3, the NN approximator (5.28) is used for the value function V_i^{kl}. Then, $\bar{u}_i^{k(l+1)}$ denotes the estimate of $u_i^{k(l+1)}$ and is uniformly approximated as

$$\bar{u}_i^{k(l+1)} = -\frac{1}{2} R_i^{-1} g_i^T(x) \nabla \phi_i^T(x) W_i^{kl}. \tag{5.40}$$

Next, (5.38) is approximated as

$$\begin{aligned} &(W_i^{kl})^T \big(\phi_i(x(t+T) - \phi_i(x(t))\big) \\ &= \int_t^{t+T} \Big(- q^T(x) Q_i^k q(x) - \sum_{j=1}^N (\bar{u}_j^{kl})^T R_j \bar{u}_j^{kl} \Big) d\tau + \bar{\epsilon}_i^{kl} \\ &\quad + (W_i^{kl})^T \int_t^{t+T} \Big(\nabla \phi_i(x(\tau)) \sum_{j=1}^N g_j(x)(u_j - \bar{u}_j^{kl}) \Big) d\tau. \end{aligned} \tag{5.41}$$

Again, use the weighted residual method in (5.41), e.g., solve W_i^{kl} by letting $\bar{\epsilon}_i^{kl} \equiv 0$ and using

BLSs. One has

$$W_i^{kl} = \Big((\Phi_i^{kl})^T \Phi_i^{kl}\Big)^{-1} (\Phi_i^{kl})^T \Omega_i^{kl}, \tag{5.42}$$

where

$$\begin{aligned}
\Omega_i^{kl} &= \big[\omega_{i_1}, \omega_{i_2}, \cdots, \omega_{i_{c_i}}\big]^T, \\
\Phi_i^{kl} &= \big[(\bar{\phi}_{i_1} + \kappa_{i_1})^T, (\bar{\phi}_{i_2} + \kappa_{i_2})^T, \cdots, (\bar{\phi}_{i_{c_i}} + \kappa_{i_{c_i}})^T\big]^T, \\
\omega_{i_{c_i}} &= \int_{t+(c_i-1)T}^{t+c_iT} \big(q^T(x) Q_i^k q(x) + \sum_{j=1}^{N} (\bar{u}_j^{kl})^T R_j \bar{u}_j^{kl}\big) d\tau, \\
\bar{\phi}_{i_{c_i}} &= -\phi_i^T(x(t+c_iT)) + \phi_i^T(x(t+(c_i-1)T)), \\
\kappa_{i_{c_i}} &= -\int_{t+(c_i-1)T}^{t+c_iT} \nabla\phi_i \sum_{j=1}^{N} g_j(x)(u_j - \bar{u}_j^{kl}) d\tau.
\end{aligned}$$

To find unique solution of (5.42), one requires that $c_i \geq M_i$ such that rank$((\Phi_i^{kl})^T \Phi_i^{kl}) = M_i$. When W_i^{kl} converges to W_i^k, one defines $\hat{\varphi}_j^k = (W_j^k)^T \nabla\varphi_j \bar{g}_j \nabla\varphi_j^T W_j^k$ and refers to (5.39) so as to update Q_i^{k+1} by

$$\begin{aligned}
&\int_t^{t+T} q^T(x) Q_i^{k+1} q(x) + (W_i^k)^T \big(\phi_i(x(t+T)) - \phi_i(x(t))\big) \\
&= \int_t^{t+T} \Big(u_{ie}^{*T} R_i u_{ie}^* + u_{ie}^{*T} g_i^T \nabla\varphi_i(x) W_i^k - \frac{1}{4} \sum_{j \in -i} \hat{\varphi}_j^k\Big) d\tau.
\end{aligned} \tag{5.43}$$

Let $\mu_i \geq n+1$ hold, such that rank$(\Gamma_i^T \Gamma_i) = n+1$ with Γ_i defined in (5.35). One uniquely solves Q_i^{k+1} by

$$\text{vecs}(Q_i^{k+1}) = \big(\Gamma_i^T \Gamma_i\big)^{-1} \Gamma_i^T \theta_i^k, \tag{5.44}$$

where

$$\theta_i^k = [h_{i_1}^k \quad h_{i_2}^k \quad \cdots \quad h_{i_{\mu_i}}^k]^T,$$
$$h_{i_{\mu_i}}^k = \int_{t+(\mu_i-1)T}^{t+\mu_i T} \big(u_{ie}^{*T} R_i u_{ie}^* + u_{ie}^{*T} g_i^T(x) \nabla\varphi_i(x) W_i^k - \sum_{j\in -i} \hat{\varphi}_j^k\big) d\tau$$
$$- (W_i^k)^T \big(\phi_i(x(t+T)) - \phi_i(x(t))\big).$$

5.6 Simulation Studies

This section presents two examples to verify the proposed algorithms: Example 1 with homogeneous control inputs using Algorithm 5.2, and Example 2 with heterogeneous control inputs using Algorithm 5.3.

Consider both the learner and the expert to be the four-player nonlinear system:

$$\dot{s} = f(s) + g_1(s)v_1 + g_2(s)v_2 + g_3(s)v_3 + g_4(s)v_4, \tag{5.45}$$

where s denotes x or x_e, v_i denotes u_i or u_{ie}, $i \in \{1, 2, 3, 4\}$.

Example 1: Consider the system (5.45) with homogeneous control inputs as

$$f(s) = \begin{bmatrix} -s_1 \\ -s_2^3 \end{bmatrix}, \; g_j(s) = \begin{bmatrix} 0 \\ s_1 \end{bmatrix}, \forall i \in \{1, 2, 3, 4\},$$

where s_n denotes the nth element of the state x or x_e.

Consider the function $q(s)$ in both (5.2) and (5.6) as $q(s) = \begin{bmatrix} s_1^2 & s_2^2 \end{bmatrix}^T$. Select

$$Q_{1e} = \begin{bmatrix} 2 & 0.56 \\ 0.56 & 2 \end{bmatrix}, Q_{2e} = \begin{bmatrix} 1.6 & 0.1 \\ 0.1 & 1 \end{bmatrix}, \tag{5.46}$$

$$Q_{3e} = \begin{bmatrix} 2.4 & 0.73 \\ 0.73 & 2 \end{bmatrix}, Q_{4e} = \begin{bmatrix} 1.2 & 0.1 \\ 0.1 & 1 \end{bmatrix}, \tag{5.47}$$

$R_{1e} = 2$, $R_{2e} = 1$, $R_{3e} = 3$ and $R_{4e} = 1.5$ for (5.2). According to the converse Hamilton-Jacobi-Bellman approach [78], one obtains the reward functions of the expert players to be $V_{1e} = 0.4x_{e_1}^4 + x_{e_2}^2$, $V_{2e} = 0.6x_{e_1}^4 + 0.5x_{e_2}^2$, $V_{3e} = 0.5x_{e_1}^4 + x_{e_2}^2$ and $V_{4e} = 0.3x_{e_1}^4 + 0.5x_{e_2}^2$.

For the learner rewards (5.6), select $R_1 = 3$, $R_2 = 2$, $R_3 = 1.5$, $R_4 = 1$ and the initial $Q_1^0 = Q_2^0 = Q_3^0 = Q_4^0 = 0.5 * I_2$. Select the activation functions as $\phi_i(x) = [x_1^4\ x_1^2x_2^2\ x_2^4\ x_1^2\ x_2^2]^T$ and $\varphi_i(x) = [x_1^4\ x_2^4\ x_1^3x_2\ x_1x_2^3\ x_1x_2]^T$ for the learner. Set initial states $x(0) = x_e(0) = [3\ -3]$ and small thresholds $\varepsilon_i = e_i = 0.001$.

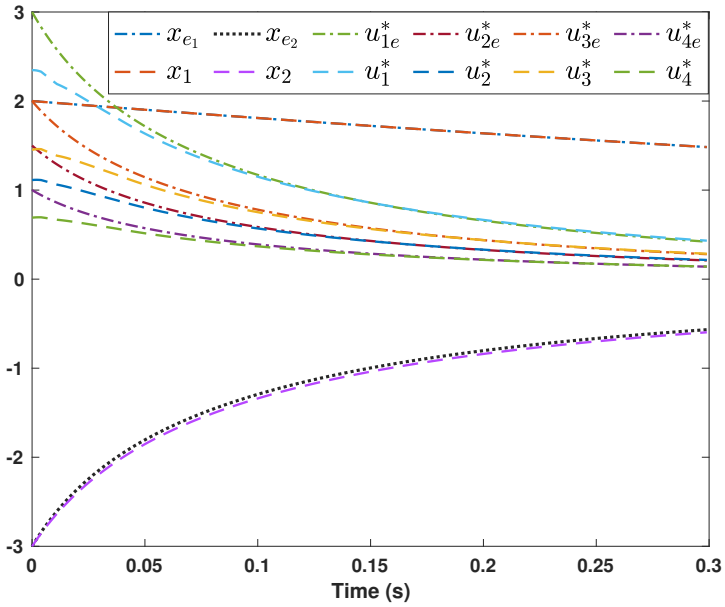
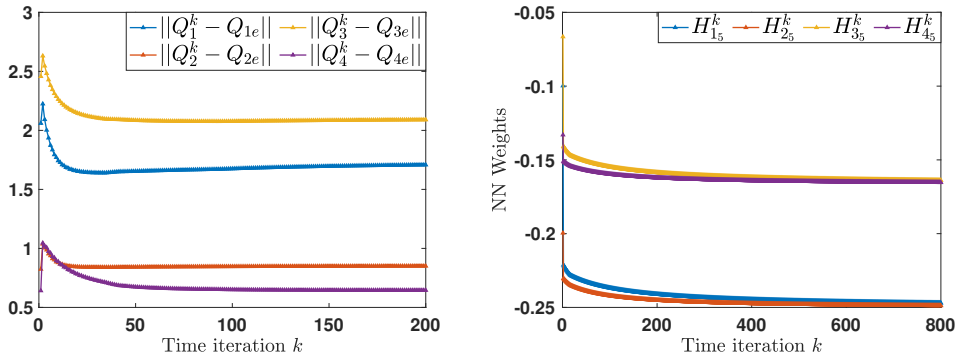


Figure 5.1: Trajectories of the learner and the expert.

Figure 5.1 shows that the learner's trajectories gradually converge to that of expert's by imple-



(a) (b)

Figure 5.2: (a) Iteration of performance weights Q_i^k; (b) Iteration of NN weights $H_{i_5}^k$ for all $i \in \{1, 2, 3, 4\}$ using Algorithm 5.2.

menting Algorithm 5.2. Here, the learner's trajectories are plotted using the learning NN weights instead of converged weights. Figure 5.2 (a) shows the norm of $\|Q_i^k - Q_{ie}\|$ for each $i \in \{1, 2, 3, 4\}$. Note that Q_i^k of each i converges. This is consistent with the convergence analysis in Theorem 5.1. The converged value $\tilde{Q}_i$ is not equal to Q_{ie} because, as shown in Definition 5.3, multiple different expert's reward functions can yield the same behaviour. Figure 5.2 (b) shows the corresponding convergence of the NN weights $H_{i_5}^k$, which leads to $u_i^k \to u_{ie}^*$, $i = \{1, 2, 3, 4\}$. Here, $H_{i_5}^k$ is used to represent the results of all NN weights because $x_1 x_2$ is the only term to determine u_i^k.

Example 2: Consider the system (5.45) with heterogeneous control inputs as

$$g_1(s) = \begin{bmatrix} 0 \\ s_1 \end{bmatrix}, g_2(s) = \begin{bmatrix} 0 \\ s_2 \end{bmatrix}, g_3(s) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, g_4(s) = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

and $f(s) = \begin{bmatrix} -6s_1 \\ -s_2^3 + s_2 \end{bmatrix}$. Then, select the weights

$$Q_{1e} = \begin{bmatrix} 3.6 & -0.18 \\ -0.18 & 1.42 \end{bmatrix}, Q_{2e} = \begin{bmatrix} 7.2 & -0.42 \\ -0.42 & 1.38 \end{bmatrix},$$
$$Q_{3e} = \begin{bmatrix} 4 & -0.25 \\ -0.25 & 0.5 \end{bmatrix}, Q_{4e} = \begin{bmatrix} 8 & -0.12 \\ -0.12 & 0.3 \end{bmatrix},$$

and $R_{1e} = R_{2e} = 1$, $R_{3e} = R_{4e} = 0.5$. Based on [78], the reward functions of expert's four players are $V_{1e} = 0.225x_{e_1}^4 + 0.65x_{e_2}^2$, $V_{2e} = 0.45x_{e_1}^4 + 0.65x_{e_2}^2$, $V_{3e} = 0.25x_{e_1}^4 + x_{e_2}^2$ and $V_{4e} = 0.5x_{e_1}^4 + x_{e_2}^2$.

In the performance (5.6), select $R_1 = 0.5$, $R_2 = 0.25$, $R_3 = 1$, $R_4 = 1.5$, and initial $Q_1^0 = Q_2^0 = \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}$ and $Q_3^0 = Q_4^0 = I_2$.

Implement the Algorithm 5.3 by selecting the activation function for V_i^k as $\phi_i(x) = \begin{bmatrix} x_1^4 & x_1^2 x_2^2 & x_2^4 & x_1^2 & x_1 x_2 & x_2^2 \end{bmatrix}^T$. Set initial $s(0) = [6 \ -6]$ and thresholds $\varepsilon_i = e_i = 0.001$.

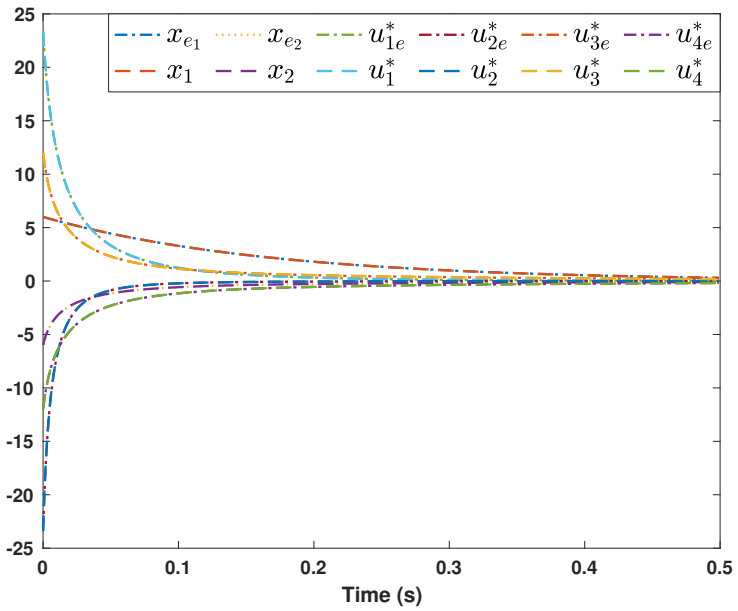
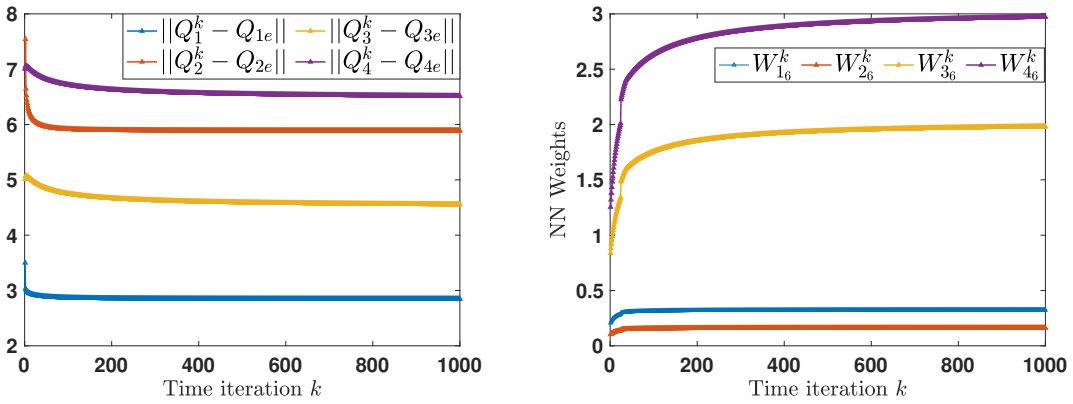


Figure 5.3: Trajectories of the learner and the expert.



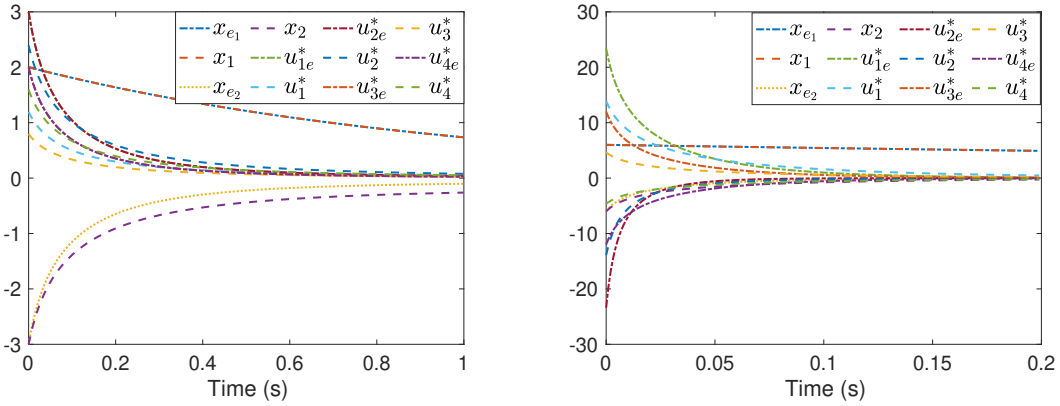
(a) (b)

Figure 5.4: (a) Iteration of performance weights Q_i^k; (b) Iteration of NN weights $W_{i_5}^k$ for all $i \in \{1, 2, 3, 4\}$ using Algorithm 5.3.

Figure 5.3 plots the learner's trajectories by using the converged NN weights. They are the same as that of the expert. Figure 5.4 (a) shows the norm of $\|Q_i^k - Q_{ie}\|$. Again, note that Q_i^k converges but does not equal to Q_{ie} for all $i \in \{1, 2, 3, 4\}$ because of the degeneracy of the optimal reward function. Figure 5.4 (b) shows the convergence of the NN weights $W_{i_5}^k$, which leads to $u_i^k \to u_{ie}^*$. Here, $W_{i_5}^k$ is used to represent the results of NN weights W_i^k because W_{i_5} is the key element to determine u_i^k.

Now one compares the imitation performance between RL [105] and the proposed inverse RL methods for two examples above. For RL, set the reward weights as the same as that of the learner. Other parameters are kept the same. Figure 5.5 (a) and Figure 5.5 (b) show the imitation tracking performance using RL for examples 1 and 2, respectively. Note that they have slower imitation speed compared with Figure 5.1 and Figure 5.3.

This work concludes that because the proposed inverse RL methods reconstructs experts' reward weights and control policies, this yields a better imitation performance than RL that manually specify rewards. Thus, inverse RL is a useful tool for the applications of tracking and imitation learning compared with RL [2, 3]. However, as the computational cost is the main problem in inverse RL problems [1], the computational efficiency is encouraged while designing new inverse RL methods in the future.



(a) (b)

Figure 5.5: Trajectories of the learner and the expert using RL for (a) example 1 and (b) example 2.

5.7 Conclusion

This chapter studies inverse RL for nonlinear continuous-time systems described by multi-player differential dynamic equations. This chapter fills in the gap that inverse RL has not been studied for multiplayer apprentice games described by nonlinear differential equations. The games are solved by the learner using the expert's trajectories to find the expert's performance reward functions. Based on a rigorously developed model-based inverse RL algorithm, this work proposes a completely model-free inverse RL algorithm for homogeneous control inputs and a partially model-free inverse RL algorithm for heterogeneous control inputs. The proposed framework may be potentially extended to multi-agent graphical games where there are more than one expert and more than one follower learner.

Chapter 6: FUTURE DIRECTIONS

We envision a few research directions that can further improve on our work.

1. In the studies of distributed Kalman filtering for consensus problems, the future work could focus on the finite-time stability or state constraints [87, 108] or the effects of impaired communication channel [102].

2. In the studies of robustness margins for distributed estimation problems, the future might be potentially extended to some moving-horizon estimation schemes as in [131, 132].

3. In the studies of inverse RL for two-player zero-sum games, the future work could focus on the design of online tuning algorithms where the cost functions, control polices, and state-reward weights update simultaneously.

4. In the studies of inverse RL for multiplayer non-zero-sum games, it would be more interesting to discuss the one-loop inverse RL algorithms that are more computationally efficient because the inverse RL in this work is two-loop and scales poor with the increased dimension of states and players.

BIBLIOGRAPHY

[1] Nematollah Ab Azar, Aref Shahmansoorian, and Mohsen Davoudi. From inverse optimal control to inverse reinforcement learning: A historical review. *Annual Reviews in Control*, 50:119–138, 2020.

[2] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[3] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine learning*, pages 1–8, Louisville, USA, 2004.

[4] Hisham Abou-Kandil, Gerhard Freiling, Vlad Ionescu, and Gerhard Jank. *Matrix Riccati Equations In Control And Systems Theory*. Birkhäuser, 2012.

[5] Murad Abu-Khalaf, Frank L Lewis, and Jie Huang. Neurodynamic programming and zero-sum games for constrained control systems. *IEEE Transactions on Neural Networks and Learning Systems*, 19(7):1243–1252, 2008.

[6] Brian DO Anderson, JB Pren, and SL Dickerson. *Linear Optimal Control*. Prentice-Hall: Upper Saddle River, NJ, USA, 1971.

[7] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, page 103500, 2021.

[8] Christopher G Atkeson and Stefan Schaal. Robot learning from demonstration. In *Proceedings of the 14th International Conference on Machine Learning*, volume 97, pages 12–20, Nashville, USA, 1997.

[9] Monica Babes, Vukosi N Marivate, Kaushik Subramanian, and Michael L Littman. Apprenticeship learning about multiple intentions. In *International Conference on Machine Learning*, 2011.

[10] Tamer Başar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory*. SIAM, 1998.

[11] Giorgio Battistelli, Luigi Chisci, Giovanni Mugnai, Alfonso Farina, and Antonio Graziano. Consensus-based linear and nonlinear filtering. *IEEE Transactions on Automatic Control*, 60(5):1410–1415, 2015.

[12] Vinay A Bavdekar, Anjali P Deshpande, and Sachin C Patwardhan. Identification of process and measurement noise covariance for state and parameter estimation using extended kalman filter. *Journal of Process control*, 21(4):585–601, 2011.

[13] Hendrik Wade Bode et al. Network analysis and feedback amplifier design. 1945.

[14] Daniel Brown and Scott Niekum. Efficient probabilistic performance bounds for inverse reinforcement learning. In *Proceedings of the Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, volume 32, 2018.

[15] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, 2(5):483–502, 2002.

[16] Ci Chen and Anthony Holohan. Stability robustness of linear quadratic regulators. *International Journal of Robust and Nonlinear Control*, 26(9):1817–1824, 2016.

[17] Seungwon Choi, Suseong Kim, and H Jin Kim. Inverse reinforcement learning control for trajectory tracking of a multirotor uav. *International Journal of Control, Automation and Systems*, 15(4):1826–1834, 2017.

[18] Kai-Fung Chu, Albert YS Lam, Chenchen Fan, and Victor OK Li. Disturbance-aware neuro-optimal system control using generative adversarial control networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020, doi: 10.1109/TNNLS.2020.3022950.

[19] Domenico Ciuonzo and P Salvo Rossi. Distributed detection of a non-cooperative target via generalized locally-optimum approaches. *Information Fusion*, 36:261–274, 2017.

[20] Domenico Ciuonzo and P Salvo Rossi. Quantizer design for generalized locally optimum detectors in wireless sensor networks. *IEEE Wireless Communications Letters*, 7(2):162–165, 2017.

[21] Abhijit Das, Frank Lewis, and Kamesh Subbarao. Backstepping approach for controlling a quadrotor using lagrange form dynamics. *Journal of Intelligent and Robotic Systems*, 56(1):127–151, 2009.

[22] Manuel De la Sen and Santiago Alonso-Quesada. On finite-time consensus objectives in time-varying interconnected discrete linear dynamic systems under internal and external delays. *Advances in Mechanical Engineering*, 10(7):1–24, 2018.

[23] Michael A Demetriou. Guidance of mobile actuator-plus-sensor networks for improved control and estimation of distributed parameter systems. *IEEE Transactions on Automatic Control*, 55(7):1570–1584, 2010.

[24] Christos Dimitrakakis and Constantin A Rothkopf. Bayesian multitask inverse reinforcement learning. In *European workshop on reinforcement learning*, pages 273–284. Springer, 2011.

[25] John Doyle and Guter Stein. Robustness with observers. *IEEE Transactions on Automatic Control*, 24(4):607–611, 1979.

[26] John C Doyle. Guaranteed margins for lqg regulators. *IEEE Transactions on Automatic Control*, 23(4):756–757, 1978.

[27] Bruce A Finlayson. *The Method Of Weighted Residuals And Variational Principles*. SIAM, 2013.

[28] Paul Frihauf, Miroslav Krstic, and Tamer Basar. Nash equilibrium seeking in noncooperative games. *IEEE Transactions on Automatic Control*, 57(5):1192–1207, 2012.

[29] Xiaohua Ge, Qing-Long Han, Xian-Ming Zhang, Lei Ding, and Fuwen Yang. Distributed event-triggered estimation over sensor networks: A survey. *IEEE Transactions on Cybernetics*, 50(3):1306–1320, 2020.

[30] Xiaohua Ge, Qing-Long Han, Maiying Zhong, and Xian-Ming Zhang. Distributed krein space-based attack detection over sensor networks under deception attacks. *Automatica*, 109(108557), 2019.

[31] Wassim M Haddad and VijaySekhar Chellaboina. *Nonlinear Dynamical Systems And Control: A Lyapunov-Based Approach*. Princeton University Press, Princeton, USA, 2011.

[32] Hamid R Hashemipour, Sumit Roy, and Alan J Laub. Decentralized structures for parallel kalman filtering. *IEEE Transactions on Automatic Control*, 33(01):88–94, 1988.

[33] Iyad Hashlamon. A new adaptive extended kalman filter for a class of nonlinear systems. *Journal of Applied and Computational Mechanics*, 6(1):1–12, 2019.

[34] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *NeurIPS 30th Annual Conference on Neural Information Processing Systems*, pages 4565–4573, 2016.

[35] Wenfeng Hu, Lu Liu, and Gang Feng. Consensus of linear multi-agent systems by distributed event-triggered strategy. *IEEE Transactions on Cybernetics*, 46(1):148–157, 2016.

[36] Esa Hyytia, Pasi Lassila, and Jorma Virtamo. Spatial node distribution of the random waypoint mobility model with applications. *IEEE Transactions on Mobile Computing*, 5(6):680–694, 2006.

[37] Mahdi Imani and Ulisses M Braga-Neto. Control of gene regulatory networks using bayesian inverse reinforcement learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(4):1250–1261, 2018.

[38] Mahdi Imani and Seyede Fatemeh Ghoreishi. Scalable inverse reinforcement learning through multifidelity bayesian optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 2021, doi:10.1109/TNNLS.2021.3051012.

[39] Jairo Inga, Esther Bischoff, Timothy L Molloy, Michael Flad, and Sören Hohmann. Solution sets for inverse non-cooperative linear-quadratic differential games. *IEEE Control Systems Letters*, 3(4):871–876, 2019.

[40] Frédéric Jean and Sofya Maslovskaya. Inverse optimal control problem: the linear-quadratic case. In *57th IEEE Conference on Decision and Control*, pages 888–893, Fontainebleau, USA, 2018.

[41] Honghai Ji, Frank L Lewis, Zhongsheng Hou, and Dariusz Mikulski. Distributed information-weighted kalman consensus filter for sensor networks. *Automatica*, 77:18–30, 2017.

[42] He Jiang, Huaguang Zhang, Yanhong Luo, and Ji Han. Neural-network-based robust control schemes for nonlinear multiplayer systems with uncertainties via adaptive dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(3):579–588, 2019.

[43] Marcus Johnson, Shubhendu Bhasin, and Warren E Dixon. Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 142–147, Orlando, USA, 2011.

[44] Miles Johnson, Navid Aghasadeghi, and Timothy Bretl. Inverse optimal control for deterministic continuous-time nonlinear systems. In *52nd IEEE Conference on Decision and Control*, pages 2906–2913, Firenze, Italy, 2013.

[45] Rudolf Emil Kalman. When is a linear control system optimal? *Journal of Basic Engineering*, 86(1):51–60, 1964.

[46] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[47] Ahmed T Kamal, Jay A Farrell, and Amit K Roy-Chowdhury. Information weighted consensus filters and their application in distributed camera networks. *IEEE Transactions on Automatic Control*, 58(12):3112–3125, 2013.

[48] Ahmed Tashrif Kamal, Jay A Farrell, and Amit K Roy-Chowdhury. Information weighted consensus. In *Proceedings of IEEE Conference on Decision and Control*, pages 2732–2737, Maui, Hawaii, 2012.

[49] Ahmed Tashrif Kamal, Jay A Farrell, Amit K Roy-Chowdhury, et al. Information weighted consensus filters and their application in distributed camera networks. *IEEE Transactions on Automatic Control*, 58(12):3112–3125, 2013.

[50] Peter Lancaster and Leiba Rodman. *Algebraic Riccati Equations*. Clarendon press, 1995.

[51] Norman Lehtomaki, NJAM Sandell, and Michael Athans. Robustness results in linear-quadratic gaussian based multivariable control designs. *IEEE Transactions on Automatic Control*, 26(1):75–93, 1981.

[52] Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. *arXiv preprint arXiv:1206.4617*, 2012.

[53] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. *NeurIPS 25th Annual Conference on Neural Information Processing Systems*, 24:19–27, 2011.

[54] Frank L Lewis, Draguna Vrabie, and Vassilis L Syrmos. *Optimal Control*. John Wiley & Sons, 2012.

[55] Frank L Lewis, Lihua Xie, and Dan Popa. *Optimal And Robust Estimation: With An Introduction to Stochastic Control Theory*. CRC press, 2017.

[56] Frank L Lewis, Hongwei Zhang, Kristian Hengster-Movric, and Abhijit Das. *Cooperative Control of Multi-Agent Systems: Optimal And Adaptive Design Approaches*. Springer Science & Business Media, 2013.

[57] Wangyan Li, Guoliang Wei, Fei Han, and Yurong Liu. Weighted average consensus-based unscented kalman filtering. *IEEE Transactions on Cybernetics*, 46(2):558–567, 2015.

[58] Xin Li, Kejun Wang, Wei Wang, and Yang Li. A multiple object tracking method using kalman filter. In *IEEE International Conference on Information and Automation*, pages 1862–1866, Harbin, China, 2010.

[59] Bosen Lian, Yan Wan, Ya Zhang, Mushang Liu, Frank.L Lewis, and Tianyou Chai. Distributed kalman consensus filter for estimation with moving targets. *IEEE Transactions on Cybernetics (early access) doi:10.1109/TCYB.2020.3029007*, 2020.

[60] Bosen Lian, Yan Wan, Ya Zhang, Mushuang Liu, Frank L Lewis, Alexandra Abad, Tina Setter, Dunham Short, and Tianyou Chai. Distributed consensus-based kalman filtering for estimation with multiple moving targets. *58th Conference on Decision and Control*, pages 3910–3915, 2019.

[61] Bosen Lian, Wenqian Xue, Frank L. Lewis, and Tianyou Chai. Online inverse reinforcement learning for nonlinear systems with adversarial attacks. *International Journal of Robust Nonlinear Control, doi: 10.1002/rnc.5626*, 2021.

[62] Bosen Lian, Wenqian Xue, Frank L. Lewis, and Tianyou Chai. Robust inverse Q-learning for continuous-time linear systems in adversarial environments. *IEEE Transactions on Cybernetics, doi: 10.1109/TCYB.2021.3100749*, 2021.

[63] Xiaomin Lin, Stephen C Adams, and Peter A Beling. Multi-agent inverse reinforcement learning for certain general-sum stochastic games. *Journal of Artificial Intelligence Research*, 66:473–502, 2019.

[64] Xiaomin Lin, Peter A Beling, and Randy Cogill. Multiagent inverse reinforcement learning for two-person zero-sum games. *IEEE Transactions on Games*, 10(1):56–68, 2017.

[65] Michael L Littman. Friend-or-foe Q-learning in general-sum games. In *Proceedings of 8th International Conference on Machine Learning*, volume 1, pages 322–328, Williamstown, USA, 2001.

[66] Mushuang Liu, Yan Wan, and Frank L Lewis. Analysis of the rondom direction mobility model with a sense-and-avoid protocol. In *Proceedings of IEEE Globecom Workshops*, Singapore, 2017.

[67] Victor G Lopez, Yan Wan, and Frank L Lewis. Bayesian graphical games for synchronization in networks of dynamical systems. *IEEE Transactions on Control of Network Systems*, 7(2):1028–1039, 2019.

[68] Biao Luo, Huai-Ning Wu, and Tingwen Huang. Off-policy reinforcement learning for H∞ control design. *IEEE Transactions on Cybernetics*, 45(1):65–76, 2015.

[69] Biao Luo, Yin Yang, and Derong Liu. Policy iteration Q-learning for data-based two-player zero-sum game of linear discrete-time systems. *IEEE Transactions on Cybernetics*, 51(7):3630–3640, 2021.

[70] Lifeng Ma, Zidong Wang, Yun Chen, and Xiaojian Yi. Probability-guaranteed distributed filtering for nonlinear systems with innovation constraints over sensor networks. *IEEE Transactions on Control of Network Systems (Early Access), doi: 10.1109/TCNS.2021.3049361*, 2021.

[71] PD McMorran. Extension of the inverse nyquist method. *Electronics Letters*, 6(25):800–801, 1970.

[72] Hamidreza Modares and Frank L Lewis. Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. *Automatica*, 50(7):1780–1792, 2014.

[73] Hamidreza Modares, Frank L Lewis, and Zhong-Ping Jiang. H∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2550–2562, 2015.

[74] Timothy L Molloy, Jairo Inga, Michael Flad, Jason J Ford, Tristan Perez, and Sören Hohmann. Inverse open-loop noncooperative differential games and inverse optimal control. *IEEE Transactions on Automatic Control*, 65(2):897–904, 2020.

[75] Sriraam Natarajan, Gautam Kunapuli, Kshitij Judah, Prasad Tadepalli, Kristian Kersting, and Jude Shavlik. Multi-agent inverse reinforcement learning. In *2010 Ninth International Conference on Machine Learning and Applications*, pages 395–400. IEEE, 2010.

[76] Gergely Neu and Csaba Szepesvári. Apprenticeship learning using inverse reinforcement learning and gradient methods. *arXiv preprint arXiv:1206.5264*, 2012.

[77] Christian Neumeyer, Frans A Oliehoek, and Dariu M Gavrila. General-sum multi-agent continuous inverse optimal control. *IEEE Robotics and Automation Letters*, 6(2):3429–3436, 2021.

[78] Vesna Nevistić and James A Primbs. Constrained nonlinear optimal control: a converse hjb approach. California Institute of Technology, Pasadena, CA, USA, Tech. Rep. TR96-021, 1996.

[79] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, volume 1, pages 663–670, San Francisco, USA, 2000.

[80] Ruixin Niu and Pramod K Varshney. Performance analysis of distributed detection in a random sensor field. *IEEE Transactions on Signal Processing*, 56(1):339–349, 2008.

[81] Reza Olfati-Saber. Distributed kalman filter with embedded consensus filters. In *Proceedings of IEEE Conference Decision and Control*, pages 8179–8184, Seville, Spain, 2005.

[82] Reza Olfati Saber. Distributed kalman filtering for sensor networks. In *Proceedings of IEEE Conference on Decision and Control*, pages 0191–2216, New Orleans, LA, 2007.

[83] Fernando Ornelas, Edgar N Sanchez, and Alexander G Loukianov. Discrete-time inverse optimal control for nonlinear systems trajectory tracking. In *49th IEEE Conference on Decision and Control*, pages 4813–4818, Atlanta, USA, 2010.

[84] Jau-Woei Perng, Bing-Fei Wu, Hung-I Chin, and Tsu-Tian Lee. Gain-phase margin analysis of dynamic fuzzy control systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(5):2133–2139, 2004.

[85] Tom Pilutti, Galip Ulsoy, and Davor Hrovat. Vehicle steering intervention through differential braking. *Journal of Dynamic Systems, Measurement, and Control*, 120(3):314–321, 1998.

[86] Jianbin Qiu, Gang Feng, and Huijun Gao. Nonsynchronized-state estimation of multichannel networked nonlinear systems with multiple packet dropouts via T-S fuzzy-affine dynamic models. *IEEE Transactions on Fuzzy Systems*, 19(1):75–90, 2011.

[87] Jianbin Qiu, Kangkang Sun, Imre J Rudas, and Huijun Gao. Command filter-based adaptive NN control for MIMO nonlinear systems with full-state constraints and actuator hysteresis. *IEEE Transactions on Cybernetics*, 2019, doi: 10.1109/TCYB.2019.2944761.

[88] Jianbin Qiu, Hui Tian, Qiugang Lu, and Huijun Gao. Nonsynchronized robust filtering design for continuous-time T-S fuzzy affine dynamic systems based on piecewise lyapunov functions. *IEEE Transactions on Cybernetics*, 43(6):1755–1766, 2013.

[89] Zhihua Qu. *Cooperative Control of Dynamical Systems: Applications To Autonomous Vehicles*. Springer Science & Business Media, 2009.

[90] BSY Rao, Hugh F Durrant-Whyte, and JA Sheen. A fully decentralized multi-sensor system for tracking and surveillance. *The International Journal of Robotics Research*, 12(1):20–44, 1993.

[91] Wei Ren, Randal W Beard, and Derek B Kingston. Multi-agent kalman consensus with relative uncertainty. *Proceedings of the American Control Conference*, pages 1865–1870, 2005.

[92] Wei Ren, Randal W Beard, and Derek B Kingston. Multi-agent kalman consensus with relative uncertainty. In *Proceedings of American Control Conference*, pages 1865–1870, Portland, USA, 2013.

[93] HH Rosenbrock. Design of multivariable control systems using the inverse nyquist array. *Proceedings of the Institution of Electrical Engineers*, 116(11):1929–1936, 1969.

[94] Samira Roshany-Yamchi, Marcin Cychowski, Rudy R Negenborn, Bart De Schutter, Kieran Delaney, and Joe Connell. Kalman filter-based distributed predictive control of large-scale multi-rate systems: Application to power networks. *IEEE Transactions on Control Systems Technology*, 21(1):27–39, 2011.

[95] Michael Safonov and Michael Athans. Gain and phase margin for multiloop lqg regulators. *IEEE Transactions on Automatic Control*, 22(2):173–179, 1977.

[96] Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Learning to fly. In *Proceedings of the 9th International Workshop on Machine Learning*, pages 385–393, San Mateo, USA, 1992.

[97] Edgar N Sanchez and Fernando Ornelas-Tellez. *Discrete-Time Inverse Optimal Control For Nonlinear Systems*. CRC Press, Boca Raton, USA, 2017.

[98] Ryan Self, Moad Abudia, and Rushikesh Kamalapurkar. Online inverse reinforcement learning for systems with disturbances. In *2020 American Control Conference*, pages 1118–1123, Denver, USA, 2020.

[99] MANUEL DE LA SEN and CARLOS F ALASTRUEY. Kalman-Bucy filtering for stochastic Volterra models. *International Journal of Systems Science*, 26(2):435–456, 1993.

[100] U Shaked. Guaranteed stability margins for the discrete-time linear quadratic optimal regulator. *IEEE Transactions on Automatic Control*, 31(2):162–165, 1986.

[101] Bo Shen, Zidong Wang, and Yeung Sam Hung. Distributed H_∞ consensus filtering in sensor networks with multiple missing measurements: the finite-horizon case. *Automatica*, 46(10):1682–1688, 2010.

[102] Amirpasha Shirazinia, Subhrakanti Dey, Domenico Ciuonzo, and Pierluigi Salvo Rossi. Massive MIMO for decentralized estimation of a correlated source. *IEEE Transactions on Signal Processing*, 64(10):2499–2512, 2016.

[103] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, And Logical Foundations*. Cambridge University Press, Cambridge, UK, 2008.

[104] Jiaming Song, Hongyu Ren, Dorsa Sadigh, and Stefano Ermon. Multi-agent generative adversarial imitation learning. *arXiv preprint arXiv:1807.09936*, 2018.

[105] Ruizhuo Song, Frank L Lewis, and Qinglai Wei. Off-policy integral reinforcement learning method to solve nonlinear continuous-time multiplayer nonzero-sum games. *IEEE Transactions on Neural Networks and Learning Systems*.

[106] Demetri P Spanos, Reza Olfati-Saber, and Richard M Murray. Dynamic consensus on mobile networks. In *IFAC world congress*, pages 1–6, Prague, Czech, 2005.

[107] Koushil Sreenath, Frank L Lewis, and Dan O Popa. Simultaneous adaptive localization of a wireless sensor network. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(02):14–28, 2007.

[108] Kangkang Sun, Lu Liu, Jianbin Qiu, and Gang Feng. Fuzzy adaptive finite-time fault-tolerant control for strict-feedback nonlinear systems. *IEEE Transactions on Fuzzy Systems*, 2020, doi: 10.1109/TFUZZ.2020.2965890.

[109] Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *NeurIPS 22nd Annual Conference on Neural Information Processing Systems*, pages 1449–1456, 2008.

[110] Sayed Pouria Talebi and Stefan Werner. Distributed kalman filtering and control through embedded average consensus information fusion. *IEEE Transactions on Automatic Control*, 64(10):4396–4403, 2019.

[111] Yang Tang, Huijun Gao, Wei Zou, and Jürgen Kurths. Distributed synchronization in networks of agent systems with nonlinearities and random switchings. *IEEE Transactions on Cybernetics*, 43(1):358–370, 2013.

[112] Yu-Ping Tian and Cheng-Lin Liu. Robust consensus of multi-agent systems with diverse input delays and asymmetric interconnection perturbations. *Automatica*, 45(5):1347–1353, 2009.

[113] Arjan J Van Der Schaft. L 2-gain analysis of nonlinear systems and nonlinear state feedback hâ control. *IEEE Transactions on Automatic Control*, 37(6):770–784, 1992.

[114] Carlos J Vega, Oscar J Suarez, Edgar N Sanchez, Guanrong Chen, Santiago Elvira-Ceja, and David Rodriguez-Castellanos. Trajectory tracking on complex networks via inverse optimal pinning control. *IEEE Transactions on Automatic Control*, 64(2):767–774, 2019.

[115] Yan Wan, Kamesh Namuduri, Yi Zhou, and Shengli Fu. A smooth-turn mobility model for airborne networks. *IEEE Transactions on Vehicular Technology*, 62(7):3359–3370, 2013.

[116] Paul Werbos. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974.

[117] Junfei Xie, Yan Wan, Jae H Kim, Shengli Fu, and Kamesh Namuduri. A survey and analysis of mobility models for airborne networks. *IEEE Communications Surveys & Tutorials*, 16(3):1221–1238, Third Quarter 2014.

[118] Lihua Xie, Dan Popa, and Frank L Lewis. *Optimal And Robust Estimation: With An Introduction To Stochastic Control Theory*. CRC press, 2007.

[119] Lihua Xie, Yeng Chai Soh, and Carlos E De Souza. Robust kalman filtering for uncertain discrete-time systems. *IEEE Transactions on Automatic Control*, 39(6):1310–1314, 1994.

[120] Wenqian Xue, Jialu Fan, Victor G Lopez, Yi Jiang, Tianyou Chai, and Frank L Lewis. Off-policy reinforcement learning for tracking in continuous-time systems on two time scales. *IEEE Transactions on Neural Networks and Learning Systems*, 2020, doi: 10.1109/TNNLS.2020.3017461.

[121] Wenqian Xue, Patrik Kolaric, Jialu Fan, Bosen Lian, Tianyou Chai, and Frank L. Lewis. Inverse reinforcement learning in tracking control based on inverse optimal control. *IEEE Transactions on Cybernetics, doi: 10.1109/TCYB.2021.3062856*, 2021.

[122] Wenqian Xue, Bosen Lian, Jialu Fan, Patrik Kolaric, Tianyou Chai, and Frank L Lewis. Inverse reinforcement Q-learning through expert imitation for discrete-time systems. *IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2021.3106635*, 2021.

[123] Lantao Yu, Jiaming Song, and Stefano Ermon. Multi-agent adversarial inverse reinforcement learning. In *International Conference on Machine Learning*, pages 7194–7201. PMLR, 2019.

[124] Cishen Zhang and Minyue Fu. A revisit to the gain and phase margins of linear quadratic regulators. *IEEE Transactions on Automatic Control*, 41(10):1527–1530, 1996.

[125] Hao Zhang, Xue Zhou, Zhuping Wang, Huaicheng Yan, and Jian Sun. Adaptive consensus-based distributed target tracking with dynamic cluster in sensor networks. *IEEE Transactions on Cybernetics*, 49(5):1580–1591, 2018.

[126] Ya Zhang, Feifei Li, and Yangyang Chen. Leader-following-based distributed kalman filtering in sensor networks with communication delay. *Journal of the Franklin Institute*, 354(16):7504–7520, 2017.

[127] Ya Zhang, Lucheng Sun, and Guoqiang Hu. Distributed consensus-based multi-target filtering and its application in formation-containment control. *IEEE Transactions on Control of Network Systems*, 07(01):503–515, 2020.

[128] Ya Zhang, Yaoyao Wen, Feifei Li, and Yangyang Chen. Distributed observer-based formation tracking control of multi-agent systems with multiple targets of unknown periodic inputs. *Unmanned Systems*, 7(01):15–23, 2019.

[129] Kemin Zhou and John Comstock Doyle. *Essentials Of Robust Control*, volume 104. Prentice Hall Upper Saddle River, NJ, 1998.

[130] Shanying Zhu, Cailian Chen, Wenshuang Li, Bo Yang, and Xinping Guan. Distributed optimal consensus filter for target tracking in heterogeneous sensor networks. *IEEE Transactions on Cybernetics*, 43(6):1963–1976, 2013.

[131] Lei Zou, Zidong Wang, Hongli Dong, and Qing-Long Han. Moving horizon estimation with multirate measurements and correlated noises. *International Journal of Robust and Nonlinear Control*, 30(17):7429–7445, 2020.

[132] Lei Zou, Zidong Wang, and Donghua Zhou. Moving horizon estimation with non-uniform sampling under component-based dynamic event-triggered transmission. *Automatica*, 120(109154), 2020.