2023

Learning and control for complex multiagent systems

Vrushabh S. Donge

LEARNING AND CONTROL FOR COMPLEX MULTIAGENT SYSTEMS

by

VRUSHABH S. DONGE

DISSERTATION

Submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

THE UNIVERSITY OF TEXAS AT ARLINGTON

Department of Electrical Engineering

December 2023

Committee:

Ali Davoudi, Supervising Professor

Frank L. Lewis, Supervising Professor

Ramtin Madani

Yijing Xie

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervising professor, Dr. Ali Davoudi, for constantly motivating me, sparing their valuable time, and giving me great guidance during my doctoral research. Dr. Ali Davoudi has always been so supportive and helpful whenever I run into difficulties. Thanks to his high standards and strict requirements, I can break through barriers and make progress in my research. I wish to express my profound gratitude to my Ph.D. co-supervisor, Dr. Frank L. Lewis, for his teaching, unwavering guidance, and steadfast commitment to our research throughout these years.

I would also like to extend my gratitude to Dr. Bosen Lian, who has co-authored several papers with me, for his invaluable contributions and insightful discussions that have significantly enhanced the quality of this work. I would thank my dissertation committee members, Dr. Ramtin Madani and Dr. Yijing Xie, for their interest in my research and for taking the time to be on my dissertation committee. I extend my heartfelt thanks to all the members of my research group for their invaluable assistance, unwavering support, and the wonderful experiences shared during my doctoral studies.

I am grateful to all the teachers who have guided me through my educational journey, both in India and the United States. I consider myself extremely fortunate to have been blessed by their wisdom and knowledge. I am also deeply thankful to my family for their sacrifices, unwavering encouragement, and patience throughout my academic pursuits. Furthermore, I extend my sincere thanks to several of my friends who have provided invaluable support and assistance at various points in my career.

The research content within this dissertation has, in part, received support from ARO grant W911NF-20-1-0132.

To my family and teachers.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

ABSTRACT

LEARNING AND CONTROL FOR COMPLEX MULTIAGENT SYSTEMS

VRUSHABH S. DONGE, Ph.D.

The University of Texas at Arlington, 2023

Supervising Professor: Ali Davoudi, Frank L. Lewis

Complex multiagent systems (MASs) are pervasive in various fields, from power system networks, and autonomous robotics to traffic management, where groups of agents interact to achieve collective objectives. Effective coordination and control of such systems pose significant challenges due to their inherent complexity and the need for adaptive, efficient strategies. This dissertation explores combining data-driven approaches, reinforcement learning (RL), and control theory to address these challenges. We present novel methodologies for learning and controlling complex MASs, emphasizing the development of adaptive algorithms that can autonomously adapt to dynamic environments, collaborate with other agents, and optimize system-wide performance. Our findings offer promising insights into creating intelligent MASs that can operate efficiently and effectively in diverse applications.

This thesis navigates the intricate realm of large-scale systems, focusing on MAS and complex nonlinear structures. It introduces innovative methodologies rooted in inverse RL to tackle challenges ranging from uncovering unknown cost functions to enabling data-efficient optimal control within MAS frameworks.

The research begins by unveiling an inverse RL algorithm designed for graphical apprentice games in MAS. This algorithm employs an inner-loop optimal control update and an outer-loop inverse optimal control (IOC) update as subproblems, where reward functions that the learner MAS finds are proven to be both stabilizing and non-unique. A simulation study of DC microgrid validates the effectiveness of this approach. Expanding the scope, the thesis explores the application of decomposition principles to discrete-time RL for optimal control in networked subsystems. Here, a model-free algorithm based on online behaviors is enhanced by employing dynamic mode decomposition (DMD) to handle larger networks, validated through consensus and power system networks.

Additionally, the work advances a data-efficient model-free RL algorithm using Koopman operators for complex nonlinear systems. This methodology lifts the nonlinear system into a linear model, deriving an off-policy Bellman equation that reduces data requirements for optimal control learning. Validation within power system excitation control demonstrates its efficacy. Furthermore, the thesis addresses reward-shaping challenges in large-scale MAS using inverse RL, proposing a scalable model-free algorithm. Leveraging DMD, this approach significantly diminishes data requirements while ensuring algorithm convergence, stability, and the non-uniqueness of state-reward weights. Validation in a large-scale consensus network confirms the method's efficacy through comparisons of data sizes and computational time for reward shaping.

Through these diverse methodologies and validations across various complex systems, this thesis not only contributes theoretical advancements but also offers practical solutions for managing, controlling, and shaping behaviors within intricate large-scale networked systems.

CHAPTER 1

INTRODUCTION

1.1 Motivation

For decades, control theorists have been drawn to studying large-scale, complex dynamic systems [94]. These systems are abundant in nature and span engineering and physical realms, encompassing a wide array of examples, including computer networks, robotic networks, sensor networks, electric power systems, social networks, transportation networks, and more.

With tremendous advancements in sensing technology, engineers are now more inclined to adopt data-driven control methods [132]. These approaches involve algorithms that combine adaptive control and optimal control, allowing control actions to be learned online. Traditionally, adaptive control and optimal control have been regarded as distinct design tools [98]. However, recent findings suggest that these two methodologies can be integrated using a machine-learning technique called reinforcement learning (RL) [130]. The necessity of defining the performance index in advance limits the scope of RL tools. In contrast, inverse optimal control (IOC) reconstructs the performance index based on an agent's demonstration [45]. Similarly, inverse RL [3] establishes an unknown optimal performance index using the agent's demonstration without requiring an understanding of system dynamics.

These RL algorithms are primarily constrained by the size of the systems. Typically, these designs rely on state trajectories and their combinations to acquire knowledge for optimal feedback controllers [59, 118, 119]. The quantity of states and control inputs influences the amount of data samples required to be stored during the exploration phase of the

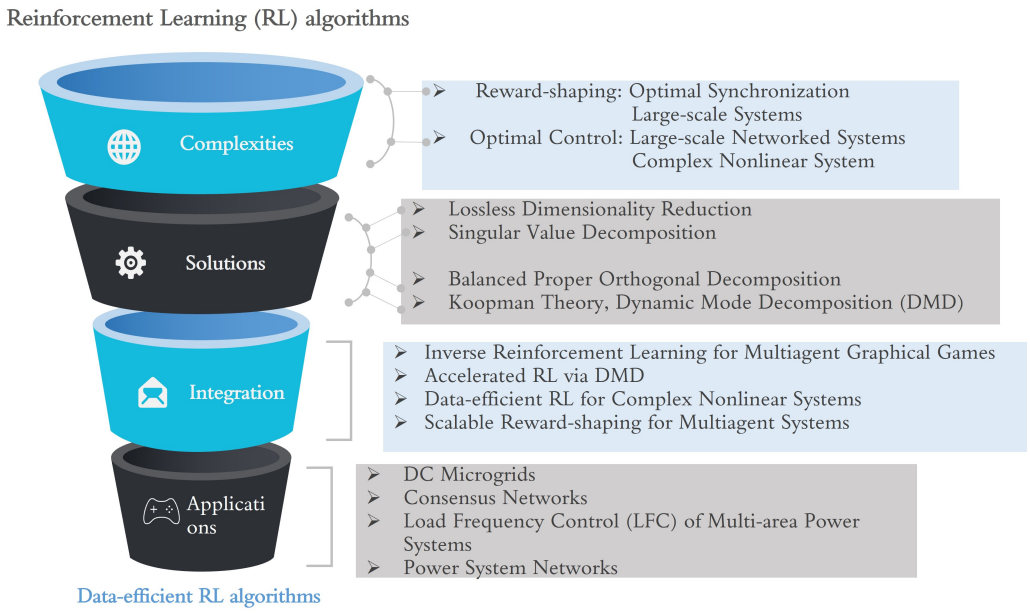


Figure 1.1. Research outline.

design process. Consequently, employing the complete-scale learning algorithms directly on physical systems with numerous states and control locations would result in prolonged exploration periods, alongside feedback control gains of higher dimensions. In this work, we introduce a scalable learning framework that integrates the concept of dimensionality reduction alongside the learning algorithms.

1.2 Research Outline

Figure 1.1 illustrates the research outline of the thesis. In an era where advancements in complex systems such as DC Microgrids, consensus Networks, and multi-area power systems are ever-increasing, the need for effective synchronization and control mechanisms becomes paramount. These systems, while integral to our modern infrastructure, pose intricate challenges due to their scale, interconnectivity, and nonlinear behavior. This thesis embarks on a journey into the heart of these complexities, aiming to unravel the intricacies inherent in large-scale networked systems. At its core lies the fusion of optimal control

theory with advanced computational methodologies. Techniques such as dynamic mode decomposition (DMD), Koopman theory, and lossless dimensionality reduction are brought together to offer novel solutions tailored to address the unique demands of these systems [111, 125]. Reward-shaping and optimal control take center stage, serving as the focal points of exploration.

Through the lens of decomposition techniques and inverse reinforcement learning, this thesis seeks to redefine how we approach the management of complex systems. These methodologies, though sophisticated, aim to provide not just theoretical frameworks but tangible, implementable solutions. Moreover, the research delves into the realm of data-efficient RL, a crucial frontier in addressing the challenges posed by complex nonlinear systems. The objective is clear to devise scalable, data-efficient strategies capable of handling the intricacies of multi-agent systems. Accelerated learning via DMD [34] and scalable reward-shaping strategies stand as promising avenues within this exploration.

1.3 Contributions

1. Chapter 2 introduces pivotal contributions focused on inverse RL algorithms within the realm of linear multi-agent graphical games featuring continuous-time differential system dynamics. Unlike prior works concentrated on single or multi-agent systems defined by Markov Decision Processes (MDPs), this work pioneers inverse RL for multi-agent graphical games. The model-based inverse RL algorithm delves into learning unknown reward functions by resolving both RL and IOC as subproblems. We achieve this by facilitating the estimation of expert reward weights, seeking Nash equilibrium, and continuously updating reward weight estimations based on expert trajectories. This lays the foundation for a model-free inverse RL algorithm designed to solve Graphical Apprentice Games without necessitating knowledge of

MAS dynamics, thus utilizing behavioral trajectory data for online implementation. The chapter culminates in a comprehensive analysis, delving into the stability, convergence, and quantification of non-uniqueness regarding state reward weights within the proposed inverse RL algorithms.

2. Chapter 3 is motivated by the integration of dynamic decomposition and RL to present a computationally feasible optimal control scheme tailored for large-scale networks. The chapter's standout contributions are outlined as follows: Firstly, it formulates the discrete-time LQR graphical problem for large-scale systems comprising linear subsystems, accommodating both coupled and decoupled dynamics and delineating stabilizing controllers. Secondly, it sheds light on the limitations of scaling associated with the model-free discrete-time RL algorithm when addressing large-scale LQR graphical problems. Subsequently, it introduces a computationally efficient discrete-time RL algorithm hinged on DMD. This novel approach diminishes data dimensions required for optimal control learning while conserving the original system's dynamic information. Lastly, the chapter substantiates the efficacy of the proposed algorithm through comprehensive theoretical and numerical analyses, underlining its computational efficiency and suitability for large-scale systems.
3. Chapter 4 introduces significant contributions aimed at easing the burdensome process of system modeling for complex nonlinear systems. The chapter unfolds as follows: Firstly, it pioneers an entirely data-driven model-based RL algorithm employing Koopman operators. This approach sidesteps the necessity for an exact model, setting it apart from existing RL methods [2, 55, 88, 155]. This characteristic renders it a more pragmatic choice for control tasks within intricate and uncertain environments. Secondly, leveraging Koopman eigenfunctions, the chapter develops a data-efficient model-free RL algorithm. This innovation truncates extensive datasets, enhancing data efficiency for optimal control learning in unknown

complex systems, a key advancement addressing longer learning times in previous works [22, 81, 85, 89, 99, 157]. Lastly, the chapter includes convergence analysis for the proposed data-efficient model-free algorithm and substantiates its efficacy through validation on a complex nonlinear power system.

4. Chapter 5 progresses through distinct stages and delivers notable contributions as follows: Initially, it formulates a reward-shaping problem within dynamically decoupled linear agents, aiming for the MAS under LQR control to replicate the behavior of the target MAS. Secondly, it harnesses DMD to achieve lossless dimensionality reduction, extracting dynamic modes from state measurements and constructing a projection matrix crucial for preserving essential dynamics. Subsequently, the chapter introduces a scalable model-free inverse RL algorithm, designed to mold an unknown reward function by solving optimal control and IOC as interconnected subtasks. Lastly, the proposed algorithm undergoes comprehensive analysis for stability and convergence, alongside quantifying the non-uniqueness of state reward weights, solidifying its theoretical foundations and applicability.

Ultimately, this research endeavors to contribute not only to academic discourse but to the practical realm, offering insights and tools that can potentially revolutionize how we manage and interact with intricate interconnected systems. The applications of these methodologies hold promise in reshaping the landscape of large-scale networked systems, potentially paving the way for more efficient, resilient, and adaptable infrastructures in the future. The publications stemming from this dissertation are listed below:

1. V. S. Donge, B. Lian, F. L. Lewis and A. Davoudi, "Multiagent Graphical Games With Inverse Reinforcement Learning," in IEEE Trans. Control Netw. Syst., vol. 10, no. 2, pp. 841-852, June 2023, doi:10.1109/TCNS.2022.3210856.

2. V. S. Donge, B. Lian, F. L. Lewis and A. Davoudi, ”Accelerated Reinforcement Learning Via Dynamic Mode Decomposition,” in IEEE Trans. Control Netw. Syst., doi:10.1109/TCNS.2023.3259060.
3. V. S. Donge, B. Lian, F. L. Lewis and A. Davoudi, ”Data-Efficient Reinforcement Learning for Complex Nonlinear Systems,” in IEEE Trans. Cybern., doi:10.1109/TCYB.2023.3324601.
4. V. S. Donge, B. Lian, F. L. Lewis, and A. Davoudi, “Efficient Reward Shaping for Multiagent Systems,” submitted to IEEE Trans. Control Netw. Syst., 2023.

Chapter 2

Multiagent Graphical Games with Inverse Reinforcement Learning *

*This chapter, titled 'Multiagent Graphical Games with Inverse Reinforcement Learning,' was originally published in IEEE Transactions on Control of Network Systems, vol. 10, no. 2, pp. 841-852, June 2023, doi: 10.1109/TCNS.2022.3210856. It has been reprinted with permission from all co-authors and is used with permission from IEEE without any revisions. This version represents the authors' accepted manuscript.

CHAPTER 2

Multi-agent Graphical Games with Inverse Reinforcement Learning

2.1 Introduction

Optimal control techniques can be designed to achieve synchronization among agents of a multi-agent system (MAS) [98]. Conventional optimal control [15] assumes the knowledge of system dynamics, whereas reinforcement learning (RL) [130] provides optimal solutions using behavior data without knowing system dynamics. However, the requirement to predefine the performance index restricts the applicability of optimal control and RL tools. Alternatively, inverse optimal control (IOC) uses an agent's demonstration to rebuild the performance index [45]. Likewise, inverse RL [3] builds an unknown optimal performance index given the agent's demonstration without perceiving system dynamics.

Observing expert demonstrations to emulate an agent to the expert's preferences is known as imitation learning. Slight changes in the learning area might lead to useless learned policies, but they generally would not affect the learned reward functions. This makes the reward function naturally more transferable [104]. Inverse RL is essentially one approach to implementing imitation learning. In inverse RL, the learner agent, or the *apprentice*, discovers the unknown reward function by observing the expert's demonstration generated by an optimal policy and conforming to common trajectories [3, 4, 12, 19, 28, 53, 113, 126, 131]. Similarities between inverse RL and imitation learning are discussed in [108]. [104] and [108] obtain a reward function from observed optimal behavior using inverse RL for Markov decision processes (MDP). [150] presents inverse RL to acquire an unknown reward function for tracking control with the support of IOC. The inverse RL-based actor-critic framework in [83] addresses adversarial attacks in a single-agent environ-

ment. The inverse RL technique remains largely unexplored in the optimal synchronization of multi-agent graphical games.

For all agents to agree on a certain quantity of interest, [27, 98] use optimal control with distributed dynamics produced by a single reward function in terms of the local synchronization error. In [95,112], off-policy RL algorithms solve the synchronization issue of MAS using graphical games. Defining the cooperative reward function over iterative learnings for the RL becomes problematic. RL techniques assume that the performance indices of agents are known upfront. In the apprentice learning task, the performance indices of the expert MAS could be hidden from the learner [3,4,113,131]. Herein, we employ graphical games [98] that blend agent dynamics via a communication network for synchronization, and integrate it with apprentice learning to obtain unknown cooperative reward functions of the learner MAS using inverse RL. This apprentice learning synchronizes the learner MAS with the expert MAS.

The majority of existing work on inverse RL does not have to acknowledge the strict stability of learner or expert system dynamics. Learner stability must be assured during the retrieval of expert performance indices. Given the system dynamics and demonstrations, agents can generate a reward function using the IOC approach [45], which also ensures their stability by choosing the correct policy for a stabilizing reward. [60] and [106] generate a state reward function for deterministic continuous-time nonlinear systems. [37] uses IOC to recover unique value functions, cost functions, and control policies of the expert within the formulation of linearly solvable MDPs. [121, 139] provide a unique IOC technique for discrete-time nonlinear system stabilization and path tracking which eliminates the need to evaluate the Hamilton-Jacobi-Bellman (HJB) equation in uncertain complex networks. Bayes learning [162] is similar to IOC in the sense that it also finds uncertain performance indices of an agent using probability sampling methods [20]. Since IOC em-

ploys deterministic models, it does not need to compute the expectation over trajectories as Bayes learning would require.

This paper defines a Graphical Apprentice Games for continuous-time linear MAS. To solve these games online, the inverse RL algorithm infers the unknown reward functions of the learner based on an expert's behavioral trajectory. Figure 2.1 illustrates the proposed method for achieving optimal apprentice synchronization using inverse RL with optimal control to extract optimal policy and IOC to revise state reward weight. Salient contributions of this paper are:

1. We explore inverse RL algorithms for linear multi-agent graphical games with continuous-time differential system dynamics. This is in contrast to the inverse RL work in [47, 102, 104, 108], where single- or multi-agent systems are defined by MDPs. To the best of our knowledge, inverse RL has not been studied for multi-agent graphical games.
2. The model-based inverse RL algorithm learns an unknown reward function for our Graphical Apprentice Games by solving both RL and IOC as subproblems. First, the learner MAS finds the corresponding Nash equilibrium given the current estimation of the expert's reward weight. Then, the learner MAS updates its current estimated reward weight by monitoring the expert's trajectories. Moreover, this algorithm enables model-free formulation for IOC to reconstruct the reward function.
3. We further propose a model-free inverse RL algorithm to solve the Graphical Apprentice Games without using knowledge of MAS dynamics. This algorithm is then implemented online using behavioral trajectory data.
4. Stability and convergence of proposed approaches are analyzed. Furthermore, the non-uniqueness of state reward weights is quantified in our inverse RL algorithm.

This paper is compiled as follows: Section II introduces the Graphical Apprentice Games followed by formulations of learner and expert MAS dynamics. Section III gives a

model-based inverse RL algorithm. Section IV presents a model-free inverse RL algorithm with online implementation. Section V presents simulation studies. Section VI concludes this work.

Notations: $\mathbb{R}^n$ is the n-dimensional Euclidean space. $\otimes$ stands for the Kronecker product. I_n indicates the n-dimensional identity matrix. $\|\cdot\|$ defines the 2-norm of a vector or a matrix. For matrix $P_i \cong [P_1, P_2, \ldots, P_n] \in \mathbb{R}^{n\times n}$,

$$\text{vec}(P) = [P_{11}\ P_{12}\ \ldots\ P_{1n}\ P_{22}\ \ldots\ P_{(n-1)n}\ P_{nn}]^T. \tag{2.1}$$

For a vector $x = [x_1,\ x_2,\ \ldots,\ x_n]^T \in \mathbb{R}^n$, vecv$(x \otimes x)$ denotes the vector $[x_1^2,\ 2x_1x_2,\ \ldots,\ 2x_1x_n,\ x_2^2,\ 2x_2x_3,\ \ldots,\ 2x_{n-1}x_n,\ x_n^2]^T$.

2.2 Problem Formulation

This section defines Graphical Apprentice Games, and formulates expert and learner MAS.

2.2.1 Graphical Apprentice Games

This paper proposes a framework for inverse RL of cooperative graphical games where the learner MAS aims to attain optimal synchronization by imitating the demonstrated behavior of the expert MAS. The graphical games centered on apprentice learning are called *Graphical Apprentice Games*. Therein, the learner seeks to imitate expert behavioral trajectories without accessing optimal policies. It infers unknown reward functions of expert MAS and obtains the same optimal control policies.

Graph topology for Graphical Apprentice Game: Consider a set of agents $\mathcal{N} \cong (1, 2, \ldots, N)$ in the communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The graph is identical between the expert MAS and the learner MAS. The vertex of an individual agent is $v_i \in \mathcal{V}$, $\forall i \in \mathcal{N}$. There are edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ between the two vertices with the connectivity weight $e_{ij} > 0$

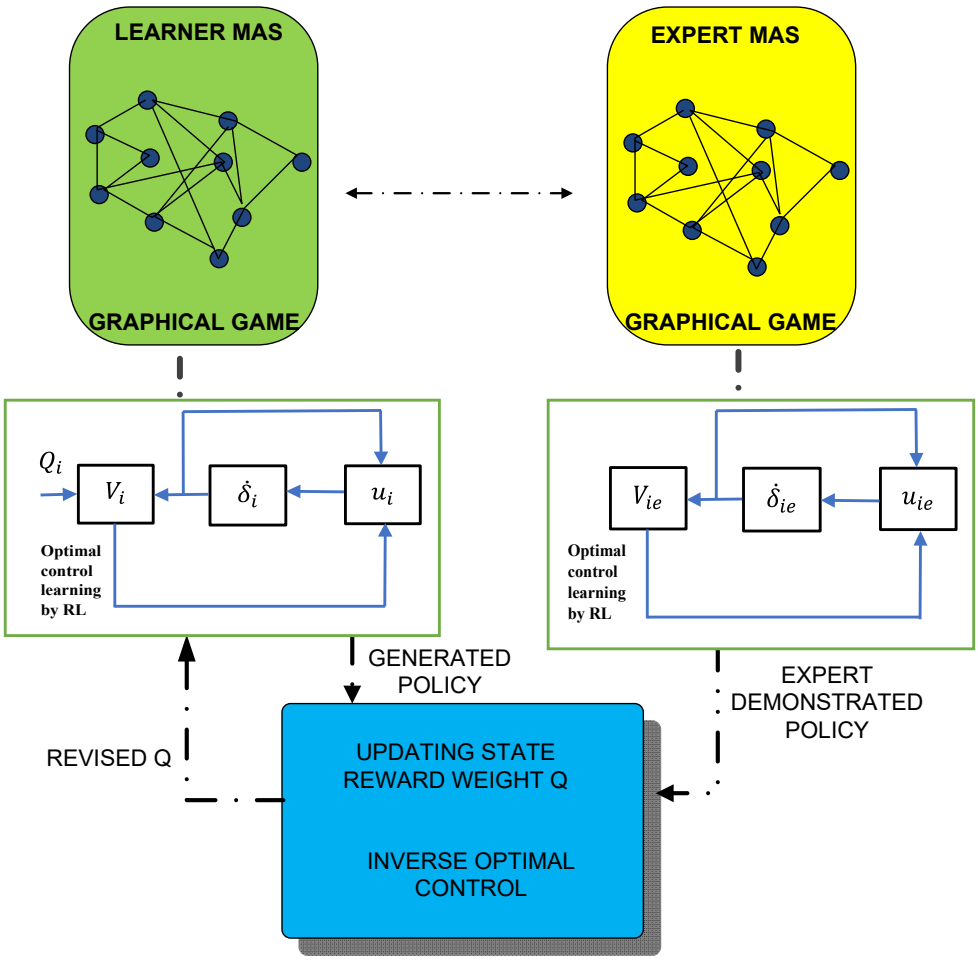


Figure 2.1. Inverse RL framework for solving Graphical Apprentice Game between expert and learner MAS: 1) Expert has optimal synchronization trajectories for states and control inputs by optimal control and, 2) Given expert's demonstrated trajectories, the learner MAS finds its unknown cost functions using an inner-loop optimal control update and an outer-loop IOC update..

if $(v_j, v_i) \in \mathscr{E}$; Otherwise, $e_{ij} = 0$. The set of vertex v_i neighbors is $\mathscr{N}_i \cong \{v_j : e_{ij} > 0\}$. Assuming no self-loops in the graph, i.e., $e_{ii} = 0$. The graph adjacency matrix is specified as $\mathscr{A} = [e_{ij}]$. The graph degree matrix is $\mathscr{D} = diag_i\{d_i\}$, where weighted in-degree of vertex i is $d_i = \sum_{j=1}^{\mathscr{N}} e_{ij}$. Laplacian matrix $\mathscr{L}$ is defined as $(\mathscr{D} - \mathscr{A})$ [146]. The progression of edges $\mathscr{E}$ through vertices $(v_{i_{s-1}}, v_{i_s}) \in \mathscr{E}$ for $s \in (2, \ldots, j)$ constitutes a directed path originating

from v_{i_1} to v_{i_j}. The graph is considered to be strongly connected if there is a directed path between every pair of vertices v_i and v_j.

2.2.1.1 Expert MAS

Consider an expert MAS with linear dynamics of each agent i, $i \in \mathcal{N}$ as

$$\dot{x}_{ie} = Ax_{ie} + Bu_{ie}, \tag{2.2}$$

where $x_{ie} \in \mathbb{R}^n$ and $u_{ie} \in \mathbb{R}^p$ denote state and control input of the expert agent i, respectively. The expert MAS has identical system matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times p}$. The pair (A,B) is assumed stabilizable. The local synchronization error of each agent i is defined as $\delta_{ie} = \sum_{j\in\mathcal{N}_i} e_{ij}(x_{ie} - x_{je})$, and its error dynamics is given by

$$\begin{aligned}\dot{\delta}_{ie} &= \sum_{j\in\mathcal{N}_i} e_{ij}(\dot{x}_{ie} - \dot{x}_{je}) \\ &= A\delta_{ie} + d_i Bu_{ie} - \sum_{j\in\mathcal{N}_i} e_{ij}Bu_{je}.\end{aligned} \tag{2.3}$$

Define the cooperative cost function of each expert agent i as

$$V_{ie}\big(\delta_{ie}(t_0), u_{ie}, u_{-ie}\big) = \int_{t_0}^{\infty} \big(\delta_{ie}^{\mathrm{T}} Q_{ie}\delta_{ie} + u_{ie}^{\mathrm{T}} R_{ie} u_{ie}\big)dt, \tag{2.4}$$

where $Q_{ie} = Q_{ie}^{\mathrm{T}} \in \mathbb{R}^{n\times n} \geq 0$ and $R_{ie} = R_{ie}^{\mathrm{T}} \in \mathbb{R}^{p\times p} > 0$ are the state reward weight and control input weight, respectively. $u_{-ie} \cong \{u_{je} | j \in \mathcal{N}_i\}$ is the set of control inputs for the neighbors of the expert agent i.

Definition 1. Nash equilibrium: Expert MAS has an $\mathcal{N}$-tuple policy $\{u_{1e}^*, u_{2e}^*, \ldots, u_{Ne}^*\}$ promising a global Nash equilibrium solution, in the sense that

$$V_{ie}^* \cong V_{ie}\big(\delta_{ie}(t_0), u_{ie}^*, u_{-ie}^*\big) \leq V_{ie}\big(\delta_{ie}(t_0), u_{ie}, u_{-ie}^*\big). \tag{2.5}$$

The optimal value function of each expert agent i has the quadratic form

$$\begin{aligned}V_{ie}^*\big(\delta_{ie}(t)\big) &= \min_{u_{ie}} \int_t^{\infty} \big(\delta_{ie}^{\mathrm{T}} Q_{ie}\delta_{ie} + u_{ie}^{\mathrm{T}} R_{ie} u_{ie}\big)d\tau \\ &= \delta_{ie}^{\mathrm{T}} P_{ie}\delta_{ie},\end{aligned} \tag{2.6}$$

where $P_{ie} = P_{ie}^T \in \mathbb{R}^{n\times n} > 0$. Given the value function $V_{ie}^*(\delta_{ie})$, and using the optimal control of cooperative multi-agent systems [98], the expert MAS has the optimal input u_{ie}^* as

$$u_{ie}^* = -d_i R_{ie}^{-1} B^{\mathrm{T}} P_{ie} \delta_{ie} = -K_{ie}\delta_{ie}, \tag{2.7}$$

where $K_{ie} \cong d_i R_{ie}^{-1} B^{\mathrm{T}} P_{ie}$, and P_{ie} satisfies the coupled HJB equation of the expert MAS as

$$\begin{aligned} 0 =& \delta_{ie}^{\mathrm{T}} Q_{ie} \delta_{ie} + d_i^2 \delta_{ie}^{\mathrm{T}} P_{ie} B R_{ie}^{-1} B^{\mathrm{T}} P_{ie} \delta_{ie} + 2\delta_{ie}^{\mathrm{T}} P_{ie} A \delta_{ie} \\ &+ 2d_i \delta_{ie}^{\mathrm{T}} P_{ie} \sum_{j\in N_i} e_{ij} B R_{je}^{-1} B^{\mathrm{T}} P_{je} \delta_{je}. \end{aligned} \tag{2.8}$$

2.2.1.2 Learner MAS

Consider the learner MAS with each agent dynamics are

$$\dot{x}_i = Ax_i + Bu_i, \tag{2.9}$$

where $x_i(t) \in \mathbb{R}^n$ and $u_i \in \mathbb{R}^p$ denote states and control inputs of each learner agent i, respectively. Note that the system dynamics A and B are the same as in (1).

The local synchronization error of each agent i is $\delta_i = \sum_{j\in\mathcal{N}_i} e_{ij}(x_i - x_j)$. The error dynamics for learner MAS is

$$\dot{\delta}_i = A\delta_i + d_i B u_i - \sum_{j\in\mathcal{N}_i} e_{ij} B u_j. \tag{2.10}$$

The cooperative cost function of each learner agent i is

$$V_i\big(\delta_i(t_0), u_i, u_{-i}\big) = \int_{t_0}^{\infty} \big(\delta_i^{\mathrm{T}} Q_i \delta_i + u_i^{\mathrm{T}} R_i u_i\big) dt, \tag{2.11}$$

where $Q_i = Q_i^{\mathrm{T}} \in \mathbb{R}^{n\times n} \geq 0$, $R_i = R_i^{\mathrm{T}} \in \mathbb{R}^{p\times p} > 0$, and $u_{-i} = \{u_j | j \in \mathcal{N}_i\}$.

Definition 2. (**Equivalent state reward weight**) Given system dynamics A and B, the weights Q_{ie} and R_{ie} in (2.8), define u_{ie}^* in (2.7). Arbitrarily selecting $R_i > 0$, one can find a $\bar{Q}_i$, $\forall i \in \mathcal{N}$ in learner HJB equation, such that there are $\mathcal{N}$-tuple V_i^∞ in the learner MAS that solve $u_i = u_{ie}^*$. Then, one has $\bar{Q}_i$ is equivalent to Q_{ie}.

Assumption 1. Communication graphs in both learner MAS and expert MAS are strongly connected.

Assumption 2. The learner MAS knows its own state reward weight and control weight, Q_i and R_i, $\forall i \in \mathcal{N}$, but does not know the expert's Q_{ie} and R_{ie}. Also, R_i can differ from R_{ie}.

Assumption 3. Each learner agent i can observe expert agent i's trajectories of u_{ie}^*, where $i \in \mathcal{N}$. Notice that expert and learner MASs have separate graphical games, and their agents are not neighbors on the same graph.

Graphical Apprentice Game Problem: Consider Assumptions 1, 2, and 3. Provided the expert MAS optimal control input u_{ie}^*, each learner agent i seeks to learn an equivalent state reward weight $\bar{Q}_i$ to Q_{ie} that satisfies (2.8), such that it presents the expert's behavior, i.e., $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$ with the expert agent i's feedback control gain K_{ie}.

Remark 1. Note that we analyze local synchronization error dynamics in (2.3) and (2.10) for the synchronization problem of MAS instead of (2.2) and (2.9). It is evident from (2.3) and (2.10) that the local synchronization error of each agent i is driven by the control input of agent i and its neighboring agents. The cooperative cost function in (2.4) and (2.11) is considered for optimal synchronization. In addition, inverse RL is developed for (2.3) and (2.10) such that $(\delta_i = \delta_{ie}) \to 0$, i.e., $x_i \to x_{ie}$.

2.3 Model-based Inverse RL Algorithm

This section builds a model-based inverse RL algorithm to solve the Graphical Apprentice Game Problem. The inverse RL algorithm is a two-loop iterative process. In the inner loop, given the current estimation Q_i of the expert's reward weight Q_{ie} in (2.8), the learner MAS finds the corresponding Nash equilibrium using RL-based optimal control learning. In the outer loop, the learner MAS updates the current Q_i towards the equiva-

lent weight Q_{ie} by monitoring the expert MAS behavioral trajectories u_{ie}^* using IOC. The learner MAS ultimately finds the expert's feedback control gain K_{ie} and the expert's behavior (x_{ie}, u_{ie}^*) as both inner and outer loops are repeated.

2.3.1 Optimal Control Learning by the Learner MAS

To find the optimal value function $V_i^*(\delta_i)$

$$\begin{aligned} V_i^*\big(\delta_i(t)\big) &= \min_{u_i} \int_t^\infty \big(\delta_i^{\mathrm{T}} Q_i \delta_i + u_i^{\mathrm{T}} R_i u_i\big) d\tau \\ &= \delta_i^{\mathrm{T}} P_i \delta_i, \end{aligned} \tag{2.12}$$

where $P_i = P_i^T \in \mathbb{R}^{n\times n} > 0$, one refers to the optimal control technique in [15, 95], and has the optimal control input of learner agent i as

$$u_i^* = -d_i R_i^{-1} B^{\mathrm{T}} P_i \delta_i = -K_i \delta_i, \tag{2.13}$$

where $K_i \cong d_i R_i^{-1} B^{\mathrm{T}} P_i$, and P_i satisfies the coupled HJB equation of the learner MAS

$$\begin{aligned} 0 =& \delta_i^{\mathrm{T}} Q_i \delta_i + {d_i}^2 \delta_i^{\mathrm{T}} P_i B R_i^{-1} B^{\mathrm{T}} P_i \delta_i \\ &+ 2\delta_i^T P_i A \delta_i + 2 d_i \delta_i^{\mathrm{T}} P_i \sum_{j\in N_i} e_{ij} B R_j^{-1} B^{\mathrm{T}} P_j \delta_j. \end{aligned} \tag{2.14}$$

The learner MAS solves (2.13) and (2.14) for each learner agent i using RL-based optimal control learning to obtain a converged optimal solution set (V_i^*, u_i^*), given a current estimate Q_i of Q_{ie}. The policy iteration in [16] can solve this optimal control learning problem as will be presented with (2.16) and (2.17) in Algorithm 1.

Remark 2. In some cases, experts might not be able to send their rewards directly to learners since: 1) experts might be unaware of learners' presence, 2) there could be competition amongst groups, and 3) experts could be even unaware of their own quantitative rewards.

2.3.2 Inverse Optimal Control to Learn the Reward Weight

Learner state reward weights are revised using IOC given the demonstration of expert u_{ie}^*. The learner MAS tries to infer $\bar{Q}_i$ satisfying (2.14) such that it defines the expert's behavior. That is, $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$. Based on IOC [45], the state reward weight $\bar{Q}_i$ is revised towards Q_i^∞ by

$$\delta_i^{\mathrm{T}} \bar{Q}_i \delta_i = u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^* - 2\delta_i^T P_i \left(A\delta_i + d_i B u_i^* - \sum_{j \in N_i} e_{ij} B u_j^* \right). \tag{2.15}$$

Repeat these loops iteratively until Q_i converges to $\bar{Q}_i$. Consequently, (2.14) defines the same behavioral trajectories as (2.8), i.e., (x_{ie}, u_{ie}^*). In Algorithm 1, the iterative form of (2.15) is presented as (2.18). The model-based inverse RL Algorithm 1 for the graphical apprentice game can now be presented.

Remark 3. Note that Algorithm 1 is developed for homogeneous MAS graphical games similar to the works in [98, 112]. In a heterogeneous case, where A and B are not identical for each agent i, Algorithm 1 may not apply. Experts' error dynamics in (2.3) becomes a function of δ_{ie} and x_{je}. Similarly, (2.10) becomes a function of δ_i and x_j for the learner. This indicates that the distributed control u_{ie} and u_i in (2.7), (2.13) should be derived based on (δ_{ie}, x_{je}) and (δ_i, x_j), respectively, instead of just δ_{ie} and δ_i.

Assumption 4. The value of e_i in the Algorithm 1 is sufficiently small so as not to affect the execution of outer loops [150]. Based on [141], we select the lower bound $e_i > 0$ satisfying $e_i \leq \beta T \prod_{i=1}^{n} |\lambda_i|$, where $\beta > 0$ is an scaling factor, T is the sampling time, and λ_i denotes eigenvalues of the closed-loop dynamics of (2.9).

Remark 4. The contributions of Algorithm 1 are three-folds. First, it solves homogeneous Graphical Apprentice Games defined by differential dynamic equations in contrast to the multi-agent inverse RL for MDPs. Second, Algorithm 1 illustrates that optimal control and IOC must be solved as subproblems in our inverse RL. Third, in contrast with IOC

Algorithm 1 Model-based inverse RL Algorithm for Graphical Apprentice Game

1 **Initialization:** For each learner agent $i \in \mathcal{N}$, select initial $Q_i^0 \geq 0$, $R_i > 0$, and stabilizing control input u_i^{00}. Set $h = 0$ and small thresholds e_i and ε_i.

2 **Outer h iteration loop using IOC**

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

4 Policy evaluation for solving P_i^{hk}

$$\begin{aligned} 0 =& \delta_i{}^{\mathrm{T}} Q_i^h \delta_i + (u_i^{hk})^{\mathrm{T}} R_i u_i^{hk} \\ &+ 2\delta_i^{\mathrm{T}} P_i^{hk} \left(A\delta_i + d_i B u_i^{hk} - \sum_{j \in N_i} e_{ij} B u_j^{hk} \right); \end{aligned} \tag{2.16}$$

5 Policy improvement for solving $u_i^{h(k+1)}$

$$u_i^{h(k+1)} = -d_i {R_i}^{-1} B^{\mathrm{T}} P_i^{hk} \delta_i; \tag{2.17}$$

6 **Stop if** $\|P_i^{hk} - P_i^{h(k-1)}\| \leq e_i$, then set $P_i^h = P_i^{hk}$, $u_i^h = u_i^{hk}$, and go to step 7, if not, set $k \leftarrow k+1$ and go to step 4;

7 **Outer h iteration loop using IOC:** Q_i^{h+1} update using expert optimal control inputs u_{ie}^*

$$\begin{aligned} \delta_i{}^T Q_i^{h+1} \delta_i =& u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h \\ &- 2\delta_i^{\mathrm{T}} P_i^h \left(A\delta_i + d_i B u_i^h - \sum_{j \in N_i} e_{ij} B u_j^h \right); \end{aligned} \tag{2.18}$$

8 **Stop if** $\|Q_i^{h+1} - Q_i^h\| \leq \varepsilon_i$. Otherwise, set $u_i^{(h+1)0} = u_i^h$, $h \leftarrow h+1$, and go to Step 3.

studies [45] that had required system dynamics, we can produce a model-free algorithm to estimate reward function towards expert MAS.

2.3.3 Convergence and Stability Analysis

Herein, we show the convergence of the proposed Algorithm 1 and the stability of the learner MAS.

Theorem 1. ***(Convergence of Algorithm 1)*** *Given Assumptions 1, 2, and 3, consider the Algorithm* 1 *for solving the Graphical Apprentice Game Problem. Select initial* $Q_i^0 \geq 0$ *and* $R_i > 0$*. Then, the learner agent* i *converges in a finite number of iterations* h*, i.e.,* $(x_i, u_i^h) \rightarrow (x_{ie}, u_{ie}^*)$*, where* u_i^h *and* u_{ie}^* *are given by* (2.17) *and* (2.7)*, respectively. Furthermore,* Q_i^h *converges to* $\bar{Q}_i$*, where* $\bar{Q}_i$ *is equivalent to* Q_{ie}^**.*

Proof. We analyze the convergence for two loops in Algorithm 1. First, we show the convergence of inner iteration loops, the policy iteration learning, which includes steps 4 and 5 of Algorithm 1. As exhibited in [95, 109], for a fixed outer iteration h, if $Q_i^h \geq 0$ and initial stabilizing control inputs u_i^{h0} for each learner agent $i, i \in \mathcal{N}$, the inner iteration loop gives the converging solution set (u_i^h, V_i^h) as $k \rightarrow \infty$ where V_i^h is optimal. Note that the stabilizing control input u_i^{h0} can be obtained from step 8.

Second, we prove the convergence of outer loops for a finite number of outer iterations h based on the convergence of inner loops. When inner iteration loops converge given the current Q_i^h, $\forall i \in \mathcal{N}$, step 4 gives

$$\begin{aligned} 0 =& \delta_i^{\mathrm{T}} Q_i^h \delta_i + (u_i^h)^{\mathrm{T}} R_i u_i^h \\ &+ 2\delta_i^{\mathrm{T}} P_i^h \left(A\delta_i + d_i B u_i^h - \sum_{j \in N_i} e_{ij} B u_j^h \right). \end{aligned} \tag{2.19}$$

The state reward weight Q_i^{h+1} of each learner agent i becomes

$$\begin{aligned}\delta_i^{\mathrm{T}} Q_i^{h+1} \delta_i =& u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h \\ & - 2\delta_i^{\mathrm{T}} P_i^h \left(A\delta_i + d_i B u_i^h - \sum_{j\in N_i} e_{ij} B u_j^h \right),\end{aligned} \tag{2.20}$$

which, combined with (2.19), yields

$$\begin{aligned}\delta_i{}^T Q_i^{h+1} \delta_i =& u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h + \delta_i^{\mathrm{T}} Q_i^h \delta_i + (u_i^h)^{\mathrm{T}} R_i u_i^h \\ =& (u_{ie}^* - u_i^h)^{\mathrm{T}} R_i (u_{ie}^* - u_i^h) + \delta_i^{\mathrm{T}} Q_i^h \delta_i.\end{aligned} \tag{2.21}$$

Given initial $Q_i^0 \geq 0$, one obtains $0 \leq Q_i^0 \leq Q_i^1 \leq \cdots \leq Q_i^h \leq Q_i^{h+1}$, and $\left\|Q_i^h\right\|$ increases for each outer iteration loop $h = 1, 2\ldots$ of each learner agent i, $i \in \mathcal{N}$.

Let $\Theta_i^h(\delta) \cong (u_{ie}^* - u_i^h)^{\mathrm{T}} R_i (u_{ie}^* - u_i^h)$. It can be inferred from (2.21) that

$$\delta_i{}^T Q_i^h \delta_i = \Theta_i^{h-1}(\delta) + \cdots + \Theta_i^0(\delta) + \delta_i^{\mathrm{T}} Q_i^0 \delta_i. \tag{2.22}$$

Assume that there are infinitely many solutions of $\bar{Q}_i$, $\forall i \in \mathcal{N}$, this is proved in Theorem 2. It is well-known that, given a Q_i^h, u_i^h is uniquely defined. Therefore, there exists, at a minimum, one $\bar{Q}_i$ such that $Q_i^0 \leq \bar{Q}_i$ is satisfied, and Q_i^h can approach $\bar{Q}_i$. If Q_i^h reaches $\bar{Q}_i$, then u_i^h approaches u_{ie}^* and $\left\|\Theta_i^h\right\|$ is decreasing, i.e., $\left\|\Theta_i^h\right\| \geq \left\|\Theta_i^{h+1}\right\|$. If $Q_i^h \to \bar{Q}_i$ such that $\|Q_i^h - \bar{Q}_i\| \leq \varepsilon_i$, i.e., there exists a small threshold ε_i. Then, $u_i^h \to u_{ie}^*$, with small threshold α_i such that $\left\|u_{ie}^* - u_i^h\right\| \leq \alpha_i$. The threshold ε_i is used to terminate Algorithm 1 at h. Thus, Algorithm 1 is stopped at h. It follows from (2.21) and (2.22) that one has

$$\begin{aligned}\left\|\delta_i{}^T \bar{Q}_i \delta_i\right\| =& \left\|\Theta_i^h(\delta) + \Theta_i^{h-1}(\delta) + \cdots + \Theta_i^0(\delta) + \delta_i^{\mathrm{T}} Q_i^0 \delta_i\right\| \\ \geq& (h+1)\alpha_i^2 \Lambda_{min}(R_i) - \left\|\delta_i^2\right\| \left\|Q_i^0\right\|.\end{aligned} \tag{2.23}$$

This yields

$$\bar{h} \approx \frac{\left\|\delta_i^2\right\| \left\|\bar{Q}_i\right\| + \left\|\delta_i^2\right\| \left\|Q_i^0\right\|}{\alpha_i^2 \Lambda_{min}(R_i)} \geq h. \tag{2.24}$$

Thus, Algorithm 1 converges after a finite number of $\bar{h}$ outer iteration loops.

Since the control input u_i^h converges to u_{ie}^*, and the system dynamics of both learner and expert are the same, one can conclude that $x_i = x_{ie}$ using the converged policy u_i^h. Therefore, the converging behavior $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$ and $(Q_1^h, Q_2^h \ldots, Q_N^h) \to (\bar{Q}_1, \bar{Q}_2 \ldots, \bar{Q}_N)$ can be achieved simultaneously. The learner's HJB equation can be written as

$$\begin{aligned} 0 =& \delta_{ie}^{\mathrm{T}} \bar{Q}_i \delta_{ie} + {d_i}^2 \delta_{ie}^{\mathrm{T}} P_i^{\infty} B R_i^{-1} B^{\mathrm{T}} P_i^{\infty} \delta_{ie} \\ &+ 2\delta_{ie}^{\mathrm{T}} P_i^{\infty} A \delta_{ie} + 2 d_i \delta_{ie}^{\mathrm{T}} P_i^{\infty} \sum_{j \in N_i} e_{ij} B R_j^{-1} B^{\mathrm{T}} P_j^{\infty} \delta_{je}. \end{aligned} \tag{2.25}$$

This gives $V_i^{\infty}(\delta_{ie})$ related to $\bar{Q}_i$, $\forall i \in \mathcal{N}$, where u_i^* is

$$u_i^* = -d_i R_{ie}^{-1} B^{\mathrm{T}} P_{ie}^{\infty} \delta_{ie}, \forall i \in \mathcal{N}. \tag{2.26}$$

It is seen from (2.26) that the feedback control gain K_i^{∞} are the same as K_{ie}. One cannot guarantee $\bar{Q}_i$ is the same as Q_{ie}, $\forall i \in \mathcal{N}$. However, if $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$ holds, then, there exists $V_i^{\infty} = V_{ie}^*$, $\forall i \in \mathcal{N}$ according to Definition 2 of equivalent state reward weight. This finishes the proof. □

Theorem 2. ***(Non-uniqueness analysis)*** *Suppose that using Algorithm 1, one has $P_i^h \to P_i^{\infty}$ and $Q_i^h \to \bar{Q}_i$. Then, $\bar{Q}_i$ satisfies*

$$\begin{aligned} &\delta_{ie}^T (\bar{Q}_i - Q_{ie}) \delta_{ie} \\ &= d_i^2 B (R_i^{-1} - R_{ie}^{-1}) B^T P_i^{\infty} \delta_i (P_i^{\infty} \delta_i)^T + 2 (P_{ie}^* \delta_{ie} - P_i^{\infty} \delta_i) \\ &\quad \times \left(A \delta_{ie} - d_i \sum_{j \in N_i} e_{ij} \big(B R_{je}^{-1} B^T P_{je} \delta_{je} - B {R_j}^{-1} B^T P_j^{\infty} \delta_j \big) \right), \end{aligned} \tag{2.27}$$

with P_{ie}^* uniquely solved by (2.8)*, and P_i^{∞} satisfying*

$$B^T P_i^{\infty} \delta_i^{\infty} = R_i R_{ie}^{-1} B^T P_{ie}^* \delta_{ie}, \tag{2.28}$$

which implies that $\bar{Q}_i$ might not be unique.

Proof. The learner agent i achieves the convergence $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$ and $(Q_1^h, Q_2^h \ldots, Q_N^h) \rightarrow (\bar{Q}_1, \bar{Q}_2 \ldots, \bar{Q}_N)$ as shown in Theorem 1. At the same time, one has P_{ie} associated with $\bar{Q}_i$. Then,

$$B^{\mathrm{T}} P_i^{\infty} = R_i R_{ie}^{-1} B^{\mathrm{T}} P_{ie}. \tag{2.29}$$

Then, subtracting (A.6) from (2.8) yields

$$\begin{aligned} &\delta_{ie}^T (\bar{Q}_i - Q_{ie}) \delta_{ie} \\ &= d_i^2 B (R_i^{-1} - R_{ie}^{-1}) B^{\mathrm{T}} P_i^{\infty} \delta_i (P_i^{\infty} \delta_i)^{\mathrm{T}} + 2 (P_{ie}^* \delta_{ie} - P_i^{\infty} \delta_i) \\ &\quad \times \left(A \delta_{ie} - d_i \sum_{j \in N_i} e_{ij} \left(B R_{je}^{-1} B^{\mathrm{T}} P_{je} \delta_{je} - B R_j^{-1} B^{\mathrm{T}} P_j^{\infty} \delta_j \right) \right). \end{aligned} \tag{2.30}$$

Note that (2.29) is consistent and has a unique solution only when $\mathrm{rank}(B) = n$. But one cannot assure this. There are an infinite number of solutions for P_i^{∞} in (2.29) when $\mathrm{rank}(B) < n$. Moreover, R_i can be different from R_{ie}. Thus, $\bar{Q}_i - Q_{ie}$ will be non-zero, and there will be an infinite number of solutions of $\bar{Q}_i$ given an infinite number of solutions for P_i^{∞} in (2.29).

This finishes the proof. □

Theorem 3. ***(Stability analysis)*** *Given Assumptions 1, 2, and 3, select initial $Q_i^0 \geq 0$ and $R_i > 0$, $\forall i \in \mathcal{N}$. Use Algorithm 1 to solve the Graphical Apprentice Game Problem. Then, each learner agent i is asymptotically stable in the learning process.*

Proof. We prove the stability of the learner MAS at each iteration of both inner and outer iteration loops in Algorithm 1 as the outer loops decide whether inner loops are stable or not. Consider the cost function (2.11) as the Lyapunov function for the learner agent i. The learner agent i will be asymptotically stable [45, 68], if one proves $V_i^h(\delta_i) \geq 0$, $\dot{V}_i^h(\delta_i) \leq 0$, and $V_i^h(\delta_i) = \dot{V}_i^h(\delta_i) = 0$ hold only when $\delta_i = 0$.

First, it is seen from (2.13) that u_i^h is affine in the δ_i. Hence, when $\delta_i = 0$, then $V_i^h(0) = 0$. Second, the convergence proof of Algorithm 1's inner loop in Theorem 1 shows

that (2.19) holds. Theorem 1 implies that $Q_i^h > 0$ for any iteration h. Then, from (2.12), one can deduce $V_i^h(\delta_i) > 0$, for $\delta_i \neq 0$, $\forall i \in \mathcal{N}$.

Third, to prove $\dot{V}_i^h(\delta_i) \leq 0, \delta_i \neq 0$, for the outer iteration loop h, $h = 0, 1, \ldots, \infty$, one has

$$\begin{aligned}\frac{dV_i^h}{dt} &= \nabla(V_i^h)^{\mathrm{T}}\dot{\delta}_i \\ &= 2\delta_i^{\mathrm{T}}P_i^h\left(A\delta_i + d_iBu_i^h - \sum_{j\in N_i} e_{ij}Bu_j^h\right). \end{aligned} \tag{2.31}$$

One can rewrite the HJB equation for each learner agent i at iteration h from (2.14) and (2.31) as

$$\begin{aligned}0 =& \delta_i^{\mathrm{T}}Q_i^h\delta_i + {d_i}^2\delta_i^{\mathrm{T}}P_i^hBR_i^{-1}B^{\mathrm{T}}P_i^h\delta_i \\ &+ 2\delta_i^{\mathrm{T}}P_i^hA\delta_i + 2d_i\delta_i^{\mathrm{T}}P_i^h\sum_{j\in N_i} e_{ij}BR_j^{-1}B^{\mathrm{T}}P_j^h\delta_j \\ =& \delta_i^{\mathrm{T}}Q_i^{(h-1)}\delta_i + \left(u_{ie}^* - u_i^{(h-1)}\right)^{\mathrm{T}}R_i\left(u_{ie}^* - u_i^{(h-1)}\right) \\ &+ {d_i}^2\delta_i^{\mathrm{T}}{P^h}_iBR_i^{-1}B^{\mathrm{T}}P_i^h\delta_i + \dot{V}_i^h(\delta_i). \end{aligned} \tag{2.32}$$

It is seen in Theorem 1 that $Q_i^h \geq 0$ for any iteration h and $R_i > 0$. Therefore, from (2.32), $\dot{V}_i^h(\delta_i) \leq 0\ \forall i = 1, 2, .., \mathcal{N}$.

Hence, the learner (2.10) is asymptotically stable for any iteration of Algorithm 1. This completes the stability analysis of the learner with optimal synchronization using Algorithm 1. This finishes the stability proof. □

Remark 5. Note that, $\forall i \in \mathcal{N}$, R_i in (2.12) is arbitrarily selected by the learner. Theorems 1 and 2 show that $u_i^h \to u_{ie}^*$ and $Q_i^h \to \bar{Q}_i$ by learning the equivalent weight to Q_{ie}.

Remark 6. It is known that expert MAS has optimal control policy u_{ie}^*, $\forall i \in \mathcal{N}$. It is seen from Theorem 1 that the learner control policy converges to the expert control policy. Therefore, the learner MAS controller also attains optimality. Each learner agent is asymp-

totically stable as shown in Theorem 3. If all the agents choose their own optimal policy $(u_1^*, u_2^*, \ldots, u_N^*)$, then, with Assumption 1, all learner agents optimally synchronize.

2.4 Model-Free Inverse RL Algorithm

In inverse RL Algorithm 1, the Graphical Apprentice Game problem requires system dynamics matrices A and B. Alternatively, this section proposes a model-free integral inverse RL algorithm to solve the Graphical Apprentice Game problem using expert-demonstrated data and learner behavioral trajectory data. The online implementation of this model-free algorithm is then provided with rigorous convergence analysis.

Applying off-policy integral RL [95] to inner iteration loops of Algorithm 1 finds the model-free form of (2.16) and (2.17). The error dynamics of learner agent i in (2.10) becomes

$$\begin{aligned}\dot{\delta}_i =& A\delta_i + d_i B u_i^{hk} - \sum_{j\in N_i} e_{ij} B u_j^{hk} \\ &+ d_i B(u_i - u_i^{hk}) - \sum_{j\in N_i} e_{ij} B(u_j - u_j^{hk}),\end{aligned} \tag{2.33}$$

where control input $u_i^{hk} \in \mathbb{R}^p$ is the update in inner k iteration loop given $Q_i^h \geq 0$ and $R_i > 0$. Taking the derivative of V_i^{hk} with respect to (2.33) yields

$$\begin{aligned}\dot{V}_i^{hk} =& (\nabla V_i^{hk})^{\mathrm{T}} \dot{\delta}_i \\ =& 2\delta_i^{\mathrm{T}} P_i^{hk} \left(A\delta_i + d_i B u_i^{hk} - \sum_{j\in N_i} e_{ij} B u_j^{hk} \right) \\ &+ 2\delta_i^{\mathrm{T}} P_i^{hk} \left(d_i B(u_i - u_i^{hk}) - \sum_{j\in N_i} e_{ij} B(u_j - u_j^{hk}) \right).\end{aligned} \tag{2.34}$$

Using (2.16) and (2.17) from Algorithm 1 in (2.34) gives

$$\begin{aligned}\dot{V}_i^{hk} =& -\delta_i{}^T Q_i^h \delta_i - (u_i^{hk})^{\mathrm{T}} R_i u_i^{hk} - 2u_i^{h(k+1)} R_i (u_i - u_i^{hk})^{\mathrm{T}} \\ &+ 2d_i^{-1} \sum_{j\in N_i} e_{ij} u_i^{h(k+1)} R_i (u_j - u_j^{hk})^{\mathrm{T}}.\end{aligned} \tag{2.35}$$

Integrating both sides of (2.35) for interval t to $t+T$ gives the Bellman equation (2.36) to be presented in Algorithm 2.

Given the prevailing Q_i^h, (2.36) satisfies the condition where an $\mathcal{N}$-tuple $\{u_1^*, u_2^*, ..., u_N^*\}$ of policy must offer a solution for the learner MAS, in the way that $V_i^* \cong V_i(\delta_i(t_0), u_i^*, u_{-i}^*) \leq V_i(\delta_i(t_0), u_i, u_{-i}^*)$. To find a model-free version for the revision of Q_i^h in (2.18), we integrate both sides of (2.18) from t to $t+T$, and yield (2.37) to be presented in Algorithm 2. The learner agent i updates Q_i^h in (2.37) towards the equivalent weight $\bar{Q}_i$ by using the Nash solutions V_i^* and u_i^* from inner loops and expert agent's demonstration u_{ie}^*.

Assumption 5. Suppose initial control input u_i^{00} is stabilizing for each learner agent i, where $i \in \mathcal{N}$, in Algorithm 2.

Remark 7. Initial stabilizing control input is a standard assumption for model-free policy iteration of RL, see [15, 58, 95, 112, 150]. In practice, it is usually found heuristically, since there could be infinitely many stabilizing policies.

Assumption 6. Assume that the learner MAS in the Algorithms 2 and 3 are persistently-excited (PE) by introducing a probing noise in the stabilizing control inputs [22, 58, 107]. The PE condition makes system states persistently present for a sufficient time to retrieve a unique reward function.

Remark 8. Note that the value of T does not affect the convergence of Algorithm 2, and is selected based on the excitation condition necessary for the numerical setup of the Batch Least squares (BLSs) solution. As shown in [141], T satisfies $T > \frac{\alpha l}{\prod_{i=1}^{n} |\lambda_i|}$, where $\alpha > 0$ is an scaling factor, $l > 0$ is the lower bound, and λ_i indicates eigenvalues of the closed-loop system.

Based on the above analysis, the Algorithm 2 of model-free inverse RL for the graphical apprentice game is now presented.

The following theorem result shows that Algorithm 2 converges to Algorithm 1.

Algorithm 2 Model-free Inverse RL Algorithm for Graphical Apprentice Game

1 **Initialization:** For each learner agent i, $i \in \mathcal{N}$, select initial $Q_i^0 \geq 0$, $R_i > 0$ and stabilizing control input u_i^{00}. Set $h = 0$ and small thresholds e_i and ε_i. Apply stabilizing $u_1, u_2, \ldots u_N$ to (20);

2 **Outer h iteration loop using IOC**

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

4 Off-policy RL for solving V_i^h and u_i^h,

$$\begin{aligned} & V_i^{hk}\big(\delta_i(t+T)\big) - V_i^{hk}\big(\delta_i(t)\big) \\ & = \int_t^{t+T} -\left(\delta_i{}^{\mathrm{T}} Q_i^h \delta_i + (u_i^{hk})^{\mathrm{T}} R_i u_i^{hk}\right) d\tau \\ & \quad - \int_t^{t+T} 2u_i^{h(k+1)} R_i (u_i - u_i^{hk})^{\mathrm{T}} d\tau \\ & \quad + \int_t^{t+T} 2d_i^{-1} \sum_{j \in N_i} e_{ij} u_i^{h(k+1)} R_i (u_j - u_j^{hk})^{\mathrm{T}} d\tau; \end{aligned} \tag{2.36}$$

5 **Stop if** $\|V_i^{hk} - V_i^{h(k-1)}\| \leq e_i$, then set $V_i^h = V_i^{hk}$, $u_i^h = u_i^{hk}$, and go to step 6; Otherwise, set $k \leftarrow k+1$ and go to step 4;

6 **Outer h iteration loop using IOC:** Q_i^h revision using expert MAS optimal control input u_{ie}^*

$$\begin{aligned} \int_t^{t+T} \delta_i{}^{\mathrm{T}} Q_i^{h+1} \delta_i d\tau = & \int_t^{t+T} (u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h) d\tau \\ & - V_i^h(t+T) + V_i^h(t); \end{aligned} \tag{2.37}$$

7 **Stop if** $\|Q_i^{h+1} - Q_i^h\| \leq \varepsilon_i$; Otherwise, set $u_i^{(h+1)0} = u_i^h$, $h \leftarrow h+1$, and go to Step 3.

Theorem 4. *(**Convergence of Algorithm 2**) The Algorithm 2 converges to Algorithm 1, and each learner agent $i \in \mathcal{N}$ has $(x_i, u_i^h) \to (x_{ie}, u_{ie}^*)$ for finite iterations h.*

Proof. To prove the convergence of Algorithm 2 to Algorithm 1, one can show the convergence of both inner iteration loops and outer iteration loops between two algorithms. First, the convergence of inner loops between Algorithm 1 and Algorithm 2 is proven by dividing (2.36) by T and taking limit, i.e.,

$$\begin{aligned}
&\lim_{T\to 0} \frac{V_i^{hk}(\delta_i(t+T)) - V_i^{hk}(\delta_i(t))}{T} \\
&+ \lim_{T\to 0} \frac{\int_t^{t+T} \left(\delta_i{}^{\mathrm{T}} Q_i^h \delta_i + (u_i^{hk})^{\mathrm{T}} R_i u_i^{hk}\right) d\tau}{T} \\
&+ \lim_{T\to 0} \frac{2\int_t^{t+T} u_i^{h(k+1)} R_i (u_i - u_i^{hk})^{\mathrm{T}} d\tau}{T} \\
&- \lim_{T\to 0} \frac{2d_i^{-1}\int_t^{t+T} \sum_{j\in N_i} e_{ij} u_i^{h(k+1)} R_i (u_j - u_j^{hk})^{\mathrm{T}} d\tau}{T} \\
&= 0.
\end{aligned} \tag{2.38}$$

Using L'Hospital's rule,

$$\begin{aligned}
&2\delta_i^{\mathrm{T}} P_i \Bigg(A\delta_i + d_i B u_i^{hk} - \sum_{j\in N_i} e_{ij} B u_j^{hk} + d_i B(u_i - u_i^{hk}) \\
&- \sum_{j\in N_i} e_{ij} B(u_j - u_j^{hk}) \Bigg) + \delta_i{}^T Q_i^h \delta_i + (u_i^{hk})^{\mathrm{T}} R_i u_i^{hk} \\
&+ 2u_i^{h(k+1)} R_i (u_i - u_i^{hk})^{\mathrm{T}} - 2d_i^{-1} \sum_{j\in N_i} e_{ij} u_i^{h(k+1)} R_i (u_j - u_j^{hk})^{\mathrm{T}} \\
&= 0.
\end{aligned} \tag{2.39}$$

Substituting the updated policies $u_i^{h(k+1)}$ from (2.17) into (2.39) gives (2.16). Therefore, the off-policy RL equation (2.36) defines the same solution as equation (2.16). For the outer-iteration loop,

$$\begin{aligned}
&\lim_{T\to 0} \frac{\int_t^{t+T} \delta_i{}^{\mathrm{T}} Q_i^{(h+1)} \delta_i d\tau}{T} \\
&= \lim_{T\to 0} \frac{\int_t^{t+T} \left(u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h\right) d\tau}{T} \\
&- \lim_{T\to 0} \frac{\left(V_i^h(t+T) - V_i^h(t)\right)}{T}.
\end{aligned} \tag{2.40}$$

Using L'Hospital's rule,

$$\begin{aligned}\delta_i{}^T Q_i^{(h+1)}\delta_i =& \left(u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i u_i^h\right) \\ & - 2\delta_i^{\mathrm{T}} P_i \Bigg(A\delta_i + d_i B u_i^h - \sum_{j\in N_i} e_{ij} B u_j^h \\ & + d_i B(u_i - u_i^h) - \sum_{j\in N_i} e_{ij} B(u_j - u_j^h)\Bigg). \end{aligned} \tag{2.41}$$

Substituting the updated policies $u_i^{h(k+1)}$ in (2.41) gives (2.18). Therefore, (2.37) defines the same solution as equation (2.18). This finishes the proof. □

2.4.0.1 Data-driven Implementation of Algorithm 2

This implementation of model-free inverse RL consists of two learning units. The first one is for the inner k iteration loop to compute the learner MAS value function V_i^{hk}. The second learning unit is in the outer h iteration loop to update Q_i^h using the expert MAS demonstration u_{ie}^* and the converged optimal solutions from inner loops. To implement steps 4 and 5 of Algorithm 2 in terms of online data using $V_i^{hk} = \delta_i^{\mathrm{T}} P_i^{hk} \delta_i$ and $u_i^{hk} = -K_i^{hk}\delta_i$, (2.36) can be rewritten as

$$\begin{aligned} & \delta_i(t+T)^{\mathrm{T}} P_i^{hk} \delta_i(t+T) - \delta_i(t)^{\mathrm{T}} P_i^{hk} \delta_i(t) \\ =& \int_t^{t+T} -\left(\delta_i{}^{\mathrm{T}}\left(Q_i^h + \left(K_i^{hk}\right)^{\mathrm{T}} R_i K_i^{hk}\right)\delta_i\right) d\tau \\ & + \int_t^{t+T} 2\left(u_i + K_i^{hk}\delta_i\right)^{\mathrm{T}} R_i \left(K_i^{h(k+1)}\delta_i\right) d\tau \\ & - \int_t^{t+T} 2d_i^{-1} \sum_{j\in N_i} e_{ij}\left(u_j + K_j^{hk}\delta_i\right)^{\mathrm{T}} R_i \left(K_i^{h(k+1)}\delta_i\right) d\tau. \end{aligned} \tag{2.42}$$

Next, using the Kronecker product for (2.42), one has

$$\begin{aligned}
&\Big(\operatorname{vecv}(\delta_i(t+T)-\delta_i(t))^{\mathrm{T}}\Big)\operatorname{vec}(P_i^{hk})\\
&\quad=\int_t^{t+T}-\big(\delta_i^{\mathrm{T}}\otimes\delta_i^{\mathrm{T}}\big)\operatorname{vec}\Big(Q_i^h+\big(K_i^{hk}\big)^{\mathrm{T}}R_iK_i^{hk}\Big)d\tau\\
&\quad\quad+\int_t^{t+T}2\Big[\big(\delta_i^{\mathrm{T}}\otimes u_i^{\mathrm{T}}\big)\big(I_n\otimes R_i\big)\\
&\quad\quad+\big(\delta_i^{\mathrm{T}}\otimes\delta_i^{\mathrm{T}}\big)\big(I_n\otimes K_i^{hk}R_i\big)\Big]\operatorname{vec}\big(K_i^{h(k+1)}\big)d\tau\\
&\quad\quad-\int_t^{t+T}2d_i^{-1}\sum_{j\in N_i}e_{ij}\Big[\big(\delta_i^{\mathrm{T}}\otimes u_j^{\mathrm{T}}\big)\big(I_n\otimes R_i\big)\\
&\quad\quad+\big(\delta_i^{\mathrm{T}}\otimes\delta_i^{\mathrm{T}}\big)\big(I_n\otimes K_j^{hk}R_i\big)\Big]\operatorname{vec}\big(K_i^{h(k+1)}\big)d\tau. \qquad (2.43)
\end{aligned}$$

The following operators are defined to compute unknown $\operatorname{vec}(P_i^{hk})$ and $\operatorname{vec}(K_i^{h(k+1)})$ in (2.43) using BLSs.

$$\begin{aligned}
\sigma_{\delta_i\delta_i}=&\Big[\big(\operatorname{vecv}(\delta_i(t+T)-\delta_i(t))\big),\ldots,\big(\operatorname{vecv}(\delta_i(t+\zeta T)\\
&-\delta_i(t+(\zeta-1)T))\big)\Big]^{\mathrm{T}}, \qquad (2.44a)
\end{aligned}$$

$$\rho_{\delta_i\delta_i}=\Big[\int_t^{t+T}(\delta_i\otimes\delta_i)d\tau,\ldots,\int_{t+(\zeta-1)T}^{t+\zeta T}(\delta_i\otimes\delta_i)d\tau\Big]^{\mathrm{T}}, \qquad (2.44b)$$

$$\rho_{\delta_iu_i}=\Big[\int_t^{t+T}(\delta_i\otimes u_i)d\tau,\ldots,\int_{t+(\zeta-1)T}^{t+\zeta T}(\delta_i\otimes u_i)d\tau\Big]^{\mathrm{T}}, \qquad (2.44c)$$

$$\begin{aligned}
\varphi_i^{hk}=&\Big[\sigma_{\delta_i\delta_i},-2\rho_{\delta_iu_i}(I_n\otimes R_i)-2\rho_{\delta_i\delta_i}(I_n\otimes K_i^{hk}R_i)\\
&+d_i^{-1}\sum_{j\in N_i}-2\rho_{\delta_iu_i}(I_n\otimes R_i)-2\rho_{\delta_i\delta_i}(I_n\otimes K_j^{hk}R_i)\Big], \qquad (2.44d)
\end{aligned}$$

$$\Omega_i^{hk}=-\rho_{\delta_i\delta_i}\operatorname{vec}\big(Q_i^h+(K_i^{hk})^{\mathrm{T}}R_iK_i^{hk}\big), \qquad (2.44e)$$

$$\zeta \geq (n+1)n/2 + mn, \tag{2.45}$$

where ζ is the group number of sampling data such that

$$\operatorname{rank}[\varphi_i^{hk} \quad \Omega_i^{hk}] = (n+1)n/2 + mn. \tag{2.46}$$

Then, P_i^{hk} and $K_i^{h(k+1)}$ can be uniquely solved by

$$\begin{bmatrix} P_i^{hk} \\ \operatorname{vec}\left(K_i^{h(k+1)}\right) \end{bmatrix} = \left((\varphi_i^{hk})^{\mathrm{T}} \varphi_i^{hk}\right)^{-1} (\varphi_i^{hk})^{\mathrm{T}} \Omega_i^{hk}. \tag{2.47}$$

Repeat (2.47) by setting $k \leftarrow k+1$ and stop on convergence of P_i^{hk} and $K_i^{h(k+1)}$ i.e., $P_i^{hk} \to P_i^h$ and $K_i^{h(k+1)} \to K_i^h$. This P_i^h and K_i^h are then used to revise Q_i^h in outer iteration loop.

Next, to implement steps 6 and 7 of Algorithm 2 in a data-driven style, (2.37) can be rewritten as

$$\int_t^{t+T} \operatorname{vecv}(\delta_i)^{\mathrm{T}} \operatorname{vecm}(Q_i^{h+1}) d\tau = \beta_i^h(t) + \gamma_i^h(t), \tag{2.48}$$

where

$$\begin{aligned} \gamma_i^h(t) &= (\operatorname{vecv}(\delta_i(t))^{\mathrm{T}} - \operatorname{vecv}(\delta_i(t+T))^{\mathrm{T}}) \operatorname{vec}(P_i^h), \\ \beta_i^h(t) &= \int_t^{t+T} \left(u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i (K_i^h \delta_i)\right) d\tau. \end{aligned}$$

Similarly, the following operators are defined to compute the state reward weight Q_i^h using BLSs

$$\eta_i = \left[\int_t^{t+T} \operatorname{vecv}(\delta_i)^{\mathrm{T}}, \ldots, \int_{t+(\iota-1)T}^{t+\iota T} \operatorname{vecv}(\delta_i)^{\mathrm{T}}\right], \tag{2.49a}$$

$$\Gamma_i^h = \left[\beta_{i_1}^h(t) + \gamma_{i_1}^h(t), \ldots, \beta_{i_\iota}^h(t) + \gamma_{i_\iota}^h(t)\right] \tag{2.49b}$$

where

$$\beta_{i_\iota}^h(t) = \int_{t+\iota T}^{t+(\iota-1)T} \left(u_{ie}^{*\mathrm{T}} R_i u_{ie}^* - 2u_{ie}^{*\mathrm{T}} R_i (K_i^h \delta_i)\right) d\tau,$$
$$\gamma_{i_\iota}^h(t) = \left(\mathrm{vecv}(\delta_i(t+(\iota-1)T))^{\mathrm{T}} - \mathrm{vecv}(\delta_i(t+\iota T))^{\mathrm{T}}\right) \mathrm{vec}(P_i^h).$$

The state reward weight $\hat{Q}_i^{h+1}$ has $(n+1)n/2$ unknown parameters, and is uniquely solved by constructing $\iota \geq (n+1)n/2$ equations using BLSs and letting the following rank condition hold

$$\mathrm{rank}[\eta_i \quad \Gamma_i^h] = (n+1)n/2. \tag{2.50}$$

Then, $\hat{Q}_l^{h+1}$ is uniquely solved from

$$\mathrm{vec}(\hat{Q}_i^{h+1}) = \left(\eta_i^T \eta_i\right)^{-1} \eta_i^{\mathrm{T}} \Gamma_i^h. \tag{2.51}$$

Repeat (2.51) by setting $h \leftarrow h+1$ until the convergence of Q_i. Otherwise, $u_i^{(h+1)0} = u_i^h$, and go to the inner iteration loop. We summarize an online implementation of the Algorithm 2 in the following Algorithm 3.

Remark 9. While using BLSs to compute unique P_i^{hk}, $K_i^{h(k+1)}$, and Q_i^{h+1}, full-rank conditions (2.46) and (2.50) are satisfied in Algorithm 3.

Theorem 5. *(**Convergence of Algorithm 3**) Given Assumptions 1, 2, and 3, and with initial $Q_i^0 \geq 0$ and $R_i > 0$, $\forall i \in \mathcal{N}$, use Algorithm 3 to solve the Graphical Apprentice Game problem given rank conditions* (2.46) *and* (2.50)*. Then, Algorithm 3 converges to Algorithm 1 and $(x_i, u_i^h) \to (x_{ie}, u_{ie}^*)$.*

Proof. It is seen from Theorem 1 that one has $Q_i^h \geq 0$ holding for $\forall i \in \mathcal{N}$ and each outer iteration h. Note that $u_i^{h(k+1)}$ and P_i^{hk} can be uniquely found using the policy iteration learning from inner loops in Algorithm 1. When inner loops have converged to u_i^h and P_i^h, Q_i^{h+1} can be uniquely determined by solving (2.37).

Note that in the data-driven implementation of Algorithm 2, P_i^{hk} and $K_i^{h(k+1)}$ can be uniquely solved by BLSs in (2.47) while satisfying the full rank condition (2.46). Then,

Algorithm 3 Data-driven Implementation of Model-free Inverse RL Algorithm 2

1 **Initialization:** For each learner agent i, $i \in \mathcal{N}$, select initial $Q_i^0 \geq 0$, $R_i > 0$ and stabilizing control input u_i^{00}. Set $h = 0$ and small thresholds e_i and ε_i. Apply stabilizing $u_1, u_2, \ldots u_N$ to the learner dynamics;

2 **Outer h iteration loop using IOC**

3 **Inner k iteration loop using optimal control:** Given h, Set $k = 0$;

4 Off-policy RL: Solves V_i^h and u_i^h using (2.47);

5 **Stop if** $\|P_i^{h(k+1)} - P_i^{hk}\| \leq e_i$, then set $K_i^h = K_i^{hk}$, $P_i^h = P_i^{hk}$, and go to step 6, if not set $k \leftarrow k+1$ and go to step 4;

6 **Outer h iteration loop using IOC:** Q_i^h revision using expert's demonstration u_{ie}^* by (2.51);

7 **Stop if** $\|Q_i^{h+1} - Q_i^h\| \leq \varepsilon_i$, if not set $u_i^{(h+1)0} = u_i^h$, $h \leftarrow h+1$, and go to Step 3.

(2.47) solves for the converged solution (P_i^h, K_i^h). Similarly, Q_i^{h+1} is uniquely solved with BLSs in (2.51) by guaranteeing the full rank condition (2.50).

It is shown that each step of Algorithm 3 is developed from Algorithm 2 and generates the same unique solution. That is, (2.36) and (2.37) of Algorithm 2 are equivalent to (2.47) and (2.51) for Algorithm 3, respectively. Moreover, Theorem 4 shows the convergence of Algorithm 2 to Algorithm 1. Therefore, Algorithm 3 converges to Algorithm 1 with the same solutions. That is, Algorithm 3 obtains the converged behavior $(x_i, u_i^*) = (x_{ie}, u_{ie}^*)$ and the equivalency of $\bar{Q}_i$ to Q_{ie}, where $i \in \mathcal{N}$. This finishes the proof. □

2.5 Simulation Results

We implement the model-free inverse RL in Algorithm 3 for optimal synchronization control of DC microgrids. The microgrid schematic is modified from [137], and given in Figure 2.2. The DC/DC notation in Figure 2.2 refers to a DC-DC buck converter and its augmented LC filter. The physical filter and distribution line parameters, as per [137], are $C = 2.2mF, L = 1.8mH, R = 0.2\Omega, R_{ij} = 0.05\Omega$, $R_l = 4\Omega$. A strongly connected directed graph topology, with three agents for either the expert and learner MAS, is considered. Each edge of the graph is assigned a weight of 1. The system dynamics for the linear expert and learner MAS are

$$\dot{x}_{ie} = Ax_{ie} + Bu_{ie}, \tag{2.52a}$$

$$\dot{x}_i = Ax_i + Bu_i. \tag{2.52b}$$

Dynamics of each agent i are adopted from [137]

$$\dot{x}_{i,1} = \frac{1}{C}x_{i,2} - \Big(\frac{1}{R_l C} + \sum_{j\in\mathcal{N}_i}\frac{1}{R_{ij}C}\Big)x_{i,1}, \tag{2.53a}$$

$$\dot{x}_{i,2} = \frac{1}{L}V_{in} - \frac{1}{L}x_{i,1} - \frac{R}{L}x_{i,2}, \tag{2.53b}$$

where $x_i = [x_{i,1}, x_{i,2}]^{\mathrm{T}} = [V, I]^{\mathrm{T}}$ is the state. V_{in} is the control input and is considered as the output voltage of the corresponding DC-DC converter. The state matrices are

$$A = \begin{bmatrix} -\Big(\frac{1}{R_l C} + \sum_{j\in\mathcal{N}_i}\frac{1}{R_{ij}C}\Big) & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix}. \tag{2.54}$$

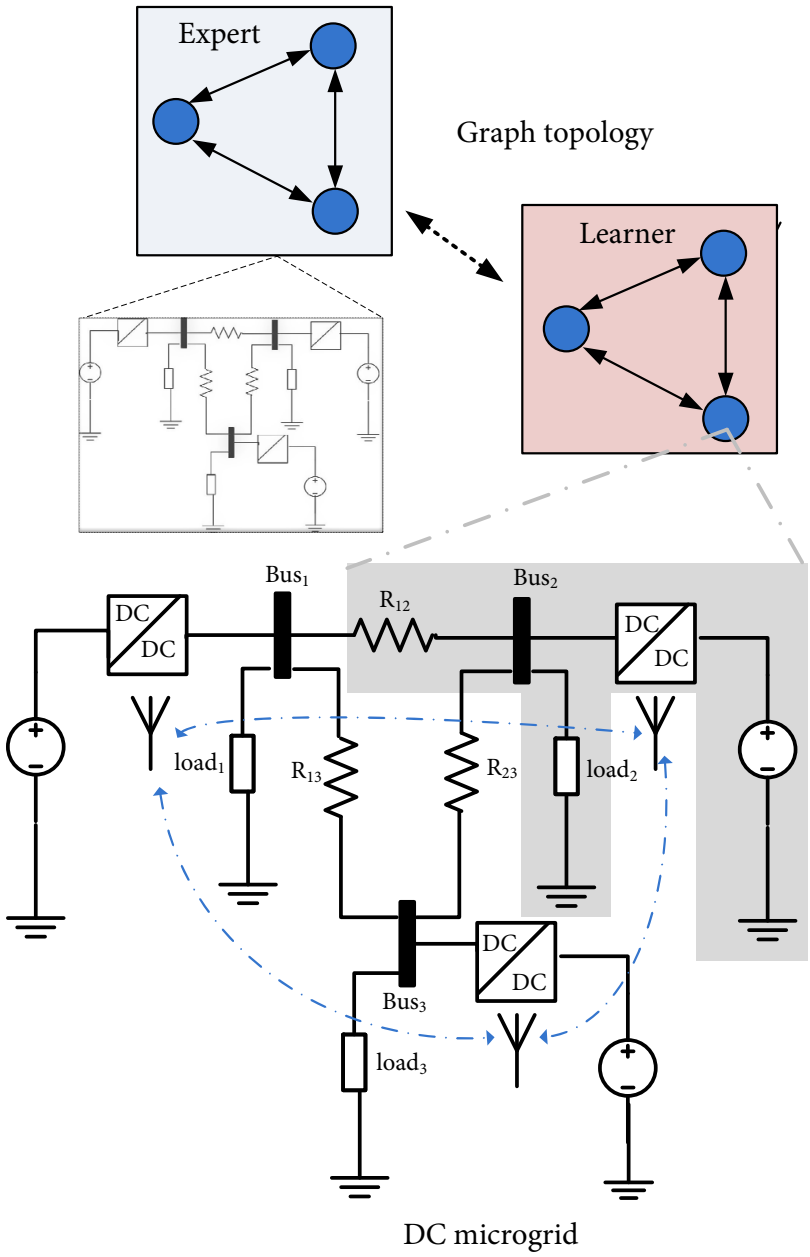


Figure 2.2. DC microgrid schematic with three distributed DC sources, the communication graph topology among DC-DC converters, and the graphical representations of the expert and the learner MAS..

TABLE I shows the parameters of the learner and expert MAS. Small thresholds e_i and ε_i, used to terminate inner and outer iteration during online data collecting, are 0.0001

Table 2.1. Simulation Parameter values

Parameter	**Expert MAS**	**Learner MAS**
State reward Weight	$Q_{ie} = 10I_2$	$Q_i = I_2$
Control weight	$R_{ie} = 1$	$R_i = 1$
Initial conditions	$\delta_{ie}^0 = [1, \quad 1]^T$	$\delta_i^0 = [1, \quad 1]^T$

and 0.03, respectively. A small random probing noise of $z_i = 0.01 \times rand(1)$ is used to satisfy the PE condition. The sampling time T is chosen as 0.008. The parameter matrix P_{ie} and the feedback control gain K_{ie} of the target expert MAS computed through optimal control with state reward weight Q_{ie} given in TABLE I are

$$P_{e_1} = \begin{bmatrix} 0.0355 & 0.1088 \\ 0.1088 & 0.0428 \end{bmatrix}, \quad K_{e_1} = \begin{bmatrix} 0.16 & -0.09 \end{bmatrix}.$$

$$P_{e_2} = \begin{bmatrix} 0.0331 & 0.1168 \\ 0.1168 & 0.0475 \end{bmatrix}, \quad K_{e_2} = \begin{bmatrix} 0.17 & -0.08 \end{bmatrix}.$$

$$P_{e_3} = \begin{bmatrix} 0.0357 & 0.1154 \\ 0.1154 & 0.0476 \end{bmatrix}, \quad K_{e_3} = \begin{bmatrix} 0.14 & -0.09 \end{bmatrix}.$$

Figure 2.3 shows the learning of the feedback control gain for learner MAS and notes that K_i^h converges to K_i^∞ associated with P_i^∞ derived from the expert MAS. The number of inner iteration loops k, to achieve convergence for all agents, is 300. Figure 2.4 demonstrates the learning of the state reward weight Q_i^h.

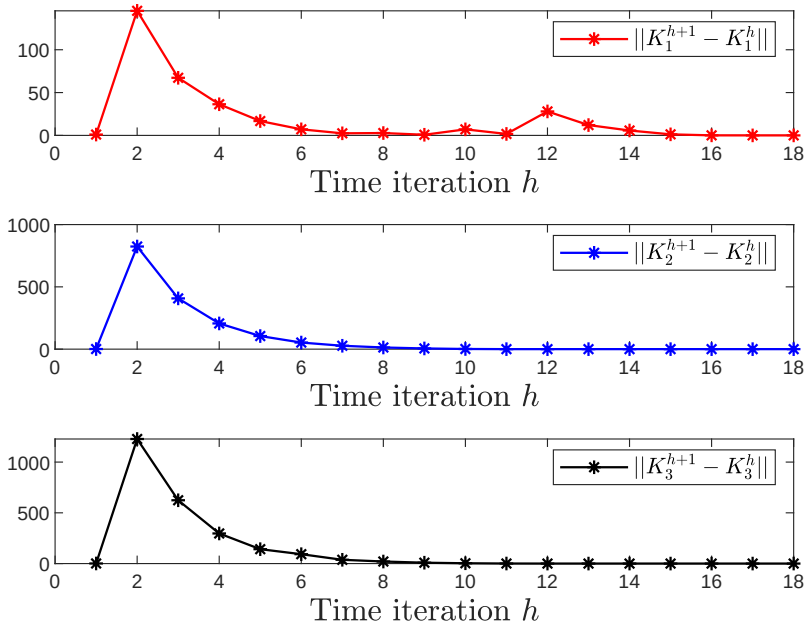


Figure 2.3. Convergence of the linear learner MAS feedback gain K_i.

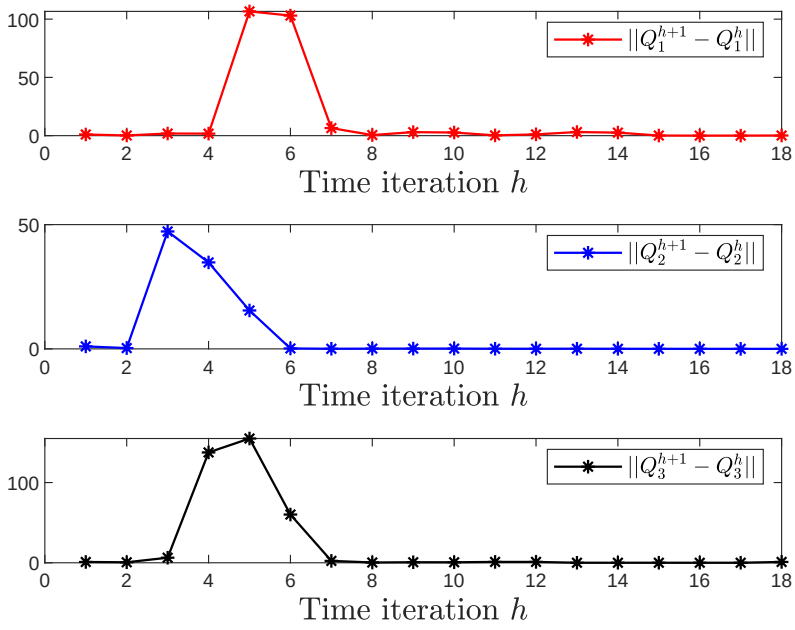


Figure 2.4. Convergence of the linear learner MAS state reward weight Q_i^h.

Note that the state reward weight Q_i^h converges to $\bar{Q}_i$. Ultimately, Q_i^h, P_i^h, and K_i^h of the learner MAS converge to

$$\bar{Q}_1 = \begin{bmatrix} -1.2697 & 0.5965 \\ 0.5965 & -0.0317 \end{bmatrix}, \quad P_1^\infty = \begin{bmatrix} 0.0384 & 0.1015 \\ 0.1015 & 0.0405 \end{bmatrix}.$$

$$\bar{Q}_2 = \begin{bmatrix} -1.4967 & 0.2055 \\ 0.2055 & -1.2834 \end{bmatrix}, \quad P_2^\infty = \begin{bmatrix} 0.0312 & 0.1126 \\ 0.1126 & 0.0428 \end{bmatrix}.$$

$$\bar{Q}_3 = \begin{bmatrix} -0.8438 & 0.1334 \\ 0.1334 & -0.8763 \end{bmatrix}, \quad P_3^\infty = \begin{bmatrix} 0.0328 & 0.1118 \\ 0.1118 & 0.0431 \end{bmatrix}.$$

$$K_1^\infty = \begin{bmatrix} 0.1504 & -0.0911 \end{bmatrix}, \quad K_2^\infty = \begin{bmatrix} 0.1721 & -0.0949 \end{bmatrix},$$

$$K_3^\infty = \begin{bmatrix} 0.1723 & -0.0932 \end{bmatrix}.$$

It is seen that $\bar{Q}_i$ is not exactly the same as the desired Q_{ie} shown in Theorem 1 of convergence. According to Definition 2, obtained $\bar{Q}_i$ is equivalent to Q_{ie} and might not be unique as per Theorem 2. However, as shown in Theorem 2 and the convergence Theorem 1, learner MAS K_i^∞ converges to K_{ie}^*.

The expert microgrid has the optimal trajectories for state and control given Q_{ie} and R_{ie}. The learner microgrid observes K_{ie}^* of the expert microgrid computed with optimal control. Using the converged K_i^∞ of the learner MAS, which starts from the same initial conditions as expert MAS, we obtain learner behavioral trajectories depicted in Figures 2.5 and 2.6. In Figure 2.5, δ_i and δ_{ei} denote the i^{th} global synchronization error of the learner and the expert MAS, respectively. It is seen from Figures 2.5 and 2.6 that the learner microgrid exhibits the same behavioral trajectory as the expert (x_{ie}, u_{ie}^*).

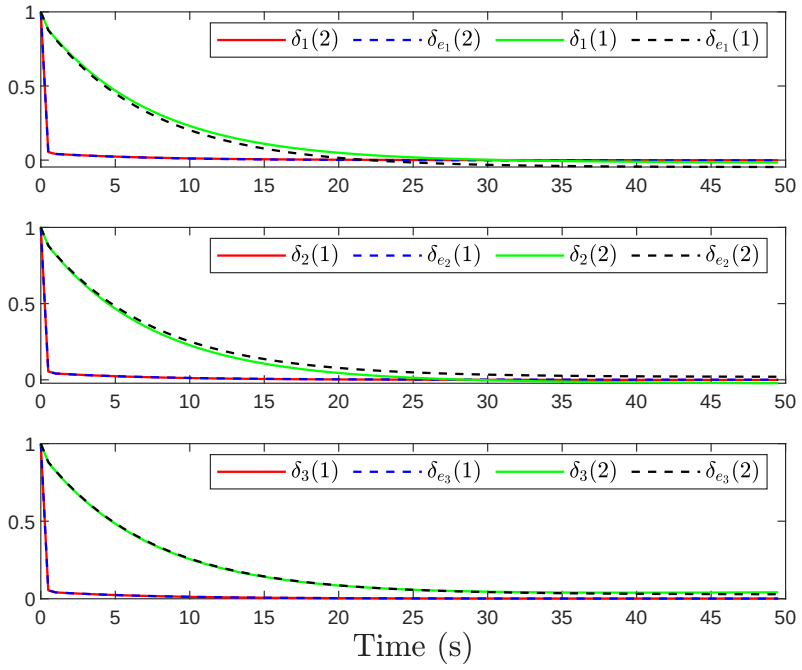


Figure 2.5. Evolution of local synchronization error of learner and expert MAS with time.

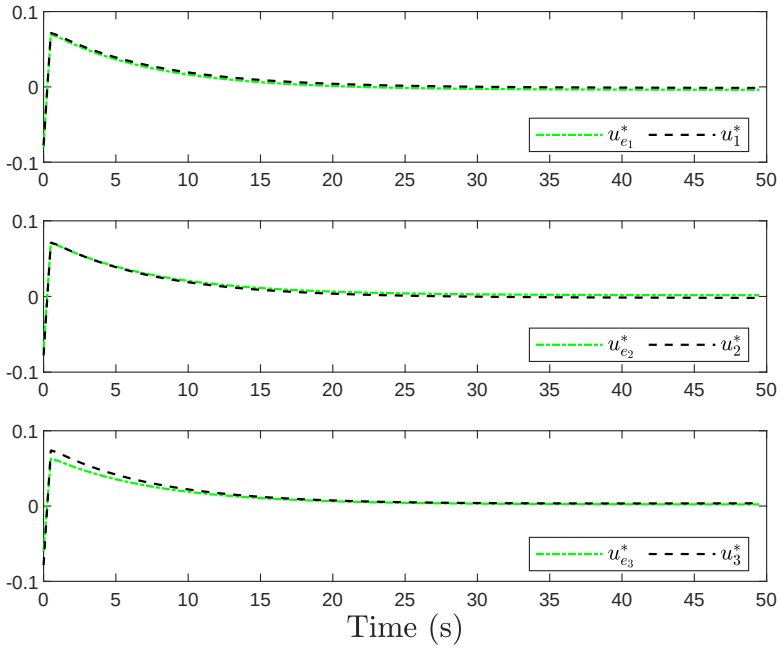


Figure 2.6. Evolution of linear MAS inputs for learner and expert with time.

2.6 Conclusion

This paper proposes an inverse RL paradigm to address the optimal synchronization of MAS. A model-based inverse RL algorithm and a model-free off-policy inverse RL algorithm solve a graphical apprentice game and attain optimal synchronization of MAS. The algorithm convergence and learner stability are ensured. The learner obtains the expert's unknown reward weights, which are not unique, from the expert's demonstrations. Potential extensions include heterogeneous nonlinear multi-agent systems, or different graph topologies for expert and learner MAS.

Future extensions could consider adaptation of the proposed tools to cooperative control of multiple power microgrids. States of islanded multi-microgrids need to be synchronized before microgrids are reconnected. Given the demonstrated trajectories of "expert" microgrids, the "learner" microgrids imitate their unknown cost functions with inverse RL such that learner states are synchronized to the common values set by the expert microgrids.

Chapter 3

Accelerated Reinforcement Learning via Dynamic Mode Decomposition *

*This chapter, titled 'Accelerated Reinforcement Learning via Dynamic Mode Decomposition,' was originally published in IEEE Transactions on Control of Network Systems, doi: 10.1109/TCNS.2023.3259060. It has been reprinted with permission from all co-authors and is used with permission from IEEE without any revisions. This version represents the authors' accepted manuscript.

CHAPTER 3

Accelerated Reinforcement Learning via Dynamic Mode Decomposition

3.1 Introduction

Large-scale systems, frequently constructed from lower-dimensional subsystems, are becoming more common [94]. Examples include the air-traffic systems, where the physical agents are decoupled dynamically but are linked via a mutual performance function, and power systems, where physical agents are coupled both dynamically and through a mutual performance function. Conventional optimal control [79] might need the knowledge of system dynamics, whereas reinforcement learning (RL) [130] could provide optimal solutions using behavior data without knowing system dynamics. As a system grows larger, distributed control design via RL is a viable option, but the iterative learning involved could become computationally expensive. It is desired to seek a low-dimensional abstraction of the original large-scale system while retaining essential dynamics.

The work in [30,101] study model-based and model-free RL algorithms with reduced-order optimal control problems by assuming a time lag in system dynamics. [59, 118, 119] study RL control algorithms for linear multi-agent systems with dimensionality reduction based on controllability and observability gramians. The majority of existing work on RL with decomposition do not extract complete dynamic information from the original system. This information could become crucial in a networked system where the control objective of each subsystem depends on its states and those of its neighbors. Moreover, the above studies on RL have been conducted in continuous time, which restricts the use of data-driven decomposition. Herein, large-scale system dynamics are formulated in discrete-time

as lumped-state dynamics with a defined global performance function that enables interactions among subsystems. This interdependence characteristic makes the system more complex and challenging to analyze.

This paper proposes an off-policy RL approach in discrete-time for the large-scale linear quadratic regulator (LQR) control problem by decomposing it into a lower-dimensional one. The off-policy approach uses two policies: one is used to generate data, namely the behavior policy, while the other is used to evaluate and improve the policy [154]. The work in [21, 140] discuss LQR design for a large-scale network of homogeneous dynamical systems. One could develop a lower-dimensional mapping of the original observations and, then, study its temporal dynamics. Data obtained over space and time could provide more relevant dynamic information compared to those obtained using just spatial data.

We use a data-driven strategy to characterize complex system dynamics having high spatial dimensionality in discrete time. Employing RL could result in limited accuracy due to potentially near-singular rank matrices and longer run times for high-dimensional systems. Dynamic mode decomposition (DMD) is a computationally viable structure to analyze spatial-temporal data that can be depicted as dynamical model realizations [111, 125]. This approach does not rely on the system model, making it appropriate for model-free RL adaptation. When singular values closer to zero are preserved, singular value decomposition (SVD)-based approaches for order reduction could involve near-singular matrices. [36, 38] show thresholding techniques and [41] shows optimal thresholding. The truncation step in SVD-based DMD selects a relatively small threshold, and sets all eigenvalues below this threshold to zero.

Furthermore, we show that DMD can extract spatio-temporal coherent patterns from data. These patterns are called essential modes that oscillate at fixed natural frequencies. Using these modes can help extract exact behaviors of the underlying original high-dimensional system into lower ones [25]. The discrete-time RL algorithm is then designed

using this truncated model. This significantly reduces the computational complexity of the discrete-time RL algorithm for optimal control learning.

The prime motivation of this paper is to illustrate that dynamic decomposition and RL can be integrated to provide a computationally-tractable optimal control scheme suitable for large-scale networks. Salient contributions of this paper are summarized as follows:

1. For a large-scale system, we formulate the LQR graphical problem in discrete time. A large-scale system is constructed by assembling linear subsystems, defined for both coupled and decoupled dynamical systems. The stabilizing controller is specified.
2. We show that the model-free discrete-time RL algorithm scales poorly when solving large-scale LQR graphical problems.
3. We develop a computationally efficient discrete-time RL algorithm based on DMD. The algorithm reduces data dimensions needed for optimal control learning while retaining the dynamic information of the original system.
4. We show the efficiency of the proposed algorithm in both theoretical and numerical analysis.

This paper is organized as follows: Section II gives preliminaries and notations. Section III introduces the large-scale LQR graphical problem, and provides a model-free discrete-time RL algorithm. Section IV presents a model-free discrete-time RL algorithm with DMD-preconditioning. Section V offers case studies. The conclusion is drawn in Section VI.

3.2 Notations and Preliminaries

Notations: The n-dimensional Euclidean space is denoted by $\mathbb{R}^n$. $\otimes$ stands for the Kronecker product. I_n indicates the n-dimensional identity matrix. $|\cdot|$ defines the absolute-

value norm of a vector. $\|\cdot\|_F$ indicates the Frobenius norm of a vector or a matrix. Y^ι stands for the complex conjugate transpose of Y. $Y^\dagger$ denotes pseudoinverse of Y matrix. $S(L) = \{\lambda_1(L), \ldots, \lambda_M(L)\}$ denotes spectrum of matrix L where λ_i is the *i-th* eigenvalue. For vectors a, b and matrix W, $a^T W b = (b^T \otimes a^T) vec(W)$. For a matrix $L = [l_{ij}] \in \mathbb{R}^{n \times n}$, $\text{vec}(L) = [l_{11}, l_{12}, \ldots, l_{1n}, l_{21}, \ldots, l_{2n}, \ldots, l_{nn}]^T \in \mathbb{R}^{n^2}$.

Graph Preliminaries: $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ denotes a graph topology with $\mathscr{M} \cong (1, 2, \ldots, M)$ vertices where $v_i \in \mathscr{V}$, $\forall i \in \mathscr{M}$. The edges $\mathscr{E} \subseteq \mathscr{V} \times \mathscr{V}$ are between the two vertices with the connectivity weight $e_{ij} > 0$ if $(v_j, v_i) \in \mathscr{E}$; Otherwise, $e_{ij} = 0$. The set of neighbors of vertex v_i is $\mathscr{M}_i \cong \{v_j : e_{ij} > 0\}$. We assume no self-loops in the graph, i.e., $e_{ii} = 0$. The weighted in-degree of vertex i is $d_i^i = \sum_{j=1}^{\mathscr{M}} e_{ij}$, and the weighted out-degree of vertex i is $d_i^o = \sum_{j=1}^{\mathscr{M}} e_{ji}$. Progression of edges $\mathscr{E}$ through vertices $(v_{i_{s-1}}, v_{i_s}) \in \mathscr{E}$ for $s \in (2, \ldots, j)$ constitutes a directed path originating from v_{i_1} to v_{i_j}. A graph is considered balanced and bi-directional where $e_{ij} = e_{ji}, d_i^i = d_i^o, \forall i, j$, i.e., undirected topology where directed path is present between every pair of vertices v_i and v_j. Laplacian matrix $\mathscr{L}$ is defined as $\mathscr{D} - \mathscr{A}$, where $\mathscr{A} = [e_{ij}], \mathscr{A} = \mathscr{A}^{\mathrm{T}}$, is the graph adjacency matrix, and $\mathscr{D} = diag\{d_i\}$ is the graph degree matrix.

3.3 Large-Scale LQR Graphical Problem

This section introduces a discrete-time LQR graphical problem for a large-scale dynamical system as a network of linear subsystems. The formulation of a large-scale system considers both decoupled and coupled fashions. Then, we propose an off-policy RL algorithm to compute the optimal control solution of large-scale systems.

3.3.0.1 Decoupled Systems

Consider a network having dynamically decoupled discrete-time linear subsystems

$$x_{i_{k+1}} = A_i x_{i_k} + B_i u_{i_k}, \quad i \in \mathscr{M}, \tag{3.1}$$

where $x_{i_k} \in \mathbb{R}^n$ and $u_{i_k} \in \mathbb{R}^m$ denote the state and control input of subsystem i, respectively. Each subsystem has local identical matrices $A_i = A_1 \in \mathbb{R}^{n\times n}$ and $B_i = B \in \mathbb{R}^{n\times m}$. (A_1, B) is assumed to be stabilizable. Note that subsystems are dynamically decoupled, but with a common objective, i.e., coupled through a performance function.

3.3.0.2 Coupled Systems

Consider a network having dynamically coupled discrete-time linear subsystems. The *i-th* subsystem dynamics, at the local level, is

$$x_{i_{k+1}} = A_{ii} x_{i_k} + A_{ij} \sum_{j\neq i, j=1}^{M} e_{ij}(x_{i_k} - x_{j_k}) + B_i u_{i_k}, \quad i \in \mathscr{M}, \tag{3.2}$$

where $x_{i_k} \in \mathbb{R}^n$ and $u_{i_k} \in \mathbb{R}^m$ denote subsystems state and control input, respectively. We assume that the network topology of the coupled system for both physical couplings and communication between subsystems coincide, and are defined by the same graphical topology $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ with the Laplacian matrix $\mathscr{L} \in \mathbb{R}^{M\times M}$. The subsystems $i \in \mathscr{V}$ relates to the local state x_{i_k}, whereas edges $(i, j) \in \mathscr{E}$ between subsystems relates to the $(x_{i_k} - x_{j_k})$. Each subsystem has identical matrices as $A_{ii} = A_1 \in \mathbb{R}^{n\times n}$, $A_{ij} = A_2 \in \mathbb{R}^{n\times n}$, and $B_i = B \in \mathbb{R}^{n\times m}$. The pairs (A_1, B) and $(A_1 + MA_2, B)$ are assumed to be stabilizable. Note that we are considering dynamically coupled M subsystems having a mutual performance function.

Define global states $\tilde{x} = (x_1^{\mathrm{T}}, \ldots, x_M^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{nM}$ and $\tilde{u} = (u_1^{\mathrm{T}}, \ldots, u_M^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{mM}$. The global dynamics of (3.2) and (3.1) are

$$\tilde{x}_{k+1} = \tilde{A}\tilde{x}_k + \tilde{B}\tilde{u}_k, \tag{3.3}$$

where global state and control matrices are $\tilde{A} = I_M \otimes A_1 \in \mathbb{R}^{nM \times nM}$ and $\tilde{B} = I_M \otimes B \in \mathbb{R}^{nM \times mM}$ for the decoupled system. Similarly, for the coupled system, the global state and control matrices are $\tilde{A} = (I_M \otimes A_1 + \mathscr{L} \otimes A_2) \in \mathbb{R}^{nM \times nM}$ and $\tilde{B} = I_M \otimes B \in \mathbb{R}^{nM \times mM}$.

3.3.1 Large-scale LQR Problem

The performance function couples the dynamic behavior for the set $\mathscr{M}$ of subsystems as

$$J(\tilde{u}_k, \tilde{x}_k) = \sum_{k=0}^{\infty} \Bigg[\sum_{i=1}^{M} x_{i_k}^{\mathrm{T}} Q_1 x_{i_k} + u_{i_k}^{\mathrm{T}} R_{ii} u_{i_k} + \sum_{i=1}^{M} \sum_{j \neq i}^{M} (x_{i_k} - x_{j_k})^{\mathrm{T}} Q_2 (x_{i_k} - x_{j_k}) \Bigg], \tag{3.4}$$

where $R_{ii} = R_{ii}^{\mathrm{T}} = R > 0$, $Q_1 = Q_1^{\mathrm{T}} \geq 0, \forall i$, and $Q_2 = Q_2^{\mathrm{T}} \geq 0, \forall i \neq j$. The global form of the performance function, J, which integrates the behavior of all subsystems, is

$$J(\tilde{u}_k, \tilde{x}_k) = \sum_{k=0}^{\infty} \left[\tilde{x}_k^{\mathrm{T}} \tilde{Q} \tilde{x}_k + \tilde{u}_k^{\mathrm{T}} \tilde{R} \tilde{u}_k \right]. \tag{3.5}$$

Herein, $\tilde{Q} = (I_M \otimes Q_1 + \mathscr{L} \otimes Q_2) \in \mathbb{R}^{nM \times nM}$ and $\tilde{R} = I_M \otimes R \in \mathbb{R}^{mM \times mM}$, given by

$$\tilde{R} = \begin{bmatrix} R & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & \dots & R \end{bmatrix}, \tilde{Q} = \begin{bmatrix} \tilde{Q}_{11} & \tilde{Q}_{12} & \dots & \tilde{Q}_{1M} \\ \vdots & \ddots & \vdots & \vdots \\ \tilde{Q}_{M1} & \dots & \dots & \tilde{Q}_{MM} \end{bmatrix},$$

where

$$\tilde{Q}_{ii} = Q_1 + (M-1)Q_2, \quad i \in \mathscr{M}, \tag{3.6a}$$

$$\tilde{Q}_{ij} = -Q_2, \quad i, j \in \mathscr{M}, i \neq j. \tag{3.6b}$$

Remark 10. Here, $Q_1, Q_2 \in \mathbb{R}^{n \times n}$ penalize local and relative state difference between vertices $i, j \in V$, respectively, with identical weights. Given (3.6), it is evident that $|\tilde{Q}_{ii}| > \sum_{i \neq j} |\tilde{Q}_{ij}|, \forall i$. This makes a matrix $\tilde{Q}$ strictly diagonally-dominant, and given $Q_1, Q_2 \geq 0$, $\tilde{Q} \geq 0$. Also, $\tilde{R} > 0$ is a block-diagonal matrix, where $R > 0 \in \mathbb{R}^{m \times m}$.

LQR Graphical Problem: Given Remark 10, find a unique stabilizing control policy $\tilde{u}_k$ that minimizes the global $J(\tilde{u}_k, \tilde{x}_k)$ in (3.5) as

$$J^*(\tilde{x}_k) = \min_{\tilde{u}_k} J(\tilde{u}_k, \tilde{x}_k) = \min_{\tilde{u}_k} \sum_{k=0}^{\infty} \left[\tilde{x}_k^{\mathrm{T}} \tilde{Q} \tilde{x}_k + \tilde{u}_k^{\mathrm{T}} \tilde{R} \tilde{u}_k\right]$$
$$= \tilde{x}_k^{\mathrm{T}} \tilde{P} \tilde{x}_k, \tag{3.7}$$

where $\tilde{P} = \tilde{P}^{\mathrm{T}} \in \mathbb{R}^{nM \times nM} > 0$ is a cost matrix.

The optimal control policy $\tilde{u}_k^*$ from [79] is given as

$$\tilde{u}_k^* = -\tilde{K}^* \tilde{x}_k, \tag{3.8}$$

where $\tilde{K} \in \mathbb{R}^{mM \times nM}$ is

$$\tilde{K}^* = (\tilde{R} + \tilde{B}^{\mathrm{T}} \tilde{P} \tilde{B})^{-1} \tilde{B}^{\mathrm{T}} \tilde{P} \tilde{A}, \tag{3.9}$$

and $\tilde{P}$ satisfies the Algebric Riccati Equation (ARE)

$$\tilde{P} = \tilde{A}^{\mathrm{T}} \tilde{P} \tilde{A} - \tilde{A}^{\mathrm{T}} \tilde{P} \tilde{B} (\tilde{R} + \tilde{B}^{\mathrm{T}} \tilde{P} \tilde{B})^{-1} \tilde{B}^{\mathrm{T}} \tilde{P} \tilde{A} + \tilde{Q}. \tag{3.10}$$

In the following Theorem 6 and Corollary 1, we prove that $\tilde{P}$ in (3.10) has a certain structure due to the global formulation of large-scale dynamics and performance function. This is important for the design of stabilizing distributed controllers. We rewrite (3.10) in an equivalent form, and set $\tilde{B}\tilde{R}^{-1}\tilde{B}^{\mathrm{T}} = H$ for simplification,

$$\tilde{Q} - \tilde{A}^{\mathrm{T}} \tilde{P} H \tilde{P} \tilde{A} - \tilde{P} = 0. \tag{3.11}$$

For decoupled and coupled large-scale systems, diagonal blocks of $\tilde{A}$ are $\tilde{A}_{ii} = A_1$ and $\tilde{A}_{ii} = A_1 + (M-1)A_2$, respectively.

Theorem 6. *Consider the discrete-time ARE in* (3.10)*, with a symmetric $\tilde{P}$ for large-scale system dynamics in* (3.3)*, to solve the LQR graphical problem in* (3.7)*. Then, certain matrix $W_{ii} = \sum_{h=1}^{M} \tilde{p}_{ih} = P_1, i \in \mathscr{M}$, where P_1 satisfies*

$$A_1^T P_1 A_1 - A_1^T P_1 B (R + B^T P_1 B)^{-1} B^T P_1 A_1 + Q_1 - P_1 = 0. \tag{3.12}$$

Proof. $\tilde{P}$ in (3.10) is symmetric, i.e., $\tilde{P}_{ij} = \tilde{P}_{ji}$. Then, let a certain matrix W_{ii} as

$$W_{ii} = \tilde{P}_{ii} + \sum_{j\neq i, j=1}^{M} \tilde{P}_{ij}, i \in \mathscr{M}. \tag{3.13}$$

For the diagonal blocks of $\tilde{P}$, i.e., $\tilde{P}_{ii}$ in (3.10), we have

$$\tilde{Q}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1}^{M} (\tilde{P}_{ih} H_{ii} \tilde{P}_{ih}) \tilde{A}_{ii} - \tilde{P}_{ii} = 0. \tag{3.14}$$

Substitute $\tilde{P}_{ii} = W_{ii} - \sum_{j\neq i, j=1}^{M} \tilde{P}_{ij}$ in (3.14) to yield

$$\begin{aligned}
&\tilde{Q}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1, h\neq i}^{M} (\tilde{P}_{ih} H_{ii} \tilde{P}_{ih}) \tilde{A}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} W_{ii} H_{ii} W_{ii} \tilde{A}_{ii} \\
&- \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1, h\neq i}^{M} \tilde{P}_{ih} H_{ii} \sum_{l=1, l\neq i}^{M} \tilde{P}_{il} \tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1, h\neq i}^{M} \tilde{P}_{ih} H_{ii} W_{ii} \tilde{A}_{ii} \\
&+ \tilde{A}_{ii}^{\mathrm{T}} W_{ii} H_{ii} \sum_{h=1, h\neq i}^{M} \tilde{P}_{ih} \tilde{A}_{ii} - W_{ii} + \sum_{j=1, j\neq i}^{M} \tilde{P}_{ij} = 0.
\end{aligned} \tag{3.15}$$

For the off-diagonal blocks of $\tilde{P}$, i.e., $\tilde{P}_{ij}$ in (3.10), we have

$$\tilde{Q}_{ij} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1}^{M} (\tilde{P}_{ih} H_{ii} \tilde{P}_{hj}) \tilde{A}_{ii} - \tilde{P}_{ij} = 0. \tag{3.16}$$

Substituting (3.13) in (3.16) leads to

$$\begin{aligned}
&\tilde{Q}_{ij} - \tilde{A}_{ii}^{\mathrm{T}} W_{ii} H_{ii} \tilde{P}_{ij} \tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \Big(\sum_{h=1, h\neq i}^{M} \tilde{P}_{ih} \Big) H_{ii} \tilde{P}_{ij} \tilde{A}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \tilde{P}_{ij} H_{ii} W_{jj} \tilde{A}_{ii} \\
&+ \tilde{A}_{ii}^{\mathrm{T}} \tilde{P}_{ij} H_{ii} \Big(\sum_{h=1, h\neq j}^{M} \tilde{P}_{jh} \Big) \tilde{A}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1, h\neq i, j}^{M} (\tilde{P}_{ih} H_{ii} \tilde{P}_{hj}) \tilde{A}_{ii} - \tilde{P}_{ij} = 0.
\end{aligned} \tag{3.17}$$

For $j \neq i$, summing up (3.17) relates to $\sum_{h=1,h\neq i}^{M} \tilde{P}_{ih}$, i.e., the summation of the off-diagonal terms leads to

$$\begin{aligned}
&\sum_{h=1,h\neq i}^{M} \tilde{Q}_{ih} - \tilde{A}_{ii}^{\mathrm{T}} W_{ii} H_{ii} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih}\tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \sum_{l=1,l\neq i}^{M} \Big(\Big(\sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} \Big) \\
&\times H_{ii}\tilde{P}_{il} \Big) \tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \Big(\tilde{P}_{ih} H \Big(\sum_{l=1,l\neq h}^{M} \tilde{P}_{hl} \Big) \Big) \tilde{A}_{ii} \\
&- \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \Big(\sum_{l,h=1,l\neq i}^{M} \Big(\tilde{P}_{ih} H_{ii} \tilde{P}_{il} \Big) \Big) \tilde{A}_{ii} \\
&- \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} H_{ii} W_{hh} \tilde{A}_{ii} - \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} = 0.
\end{aligned} \tag{3.18}$$

Note that

$$\begin{aligned}
&\tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \Big(\tilde{P}_{ih} H_{ii} \Big(\sum_{l=1,l\neq h}^{M} \tilde{P}_{hl} \Big) \Big) \tilde{A}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \Big(\sum_{l,h=1,l\neq i}^{M} \Big(\tilde{P}_{ih} \\
&\times H_{ii} \tilde{P}_{il} \Big) \Big) \tilde{A}_{ii} = \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \sum_{l=1,l\neq h}^{M} \tilde{P}_{ih} H_{ii} \tilde{P}_{hl} \tilde{A}_{ii} \\
&- \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \sum_{l=1,l\neq i,h}^{M} \tilde{P}_{ih} H_{ii} \tilde{P}_{hl} \tilde{A}_{ii} = \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} H_{ii} \tilde{P}_{hi} \tilde{A}_{ii}.
\end{aligned} \tag{3.19}$$

Substituting (3.19) in (3.18) yields

$$\begin{aligned}
&\sum_{h=1,h\neq i}^{M} \tilde{Q}_{ih} - \tilde{A}_{ii}^{\mathrm{T}} W_{ii} H_{ii} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih}\tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \sum_{l=1,l\neq i}^{M} \Big(\Big(\sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} \Big) \\
&\times H_{ii}\tilde{P}_{il} \Big) \tilde{A}_{ii} + \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} H_{ii} \tilde{P}_{hi} \tilde{A}_{ii} - \tilde{A}_{ii}^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} H_{ii} W_{hh} \tilde{A}_{ii} \\
&- \sum_{h=1,h\neq i}^{M} \tilde{P}_{ih} = 0.
\end{aligned} \tag{3.20}$$

Adding (3.20) to (3.15) with (3.6) gives

$$Q_1 - A_1^{\mathrm{T}} W_{ii} H_{ii} W_{ii} A_1 + A_1^{\mathrm{T}} \sum_{h=1,h\neq i}^{M} (\tilde{P}_{ih} H_{ii} (W_{ii} - W_{hh}) A_1 - W_{ii} = 0. \tag{3.21}$$

Summing up (3.21) over $i \in \mathscr{M}$ results in

$$\sum_{i=1}^{M} \left[Q_1 - A_1^{\mathrm{T}} W_{ii} H_{ii} W_{ii} A_1 - W_{ii} \right] = 0, \tag{3.22}$$

which is equivalent to

$$\sum_{i=1}^{M}\left[A_1^{\mathrm{T}}W_{ii}A_1 - A_1^{\mathrm{T}}W_{ii}\tilde{B}(\tilde{R}+\tilde{B}^{\mathrm{T}}W_{ii}\tilde{B})^{-1}\tilde{B}^{\mathrm{T}}W_{ii}A_1 + Q_1 - W_{ii}\right] = 0. \tag{3.23}$$

Given $\tilde{A}$ and $\tilde{B}$ as block diagonal matrices, having identical blocks, implies that discrete ARE in (3.10) is a set of M identical discrete AREs. Every $W_{ii} = \sum_{h=1}^{M}\tilde{p}_{ih}$ in (3.13) is identical, and (3.23) is a set of M identical discrete AREs

$$M(A_1^{\mathrm{T}}W_{ii}A_1 - A_1^{\mathrm{T}}W_{ii}\tilde{B}(\tilde{R}+\tilde{B}^{\mathrm{T}}W_{ii}\tilde{B})^{-1}\tilde{B}^{\mathrm{T}}W_{ii}A_1 + Q_1 - W_{ii}) = 0. \tag{3.24}$$

Putting $W_{ii} = P_1$ in (3.24) gives (3.12). This finishes the proof. □

Corollary 1. Given Theorem 6, the control policy (3.8) for the solution to (3.7) gives $\tilde{P}$ in (3.10), and can be divided into M^2 blocks of $\mathbb{R}^{n\times n}$, denoted by $\tilde{P}_{ih}$. For $i,j,h \in \mathscr{M}$, the followings hold

1. Diagonal blocks of $\tilde{P}$, i.e., $\tilde{P}_{ii}$ is $P_1 - (M-1)P_2$, where P_1 is solution of (3.12).
2. Off-diagonal blocks of $\tilde{P}$, i.e., $\tilde{P}_{ij}$, $i \neq j$, is P_2, where P_2 is associated with

$$\begin{aligned}&\bar{A}^{\mathrm{T}}(P_1 - MP_2)\bar{A} - \bar{A}^{\mathrm{T}}(P_1 - MP_2)B(R + B^{\mathrm{T}}(P_1 - MP_2)B)^{-1}\\ &\times B^{\mathrm{T}}(P_1 - MP_2)\bar{A} + (Q_1 + MQ_2) = (P_1 - MP_2).\end{aligned} \tag{3.25}$$

   Note that for decoupled and coupled large-scale systems, $\bar{A} = A_1$ are $\bar{A} = A_1 + MA_2$, respectively.

3. $\tilde{P} > 0$ is the solution to discrete ARE in (3.10) such that

$$\tilde{x}_k^{\mathrm{T}}\tilde{P}\tilde{x}_k = \tilde{x}_k^{\mathrm{T}}P_1\tilde{x}_k + \sum_{i=1}^{M}\sum_{j\neq i}^{M}(x_{ik} - x_{jk})^{\mathrm{T}}P_2(x_{ik} - x_{jk}). \tag{3.26}$$

   Herein, $\tilde{P} = (I_M \otimes P_1 - \mathscr{L} \otimes P_2) \in \mathbb{R}^{nM\times nM}$ is given by

$$\tilde{P} = \begin{bmatrix} P_1 - (M-1)P_2 & P_2 & \dots & P_2 \\ \vdots & \ddots & \vdots & \vdots \\ P_2 & \dots & \dots & P_1 - (M-1)P_2. \end{bmatrix}. \tag{3.27}$$

Remark 11. Note that the subsystem's ability to achieve the mutual goal, which is interconnected in the network, is generalized in (3.3). Therein, each subsystem has independent actuation. This formulation is standard for networked control systems [21, 94, 140], where subsystems are dynamically coupled or decoupled and have mutual performance function. It yields a more general optimal control framework as in (3.3) and (3.5) for both coupled and decoupled network systems.

Remark 12. Stabilizing distributed controllers for the global dynamics in (3.3) of (3.1) and (3.2) are constructed in (3.9), where $\tilde{P}$ in (3.9) and (3.10) has the structure given in (3.27).

In the following subsection, solutions of the LQR graphical problem, i.e., $(\tilde{P}, \tilde{K}^*)$, are computed using RL.

3.3.2 Off-Policy Discrete-time RL Algorithm

We present the model-free, off-policy discrete-time RL algorithm based on [96, 154] to solve the LQR graphical problem in Algorithm 4. The system (3.3) is rewritten as

$$\tilde{x}_{k+1} = \tilde{A}_j \tilde{x}_k + \tilde{B}(\tilde{K}^j \tilde{x}_k + \tilde{u}_k), \tag{3.28}$$

where $\tilde{A}_j = \tilde{A} - \tilde{B}\tilde{K}^j$. The fixed policy $\tilde{u}_k^j = -\tilde{K}^j \tilde{x}_k$, learned and updated by policy iteration [61], is applied to (3.3) to generate the behavioral trajectory data used in learning. The off-policy Bellman equation can be expressed as

$$\begin{aligned}
&(\tilde{x}_k^{\mathrm{T}} \otimes \tilde{x}_k^{\mathrm{T}})\mathrm{vec}(\tilde{P}^{j+1}) - (\tilde{x}_{k+1}^{\mathrm{T}} \otimes \tilde{x}_{k+1}^{\mathrm{T}})\mathrm{vec}(\tilde{P}^{j+1}) \\
&+ 2(\tilde{x}_k^{\mathrm{T}} \otimes (\tilde{u}_k + \tilde{K}^j \tilde{x}_k)^{\mathrm{T}})\mathrm{vec}(\tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{A}) \\
&- ((\tilde{K}^j \tilde{x}_k - \tilde{u}_k)^{\mathrm{T}} \otimes (\tilde{u}_k + \tilde{K}^j \tilde{x}_k)^{\mathrm{T}})\mathrm{vec}(\tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{B}) \\
&= \tilde{x}_k^{\mathrm{T}} \tilde{Q} \tilde{x}_k + \tilde{x}_k^{\mathrm{T}} (\tilde{K}^j)^{\mathrm{T}} \tilde{R} \tilde{K}^j \tilde{x}_k.
\end{aligned} \tag{3.29}$$

Next, we present a data-driven implementation of model-free RL in Algorithm 4 to solve (3.29) for $(\tilde{P}, \tilde{K}^*)$.

Algorithm 4 Data-driven Implementation of the Model-free RL Algorithm for a Large-scale System

1: **Initialization:** Given N, set $j = 0$ and small threshold e. Select stabilizing $\tilde{K}^0$ and control input $\tilde{u}_k = -\tilde{K}\tilde{x}_k + \varepsilon_k$, where ε_k is a probing noise.

2: **Data Collection**: For $j = 0, 1, 2, \ldots, N-1$, collect $\tilde{x}_k$ under a given $\tilde{u}_k$.

3: Compute (φ^j, ψ^j) defined as

$$\varphi^j = \begin{bmatrix} \tilde{x}_k^{\mathrm{T}} \tilde{Q} \tilde{x}_k + \tilde{x}_k^{\mathrm{T}} (\tilde{K}^j)^{\mathrm{T}} \tilde{R} \tilde{K}^j \tilde{x}_k \\ \vdots \\ \tilde{x}_{k+\zeta-1}^{\mathrm{T}} \tilde{Q} \tilde{x}_{k+\zeta-1} + \tilde{x}_{k+\zeta-1}^{\mathrm{T}} (\tilde{K}^j)^{\mathrm{T}} \tilde{R} \tilde{K}^j \tilde{x}_{k+\zeta-1} \end{bmatrix}, \tag{3.30}$$

$$\psi^j = \begin{bmatrix} \rho_{xx1} & \rho_{xu1} & \rho_{uu1} \\ \vdots & \vdots & \vdots \\ \rho_{xx\zeta} & \rho_{xu\zeta} & \rho_{uu\zeta} \end{bmatrix}, \tag{3.31}$$

where $s = 1, 2, \cdots, \zeta$, and

$$\rho_{xx_s} = (\tilde{x}_{k+s-1}^{\mathrm{T}} \otimes \tilde{x}_{k+s-1}^{\mathrm{T}}) - (\tilde{x}_{k+s}^{\mathrm{T}} \otimes \tilde{x}_{k+s}^{\mathrm{T}}), \tag{3.32a}$$

$$\rho_{xu_s} = 2(\tilde{x}_{k+s-1}^{\mathrm{T}} \otimes (\tilde{u}_{k+s-1} + \tilde{K}^j \tilde{x}_{k+s-1})^{\mathrm{T}}), \tag{3.32b}$$

$$\rho_{uu_s} = -(\tilde{K}^j \tilde{x}_{k+s-1} - \tilde{u}_{k+s-1})^{\mathrm{T}} \otimes (\tilde{u}_{k+s-1} + \tilde{K}^j \tilde{x}_{k+s-1})^{\mathrm{T}}. \tag{3.32c}$$

4: **Policy Evaluation**: Compute $\tilde{P}^{j+1}$ by

$$((\psi^j)^{\mathrm{T}} \psi^j)^{-1} (\psi^j)^{\mathrm{T}} \varphi^j = \begin{bmatrix} \mathrm{vec}(\tilde{P}^{j+1})^{\mathrm{T}} \\ \mathrm{vec}(\tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{A})^{\mathrm{T}} \\ \mathrm{vec}(\tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{B})^{\mathrm{T}} \end{bmatrix}. \tag{3.33}$$

5: **Policy Improvement**: Compute $\tilde{K}^{j+1}$ by

$$\tilde{K}^{j+1} = \left(\tilde{R} + \tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{B}\right)^{-1} \tilde{B}^{\mathrm{T}} \tilde{P}^{j+1} \tilde{A}. \tag{3.34}$$

6: **Stop if** $\|\tilde{K}^{j+1} - \tilde{K}^{j}\| \leq e$; Otherwise, set $j = j+1$ and go to step 2.

Note that in (3.32), $\rho_{xx_s} \in \mathbb{R}^{1\times(nM)^2}, \rho_{xu_s} \in \mathbb{R}^{1\times nmM^2}$, and $\rho_{uu_s} \in \mathbb{R}^{1\times(mM)^2}$. Thus, (3.33) has $d = (nM)^2 + (mM)^2 + nmM^2$ unknown parameters. The least square (LS) solution to (3.33) needs a full rank of $((\psi^j)^{\mathrm{T}}\psi^j)$ and, at a minimum, needs $\zeta \geq d$ samples at each iteration. Using solutions from (3.33), the feedback gain $\tilde{K}^{j+1}$ can be acquired by (3.34).

Remark 13. The terms containing system dynamics (A, B) are regarded as unknowns and are solved in (3.33) given measured data in (3.30)-(3.32) by using the LS method. Note that $\tilde{x}_k$ in (3.30)-(3.32) is collected given $\tilde{u}_k$ as shown in Step 2 of Algorithm 4. $\tilde{P}^{j+1}$ is also solved in (21). $\tilde{K}^{j+1}$ is then updated using the solution of (21). This makes the approach model-free.

Remark 14. Given the large-scale networked system in (3.3), consider Algorithm 4 for the LQR graphical problem (3.7). At each iteration j, collected operators φ^j and ψ^j in (3.30) and (3.31), where $\tilde{x}_k$ is collected given $\tilde{u}_k$, give $\tilde{P}^{j+1}$ in (3.33). As seen in [69], a unique solution $(\tilde{P}^{j+1}, \tilde{K}^{j+1})$ is obtained using an off-policy algorithm when probing noises are included in the control input for the persistence of excitation. The pair $(\tilde{P}^{j+1}, \tilde{K}^{j+1})$ can be uniquely solved by LS while satisfying the full-rank condition of ψ^j. Then, Algorithm 4 converges to the optimal solution (3.9).

Remark 15. In Algorithm 4, LS requires $(n^2 + m^2 + nm)M^2$ data samples to obtain a unique solution. This requirement scales by M^2 over samples needed for a single subsystem. Thus, for a large M, RL Algorithm 4 scales poorly when finding optimal control using policy

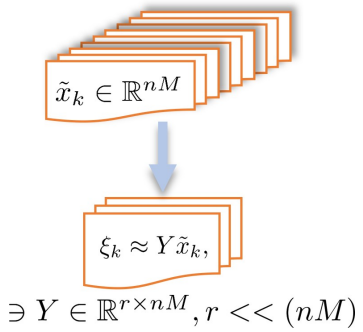


Figure 3.1. Dimensionality Reduction.

iteration. The learning time, i.e., the time required to execute steps 3 and 4 of Algorithm 1, could restrict its use in real-time control.

3.4 DMD-based Off-Policy RL

In this section, we introduce an alternative formulation for the RL algorithm to solve the problem given in Remark 15. We use the DMD method to reduce the learning time by formulating a lower-dimensional dynamic model.

Given the state measurement $\tilde{x}_k \in \mathbb{R}^{nM}$, discretized in space and collected in step 2 of Algorithm 4, a lower-dimensional state measurement ξ_k is acquired by projecting $\tilde{x}_k$ through a projection matrix Y

$$\xi_k \approx Y\tilde{x}_k, \tag{3.35}$$

where $Y \in \mathbb{R}^{r\times nM}$, $r << (nM)$, represented in Figure 3.1.

The following lemma shows that the optimal controller $\tilde{u}_k^*$ in (3.8) is learned by ξ_k rather than $\tilde{x}_k$ so that the performance function associated with lower-dimensional system model is near to the global J in (3.5).

Lemma 1. See [136]. Given system dynamics (3.3) and structure of cost matrix in (3.27), ξ_k in (3.35) satisfies

$$\xi_{k+1} = Y\tilde{A}Y^{\dagger}\xi_k + Y\tilde{B}\tilde{u}_k, \tag{3.36}$$

$$\bar{J}(\tilde{u}_k, \xi_k) = \sum_{k=0}^{\infty} \left[\xi_k^{\mathrm{T}} \tilde{Q}_r \xi_k + \tilde{u}_k^{\mathrm{T}} R \tilde{u}_k \right], \tag{3.37}$$

where $Y\tilde{A}Y^{\dagger} \in \mathbb{R}^{r\times r}$ is Hurwitz,

$$\tilde{x_k} \approx Y^{\dagger} \xi_k \tag{3.38}$$

holds for any $\tilde{u}_k, \tilde{x}_k$ and $\tilde{Q}_r = (Y^{\dagger})^{\mathrm{T}} \tilde{Q} Y^{\dagger} \geq 0 \in \mathbb{R}^{r\times r}$.

Proof. This Lemma implies that the r-dimensional state ξ_k retains the behavior of nM-dimensional state $\tilde{x}_k$, if Y satisfies Remark 17. From (3.38) and (3.5), the performance function with respect to the truncated dimension state ξ_k becomes (3.37).

Given structure of $\tilde{P} > 0$ in Corollary 1, the truncated cost matrix $\tilde{P}_r$ spans analogous eigenvalues as $\tilde{P}$, i.e., $S(\tilde{P}_r) = S(Y\tilde{P}Y^{\dagger})$. The spectrum of $\tilde{P}$ [21] is

$$\begin{aligned} S(\tilde{P}) &= S(I_M \otimes P_1 - \mathscr{L} \otimes P_2) \\ &= \bigcup_{i \in \mathscr{M}} S(P_1 - \lambda_i(\mathscr{L}) P_2), \end{aligned} \tag{3.39}$$

where $\lambda_i(\mathscr{L}) \in S(\mathscr{L})$. Then, the optimal control learning for nM-dimensional system in (3.3), with global performance function J, is analogous to learning for the truncated system (3.36) with the performance function in (3.46). This finishes the proof. □

The balanced truncation [97] can build Y, but it is infeasible for larger systems that require a solution to high-dimensional Lyapunov equations [135, 136] in the computation of discrete controllability Gramian $\Phi_c = \sum_{m=0}^{\infty} \tilde{A}^m \tilde{B} \tilde{B}^{\top} (\tilde{A}^{\top})^m$. Balanced proper orthogonal decomposition (POD) [116] approximates balanced truncation and avoids the computation of Φ_c. This makes the computation of a projection matrix Y tractable for larger systems.

Remark 16. Since the system model is unknown for the model-free algorithm, the matrix Y in (3.35) will only be built using state measurements $\tilde{x}_k$. The matrix Y indicates the measure of insufficiency for the system (3.3) to be controlled and reduces the system dimension.

Remark 17. If Y in (3.35) satisfies controllability Gramian then $(I-Y^{\dagger}Y)\tilde{x}_k \approx 0$ holds for any $\tilde{u}_k, \tilde{x}_k$ [67]. Then, matrix Y can be found from $\min_{Y\in\mathbb{R}^{r\times nM}} \sum_{j=0}^{N-1} \left\|(I-Y^{\dagger}Y)\tilde{x}_k\right\|^2$. This projection, as shown in [67], makes (3.35) hold, where $Y^{\dagger}$ is specified by the first r columns of the left-singular matrix.

Eigensystem realization algorithm in [62] is similar to balanced POD [116, 147] which is suitable for high dimensional systems. The POD and SVD can identify only the spatial patterns, whereas DMD can identify spatiotemporal patterns in $\tilde{x}_k$ [76]. Thus, firstly, DMD can be used for the balanced POD to obtain Y as shown in Remark 17, where $Y^{\dagger} = \tilde{U}$. Secondly, DMD enables the extraction of dynamic modes from collected $\tilde{x}_k$. These dynamic modes are used to precondition the data vectors in (3.32) of the underlying large-scale system (3.3) onto a lower-dimensional space. This retains the complete dynamic information of (3.3) in lower dimensions.

3.4.1 Build Y in (23) and Extract Dynamic Modes via DMD

This section presents the data-driven approach to build Y in (23) for the system (3.3) while satisfying Remark 16 and extracting DMD modes. DMD is a data-driven method [76] that examines the relationship between measurements of a dynamical system correlated by a linear operator as

$$\tilde{x}_{k+1} = T\tilde{x}_k. \tag{3.40}$$

To obtain the data-driven model of (3.3) which is the same as in (3.40), $\tilde{x}_k$ is collected given $\tilde{u}_k$ in step 2 of Algorithm 4, where $T = \tilde{A} - \tilde{B}\tilde{K}$ and $T \in \mathbb{R}^{nM\times nM}$. Given $\tilde{x}_k$, $(X_1, X_2) \in \mathbb{R}^{nM\times(p-1)}$ is arranged such that X_2 is the time-shifted matrix of X_1

$$X_1 = \begin{bmatrix} | & | & \dots & | \\ x_1 & x_2 & \dots & x_{p-1} \\ | & | & \dots & | \end{bmatrix}, \quad X_2 = \begin{bmatrix} | & | & \dots & | \\ x_2 & x_3 & \dots & x_p \\ | & | & \dots & | \end{bmatrix},$$

where p is the complete number of snapshots. The data-driven dynamic model results in

$$X_2 = TX_1. \tag{3.41}$$

The LS solution to (3.41) can be obtained by minimizing the norm of $\|X_2 - TX_1\|_F$. This minimization problem can be efficiently solved via (3.42), where the pseudoinverse of X_1 is computed by SVD

$$T = X_2 X_1^{\dagger}. \tag{3.42}$$

where $X_1 = USV^{\iota}$, $U \in \mathbb{R}^{nM \times nM}$, $S \in \mathbb{R}^{nM \times nM}$, and $V \in \mathbb{R}^{(p-1) \times nM}$. At this stage, SVD eliminates nonessential singular values in S such that S is a square matrix enabling pseudoinverse of X_1. Note that S is a diagonal matrix holding singular values σ_i of X_1, for $i = 1, \ldots, nM$, which represents energy ranking [42, 125]. The thresholding of matrix S at rank r is used to truncate X_1 while retaining a high percentage of energy content. This truncation to rank r results in $\tilde{X_1} \approx \tilde{U}\tilde{S}\tilde{V}^{\iota}$, where $\tilde{U} \in \mathbb{R}^{nM \times r}$, $\tilde{S} \in \mathbb{R}^{r \times r}$, and $\tilde{V} \in \mathbb{R}^{(p-1) \times r}$. Herein, the projection matrix $Y = \tilde{U}^{\iota}$ is obtained. Note that SVD provides the projection between two different dimensional spaces ($\mathbb{R}^{nM} \to \mathbb{R}^{r}$).

Remark 18. The percentage energy content in each σ is given by $E_{\sigma_i} = \frac{\sigma_i}{\sum_{i=1}^{nM} \sigma_i}$. The r can be any positive integer between 1 to nM determined by a strict threshold μ. The value of $\mu \geq 0$ is chosen for the matrix S such that $\sigma_i \geq \mu \sum_{i=1}^{nM} \sigma_i$. This thresholding using μ keeps r leading σ's with high energy and eliminates remaining, i.e., $r+1$ to nM with lower energy.

The approximation of matrix T in (3.42) can be computed as

$$T \approx \tilde{T} = X_2 \tilde{V} \tilde{S}^{-1} \tilde{U}^{\iota}, \tag{3.43}$$

and dynamic model (3.41) results in

$$X_2 \approx \tilde{T} X_1, \tag{3.44}$$

where $\tilde{T}$ has the same dimension as T. For the large-scale networked system (3.3), (3.5) defines how one subsystem interacts with others. Eigenvalue analysis is critical to retrieve

complete dynamic information from state measurement $\tilde{x}_k$. It is possible to extract dynamic modes efficiently from a dynamic model, where $\tilde{x}_k$ is projected onto a linear subdomain of dimension r. The lower-dimensional dynamic model becomes

$$\begin{aligned}\xi_{k+1} &= Y\tilde{T}Y^{\dagger}\xi_k \\ &= YX_2\tilde{V}\tilde{S}^{-1}\xi_k \triangleq \bar{T}\xi_k,\end{aligned} \tag{3.45}$$

where $\bar{T} \triangleq YX_2\tilde{V}\tilde{S}^{-1}$ and $\bar{T} \in \mathbb{R}^{r\times r}$. Given Lemma 1, one can find a unique stabilizing control policy $\tilde{u}_k$ that minimizes the global $J(\tilde{u}_k, \tilde{x}_k)$ in (3.5) as

$$\begin{aligned}\bar{J}^*(\tilde{x}_k) = \min_{\tilde{u}_k} \bar{J}(\tilde{u}_k, \tilde{x}_k) &= \sum_{k=0}^{\infty} \left[\xi_k^{\mathrm{T}}\tilde{Q}_r\xi_k + \tilde{u}_k^{\mathrm{T}}R\tilde{u}_k\right] \\ &= \xi_k^{\mathrm{T}}\tilde{P}_r\xi_k,\end{aligned} \tag{3.46}$$

where $\tilde{P}_r \in \mathbb{R}^{r\times r} > 0$ is a truncated cost matrix.

Remark 19. Note that state measurement at each snapshot can be viewed as a sample of a signal. Since these samples are stacked over p snapshots, (X_1, X_2) forms a time-series signal. Thus, T in (3.41) provides a temporal mapping of the state measurement signal. Eigenanalysis of T gives dynamic modes to truncate data vectors in (3.32), so that dynamic information of (3.3) is preserved [125]. The eigenvalue analysis of $\bar{T}$ is more feasible than that of $T \approx \tilde{T}$, where $r << (nM)$. This analysis provides projection within the same dimensional space ($\mathbb{R}^r \to \mathbb{R}^r$) as obtained in (3.45).

The next theorem shows the extraction of complete dynamic information via eigenvalue analysis for the system of high dimensions, in contrast to [136]. This information is then used to truncate data vectors in (3.32) within the RL framework.

Theorem 7. *Given the reduced-order dynamic model in* (3.45)*, eigendecomposition of* $\bar{T}$ *provides essential dynamic modes to precondition data vectors in* (3.32)*, i.e., to project* $\rho_{xx_s} \in \mathbb{R}^{1\times(nM)^2}, \rho_{xu_s} \in \mathbb{R}^{1\times nmM^2}$, *and* $\rho_{uu_s} \in \mathbb{R}^{1\times(mM)^2}$ *onto a r-dimensional space.*

Proof. Let $\mathbb{P} = \tilde{U}\tilde{U}^\iota$ denote the orthogonal projection onto the space of $\tilde{X_1}$, where $\tilde{X_1} \approx \tilde{U}\tilde{S}\tilde{V}^\iota$. $\tilde{U}^\iota\tilde{U}$ is the identity matrix. If X_2 lies in span of X_1, then $\mathbb{P}\tilde{T} = \tilde{T}$.

$$\tilde{U}^\iota\tilde{T}\tilde{U} = \tilde{U}^\iota X_2\tilde{V}\tilde{S}^{-1}\tilde{U}^\iota\tilde{U} = \tilde{U}^\iota X_2\tilde{V}\tilde{S}^{-1} = \bar{T}. \tag{3.47}$$

Eigendecomposition of $\bar{T}$ generates eigenvalues λ and eigenvectors v that can examine essential characteristics of the system (3.3), such as oscillation modes and natural frequencies [117, 136],

$$\bar{T}v = \lambda v. \tag{3.48}$$

It is clear from (3.43) and (3.45) that eigenvalues of $\tilde{T}$ and $\bar{T}$ are analogous [125], given $Y = \tilde{U}^\iota$.

$$\tilde{T}v = X_2\tilde{V}\tilde{S}^{-1}\tilde{U}^\iota v = YX_2\tilde{V}\tilde{S}^{-1}v = \bar{T}v = \lambda v. \tag{3.49}$$

As shown in Remark 19, analyzing $\bar{T}$ is computationally tractable. Given the dynamic model in (3.45) and (3.40), dynamic modes of $T \approx \tilde{T}$, i.e., θ, and eigenvectors of $\bar{T}$, i.e., v, are related by a linear transformation,

$$\theta = \frac{1}{\lambda}X_2\tilde{V}\tilde{S}^{-1}v. \tag{3.50}$$

We have

$$\mathbb{P}\theta = \tilde{U}\tilde{U}^\iota\theta = \tilde{U}\tilde{U}^\iota\frac{1}{\lambda}X_2\tilde{V}\tilde{S}^{-1}v = \tilde{U}\frac{1}{\lambda}\bar{T}v = \tilde{U}v = \theta. \tag{3.51}$$

The columns of $\theta \in \mathbb{R}^{nM\times r}$ are called the DMD modes for linear systems, and are the exact eigenvectors of $T \approx \tilde{T} \in \mathbb{R}^{nM\times nM}$ [125, 136],

$$\begin{aligned}\tilde{T}\theta = \mathbb{P}\tilde{T}\theta &= (\tilde{U}\tilde{U}^\iota)(X_2\tilde{V}\tilde{S}^{-1}\tilde{U}^\iota)(\frac{1}{\lambda}X_2\tilde{V}\tilde{S}^{-1}v)\\ &= \tilde{U}\tilde{U}^\iota X_2\tilde{V}\tilde{S}^{-1}v = \tilde{U}\bar{T}v = \lambda\theta.\end{aligned} \tag{3.52}$$

The updated vectors are computed from (3.32) as

$$\hat{\rho}_{xx} = \rho_{xx_s}(\theta \otimes \theta) \in \mathbb{R}^{1\times r^2}, \tag{3.53a}$$

$$\hat{\rho}_{xu} = \rho_{xu_s}(I_{mM} \otimes \theta) \in \mathbb{R}^{1\times r(mM)}, \tag{3.53b}$$

$$\hat{\rho}_{uu} = \rho_{uu_s}(I_{mM} \otimes I_{mM}) \in \mathbb{R}^{1\times (mM)^2}. \tag{3.53c}$$

□

Remark 20. The truncation of data vectors in (3.53) is executed with DMD modes, unlike to [119] which uses SVD modes. DMD modes differ from individual SVD modes, and they preserve dynamic information in lower dimensions as given in Remark 19. As shown in Theorem 7, the dynamic model can be decomposed into dynamic modes, where eigenvalues characterize the temporal nature of the associated dynamic modes θ. Theorem 7 pertains to higher dimensional networked systems, in contrast to [136]. Moreover, DMD modes are extracted within the RL algorithm that iteratively truncates learned datasets to lower dimensions.

3.4.2 DMD-based Model-free Off-policy RL

This subsection proposes the DMD-preconditioned off-policy discrete-time RL algorithm in Algorithm 5. Unlike Algorithm 4, Algorithm 5 has an additional step of DMD-preconditioning to project large-scale system data vectors to a truncated order while preserving dynamic information. The policy evaluation and improvement steps are the same as Algorithm 4 but use a lower-dimensional state ξ_k.

Note that in (3.55), $\hat{\rho}_{xx} \in \mathbb{R}^{1\times r^2}, \hat{\rho}_{xu} \in \mathbb{R}^{1\times r(mM)}$, and $\hat{\rho}_{uu} \in \mathbb{R}^{1\times (mM)^2}$. Thus, (3.56) has $d = r^2 + (mM)^2 + r(mM)$ unknown parameters. The LS solution to (3.56) needs a full rank of $((\psi_r^j)^{\mathrm{T}}\psi_r^j)$ and, at a minimum, needs $\zeta \geq d$ samples at each iteration.

The following theorem proves that Algorithm 4 and Algorithm 5 have the identical convergence characteristics.

Algorithm 5 Data-driven Implementation of DMD-based Model-free RL Algorithm for a Large-scale System

1: **Initialization**: Given N, set $j = 0$ and small threshold e. Select stabilizing $\tilde{K}_r^0$, $\tilde{K}^0$, and control input $\tilde{u}_k = -\tilde{K}\tilde{x}_k + \varepsilon_k$, where ε_k is a probing noise.

2: **Data Collection**: Same as step 2 in Algorithm 4 and compute (3.32).

3: **DMD Preconditioning**: Set strict threshold μ. Compute reduced-order data vectors $(\hat{\rho}_{xx}, \hat{\rho}_{xu}, \hat{\rho}_{uu})$ in (3.53) and operators (φ_r^j, ψ_r^j) given by

$$\varphi_r^j = \begin{bmatrix} \xi_k^{\mathrm{T}} \tilde{Q}_r \xi_k + \xi_k^{\mathrm{T}} (\tilde{K}_r^j)^{\mathrm{T}} \tilde{R} \tilde{K}_r^j \xi_k \\ \vdots \\ \xi_{k+\zeta-1}^{\mathrm{T}} \tilde{Q}_r \xi_{k+\zeta-1} + \xi_{k+\zeta-1}^{\mathrm{T}} (\tilde{K}_r^j)^{\mathrm{T}} \tilde{R} \tilde{K}_r^j \xi_{k+\zeta-1} \end{bmatrix}, \tag{3.54}$$

$$\psi_r^j = \begin{bmatrix} \hat{\rho}_{xx} & \hat{\rho}_{xu} & \hat{\rho}_{uu} \end{bmatrix}. \tag{3.55}$$

4: **Policy Evaluation**: Compute $\tilde{P}_r^{j+1}$ by

$$((\psi_r^j)^{\mathrm{T}} \psi_r^j)^{-1} (\psi_r^j)^{\mathrm{T}} \varphi_r^j = \begin{bmatrix} \mathrm{vec}(\tilde{P}_r^{\,j+1})^{\mathrm{T}} \\ \mathrm{vec}((Y\tilde{B})^{\mathrm{T}} \tilde{P}_r^{\,j+1} Y \tilde{A} Y^{\dagger})^{\mathrm{T}} \\ \mathrm{vec}((Y\tilde{B})^{\mathrm{T}} \tilde{P}_r^{\,j+1} Y \tilde{B})^{\mathrm{T}} \end{bmatrix}. \tag{3.56}$$

5: **Policy Improvement**: Compute $\tilde{K}_r^{j+1}$ by

$$\tilde{K}_r^{\,j+1} = \left(R + (Y\tilde{B})^{\mathrm{T}} \tilde{P}_r^{\,j+1} Y \tilde{B}\right)^{-1} (Y\tilde{B})^{\mathrm{T}} \tilde{P}_r^{\,j+1} A. \tag{3.57}$$

6: **Stop if** $\|\tilde{K}_r^{\,j+1} - \tilde{K}_r^{\,j}\| \leq e$; Otherwise, set $j = j + 1$, and go to step 2.

7: **Collect**: $\tilde{K}^{j+1} = Y \tilde{K}_r^{\,j+1}$.

Theorem 8. ***(Convergence of Algorithm 5)*** *Given the networked system* (3.3)*, consider Algorithm 5 for the LQR graphical problem* (3.7)*. Then, Algorithm 5 converges to the optimal solution* (3.9)*, and P satisfies discrete-time ARE* (3.10)*, i.e, Algorithm 5 converges to Algorithm 4.*

Proof. It is seen from Lemma 1 that, given the large-scale networked system in (3.3), ξ_k satisfies (3.36), i.e., $\tilde{x_k} \approx Y^{\dagger}\xi_k$ holds $\forall \tilde{u}_k, \tilde{x}_k$ with the stable $Y\tilde{A}Y^{\dagger}$. Then, the control policy $\tilde{u}_k = -\tilde{K}_r^{j+1}\xi_k$ stabilizes system (3.36) at every iteration j, where $\tilde{K}^{j+1} = Y\tilde{K_r}^{j+1}$. Therefore, $Y\tilde{A}Y^{\dagger} - Y\tilde{B}\tilde{K}_r$ is Hurwitz. Given Remark 17, if Y is found then, similar to Algorithm 4, decomposed off-policy RL can be achieved, where $\tilde{K}_r = Y^{\dagger}\tilde{K} \in \mathbb{R}^{mM\times r}$ is

$$\tilde{K_r}^{*} = (R + (Y\tilde{B})^{\mathrm{T}}\tilde{P}_r^{*}(Y\tilde{B}))^{-1}(Y\tilde{B})^{\mathrm{T}}\tilde{P}_r^{*}(Y\tilde{A}), \tag{3.58}$$

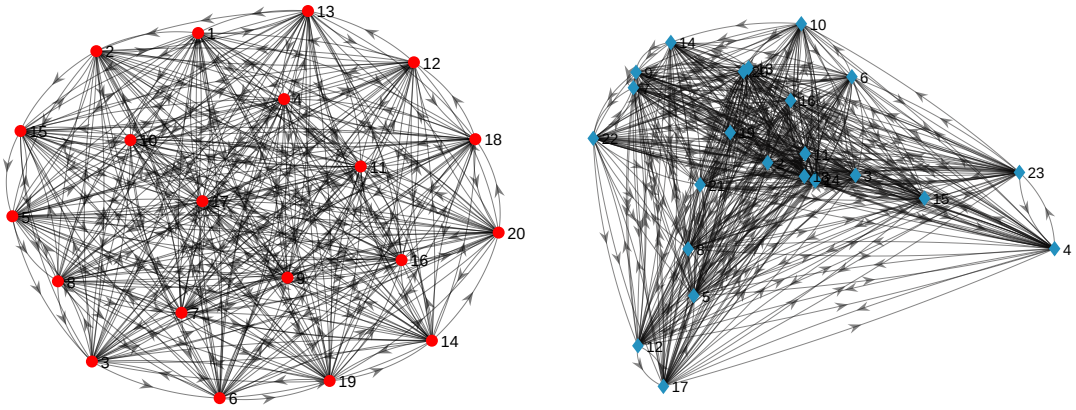
and $\tilde{P}_r = Y\tilde{P}Y^{\dagger} > 0 \in \mathbb{R}^{r\times r}$ satisfies

$$\begin{aligned}\tilde{P_r}^{*} = (Y\tilde{A})^{\mathrm{T}}\tilde{P_r}^{*}(Y\tilde{A}) - (Y\tilde{A})^{\mathrm{T}}\tilde{P}_r^{*}(Y\tilde{B})(R+ \\ (Y\tilde{B})^{\mathrm{T}}\tilde{P}_r^{*}(Y\tilde{B})^{-1})(Y\tilde{B})^{\mathrm{T}}\tilde{P}_r^{*}(Y\tilde{A}) + \tilde{Q}_r.\end{aligned} \tag{3.59}$$

Per Remark 14, Algorithm 4 converges to the optimal solution $(\tilde{K}^{*}, \tilde{P}^{*})$. Let $\tilde{K}_r$ be the initial stabilizing feedback gain matrix. Then, similar to Algorithm 4, given Remark 14, $(\tilde{P_r}^{j+1}, \tilde{K_r}^{j+1})$ is uniquely obtained by LS in (3.56) while satisfying the full-rank condition of ψ_r^j in (3.55) derived from data matrices in (3.53). Thus, Algorithm 5 converges to Algorithm 4, i.e., to the optimal solution. This finishes the proof. □

3.4.3 Computation Complexity

In this section, we show that Algorithm 5 has more computational tractability than Algorithm 4. As given in Remark 15, LS requires $(n^2+m^2+nm)M^2$ data samples to obtain a unique solution in Algorithm 4, whereas Algorithm 5 would require $r^2+(mM)^2+r(mM)$ samples.



(a) Consensus network with 20 subsystems (b) 24-area power system

Figure 3.2. Topologies considered for dynamically (a) decoupled and (b) coupled networks..

Considering ζ as data samples and d as unknown parameters, the computational complexity of LS for $\zeta \geq d$ is of the order $O(d^2\zeta)$ [110]. When finding optimal control using policy iteration for large values of M, Algorithm 4 has higher complexity of the order $O((n^2+m^2+nm)M^2)^2\zeta)$ compared to Algorithm 5 with order $O((r^2+(mM)^2+r(mM))^2\zeta)$, where $r << (nM)$. Therefore, Algorithm 5 is more tractable than Algorithm 4. Algorithm 5 conserves a substantial amount of learning time provided that r is small enough. This will be verified through the following numerical simulation studies.

3.5 Simulation Studies

We verify the proposed DMD-based model-free RL Algorithm 5 by simulating two large-scale networks. First, the consensus network problem is discussed in Section V-A, where each subsystem is dynamically decoupled but linked with a mutual performance function. Second, the load frequency control (LFC) for multi-area power systems [153] is presented in Section V-B, where subsystems are physically coupled and have a mutual performance function.

3.5.1 Consensus Network

Figure 3.2(a) shows a large-scale network with 20 subsystems. Consider the LQR graphical problem in (3.7) with $M = 20$. The linear large-scale networked dynamical system becomes

$$\tilde{x}_{k+1} = \tilde{A}\tilde{x}_k + \tilde{B}\tilde{u}_k, \tag{3.60}$$

where $\tilde{x} = (x_1^{\mathrm{T}}, \ldots, x_{20}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{60}$, and $x_i \in \mathbb{R}^3$ is the state of the subsystem i. Here, $\tilde{A} = I_{20} \otimes A \in \mathbb{R}^{60\times 60}$ and $\tilde{B} = I_{20} \otimes B \in \mathbb{R}^{60\times 20}$, where $(A \in \mathbb{R}^{3\times 3}, B \in \mathbb{R}^{3\times 1})$ is adopted from [69] as

$$A = \begin{bmatrix} 0.9064 & 0.0816 & -0.0005 \\ 0.0743 & 0.9012 & -0.0007 \\ 0 & 0 & 0.1326 \end{bmatrix}, B = \begin{bmatrix} 0.0015 \\ 0.0096 \\ 0.8673 \end{bmatrix}. \tag{3.61}$$

One can select penalizing local state weight $Q_1 = I_3$, relative state weight $Q_2 = 0.5I_3$, and control weight $R = 1$. The global performance function in (3.5) uses $\tilde{Q} = I_{20} \otimes Q_1 + \mathscr{L} \otimes Q_2$ and $\tilde{R} = I_{20} \otimes R$ for the Laplacian matrix $\mathscr{L} \in \mathbb{R}^{20\times 20}$ related with the undirected $\mathscr{G}$.

We first collect state data of the large-scale system under a given $\tilde{u}_k$ in (3.8), i.e., $\tilde{x}_k$ over $k = \{0 \ldots 2000\}$. In step 1 of Algorithm 5, consider the probing noise $\varepsilon_k = 0.1sin(9.8k) + 0.1sin(10k) + 0.1cos(10k) + 0.1cos(10.2k)$. Compute data vectors ($\rho_{xx_s}, \rho_{xu_s}$, ρ_{uu_s}) in step 2 of Algorithm 5 as in (3.32). Given the collected $\tilde{x}_k$, time-shifted matrices are generated as shown in (3.41). The strict threshold μ, selected for singular values, is 10^{-10} for a concise truncation. Figure 3.3 shows singular values captured for the balanced truncation in DMD. We compute dynamic modes θ using (3.50) to project (ρ_{xx_s}, ρ_{xu_s}, ρ_{uu_s}) to lower dimensions, i.e., ($\hat{\rho}_{xx}, \hat{\rho}_{xu}, \hat{\rho}_{uu}$) in (3.53), to conclude DMD-preconditioning in step 3 of Algorithm 5.

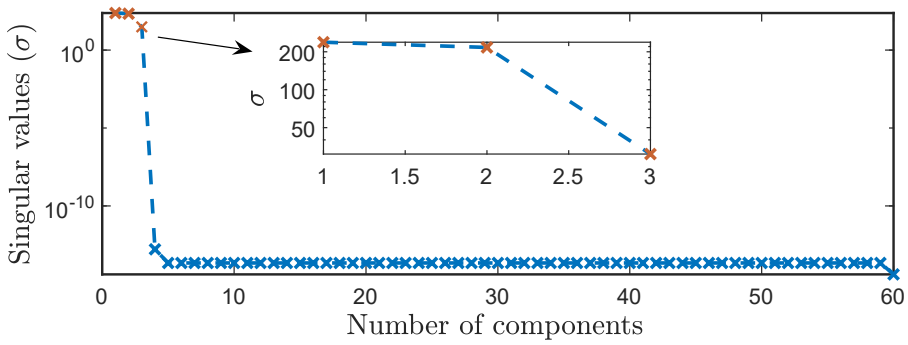


Figure 3.3. Low-rank approximation for DMD (3 dominant singular values)..

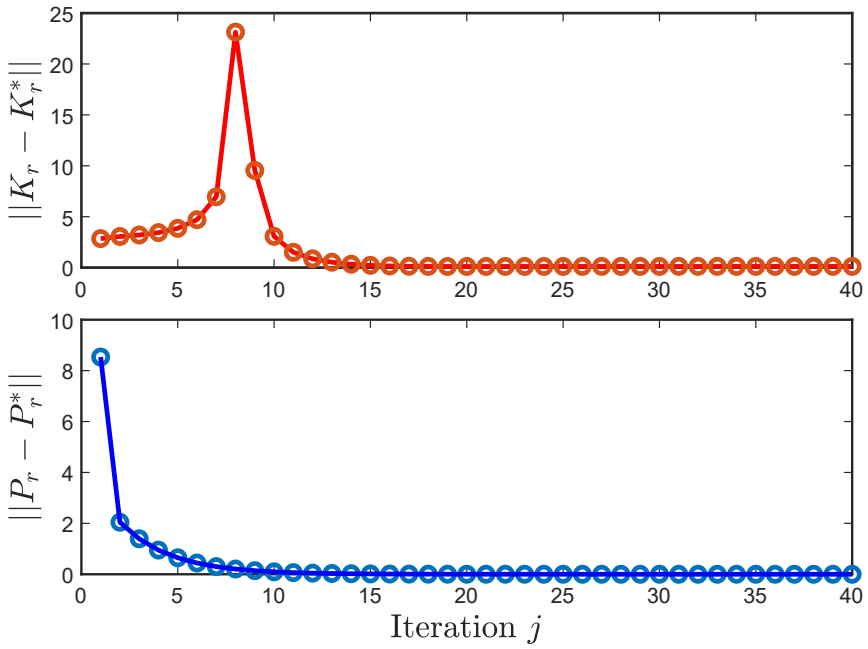


Figure 3.4. Convergence of the feedback gain K_r and performance cost P_r..

The feedback gain $\tilde{K}_r$ for the lower dimension is computed in (3.57). Figure 3.4 shows the convergence of feedback gain $\tilde{K}_r$ and performance cost $\tilde{P}_r$ over iteration j. The solution of LQR problem yields the feedback gain matrix $\tilde{K}$ in (3.9) which is computed in step 7 of Algorithm 5.

Table 3.1 shows the computational time for policy improvement with the proposed dimensionality reduction for the two cases of 20 and 12 subsystems. Figure 3.5 shows the

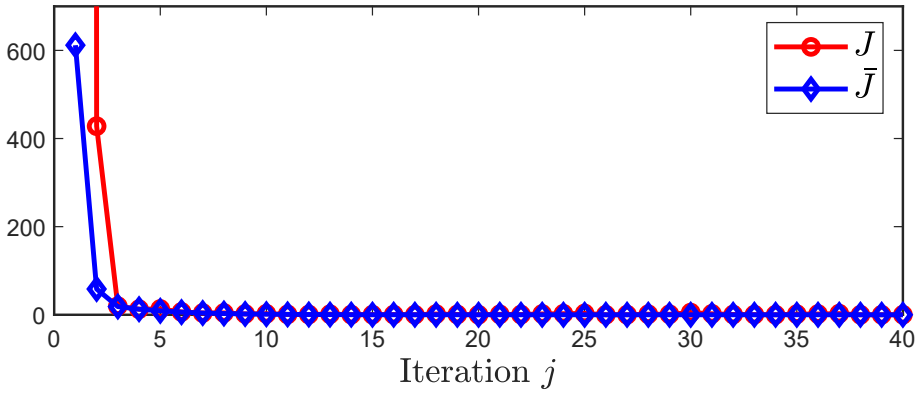


Figure 3.5. Convergence of the performance functions J and $\bar{J}$..

Table 3.1. Computational Time For Policy Improvement

Number of subsystems	**Spatio-temporal dimensions of state matrix before and after truncation**	**Computational time** (*Seconds*)
$M = 20$	$\mathbb{R}^{60\times 2000}$	116
	$\mathbb{R}^{3\times 2000}$	1
$M = 12$	$\mathbb{R}^{36\times 2000}$	70
	$\mathbb{R}^{3\times 2000}$	0.24

convergence of the performance function J in (3.5) and $\bar{J}$ in (3.46) for state matrices of dimensions 60 and 3, respectively. A similar conclusion can be drawn for a case with the state matrices of dimensions 36 and 3. At a minimum, fifteen iterations are required to achieve the optimal performance for $e = 0.003$. Table 3.1 and Fig. 3.5 indicate that the reduction in dimension significantly improves the learning time, whereas comparable optimal performance is achieved for nearly the same number of iterations. All computations are executed on an Intel(R) Xeon(R) W-10855M 2.80 GHz, 32 GB RAM, with MATLAB 2021a.

3.5.2 LFC of a Multi-area Power System

In this case study, a large-scale LQR graphical problem is formulated for LFC of multi-area power systems. Figure 3.2(b) shows a network of 24-areas power system. Load variation in the *i-th* area cause its frequency f_i to deviate from its reference value, and affects *j-th* area due to transient variations in f_j.

Figure 3.6 illustrates the layout of the *i-th* area LFC model for a multi-area power system. A generator, governor, and turbine are located in each *i-th* area. Using the linearized LFC model for multi-area power systems, adopted from [8], we examine the dynamics of the *i-th* area without load disturbance

$$\Delta \dot{f}_i = \frac{-1}{T_{pi}} \Delta f_i + \frac{K_{pi}}{T_{pi}} \Delta P_{mi} - \frac{K_{pi}}{T_{pi}} \Delta P_{tie,i}, \tag{3.62a}$$

$$\Delta \dot{P}_{mi} = \frac{-1}{T_{ti}} \Delta P_{mi} + \frac{1}{T_{ti}} \Delta P_{gi}, \tag{3.62b}$$

$$\Delta \dot{P}_{gi} = \frac{-1}{R_i T_{gi}} \Delta f_i - \frac{1}{T_{gi}} \Delta P_{gi} + \frac{u_i}{T_{gi}}, \tag{3.62c}$$

$$\Delta \dot{P}_{tie,i} = 2\pi \sum_{j \neq i, j=1}^{M} T_{ij} (\Delta f_i - \Delta f_j), \tag{3.62d}$$

where definitions for *i-th* area state variables and parameters are summarized in Table 3.2.

The input to the controller is $ACE_i = \Delta P_{tie,i} + \beta_i \Delta f_i$, and the usual choice of β_i is $\frac{1}{R_i} + D_i$. At the local level, the system dynamics of the *i-th* area can be given as

$$\dot{x}_i = A_1 x_i + A_2 \sum_{j \neq i, j=1}^{M} T_{ij} (x_i - x_j) + B u_i, \quad i \in \mathscr{M}, \tag{3.63}$$

Table 3.2. Definitions for LFC of a multi-area power system

Variables	**Definition**
$\Delta f_i, \Delta P_{mi}, \Delta P_{gi}, \Delta P_{tie,i}$	Frequency deviation, Generator output deviation, Governor valve deviation, and Tie-line power deviation, respectively
T_{pi}, T_{gi}, T_{ti}	Time constants of power system, governor and turbine, respectively
K_{pi}	Gain of power system
$u_i = \Delta P_{ci}$ P_{ci}	Control input Automatic generation control
R_i	Speed regulation parameters of governor
T_{ij}	Gain of tie-line interconnection between *i-th* and *j-th* area
ACE_i	Area control error signal
D_i	Load dependency factor

where $x_i = [\Delta f_i, \Delta P_{mi}, \Delta P_{gi}, \Delta P_{tie,i}]^{\mathrm{T}}$, and

$$A_1 = \begin{bmatrix} \frac{-1}{T_p} & \frac{K_p}{T_p} & 0 & \frac{-K_p}{T_p} \\ 0 & \frac{-1}{T_t} & \frac{1}{T_t} & 0 \\ \frac{-1}{RT_g} & 0 & \frac{-1}{T_g} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 2\pi & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{T_g} \\ 0 \end{bmatrix}. \tag{3.64}$$

The subscript i is ignored in A_1, A_2, and B, for brevity, as areas are considered to have identical dynamics. The numerical values of parameters given in Table 3.3 are adopted from [8].

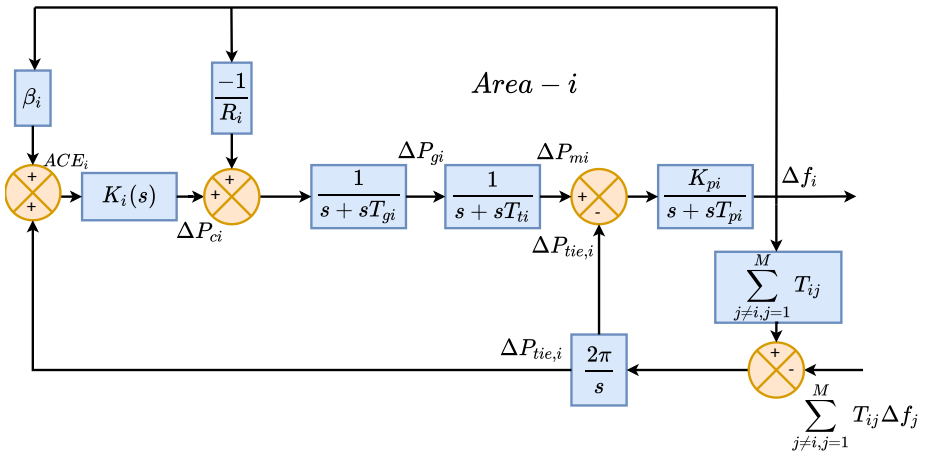


Figure 3.6. Block diagram of the LFC model in M-area power systems with tie-line interconnections..

Table 3.3. Parameters of the multi-area power system

Parameter	T_p (s)	T_g (s)	T_t (s)	K_p (MW/MW)	R (Hz/MW)	T_{ij} (MW/Hz)
i-th Area	20	0.08	0.3	120	2.4	0.015

For $M = 24$, global dynamics with states $\tilde{x} = (x_1^{\mathrm{T}}, \ldots, x_{24}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{96}$ and control inputs $\tilde{u} = (u_1^{\mathrm{T}}, \ldots, u_{24}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{24}$ are

$$\dot{\tilde{x}} = \tilde{A}\tilde{x} + \tilde{B}\tilde{u}, \tag{3.65}$$

where $\tilde{A} = (I_M \otimes A_1 + \mathscr{L} \otimes A_2) \in \mathbb{R}^{96\times 96}$, and $\tilde{B} = I_M \otimes B \in \mathbb{R}^{96\times 24}$. Select penalizing local state weight $Q_1 = 2I_4$, relative state weight $Q_2 = 0.01I_4$, and control weight $R = 1$. Performance function in (3.5) uses $\tilde{Q} = I_{24} \otimes Q_1 + \mathscr{L} \otimes Q_2$ and $\tilde{R} = I_{24} \otimes R$ with the Laplacian matrix $\mathscr{L} \in \mathbb{R}^{24\times 24}$.

In order to design a discrete-time LQR control policy, system (3.65) is discretized using the Zero-Order-Hold (ZOH) method with a sampling period of $\tau = 1$ seconds. The resulting discrete-time system dynamics become

$$\tilde{x}_{(k+1)\tau} = \tilde{A}_d \tilde{x}_{k\tau} + \tilde{B}_d \tilde{u}_{k\tau}, \tag{3.66}$$

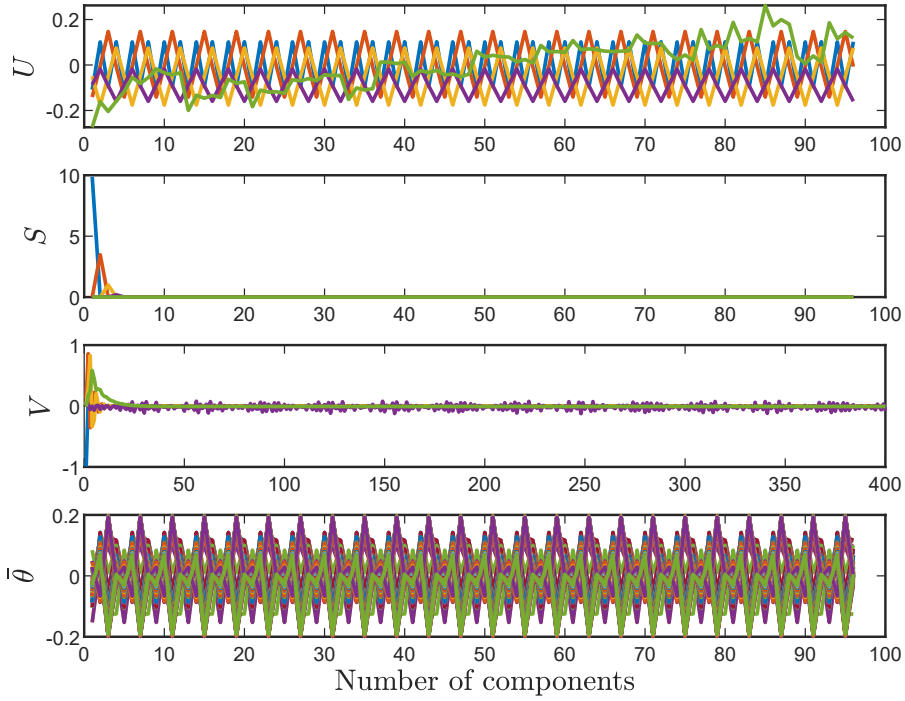


Figure 3.7. Total dynamic modes $\bar{\theta}$ (real part only) of M-area power system..

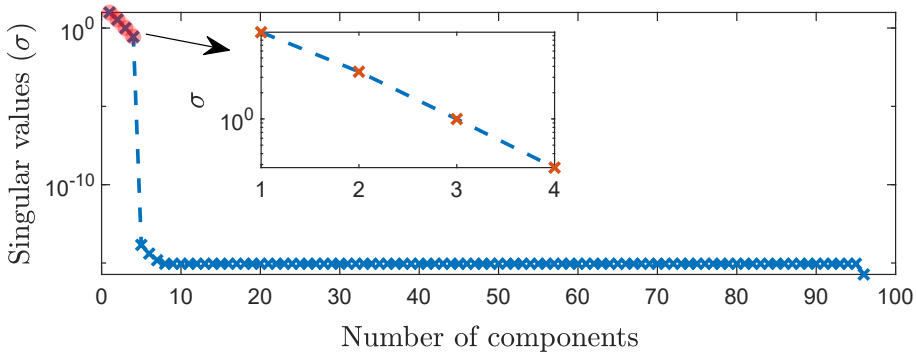


Figure 3.8. Low-rank approximation for DMD (4 dominant singular values)..

where $A_d = e^{\tilde{A}\tau} \in \mathbb{R}^{96\times 96}$, and $B_d = \int_0^\tau e^{\tilde{A}\tau}\tilde{B}dt \in \mathbb{R}^{96\times 24}$. The LFC in multi-area power system is an example of a large-scale system having multiple sampling periods. That is, control signals sent to areas are in discrete time with a sampling period of 1-5 seconds (s). Thus, we use discrete-time LFC [142].

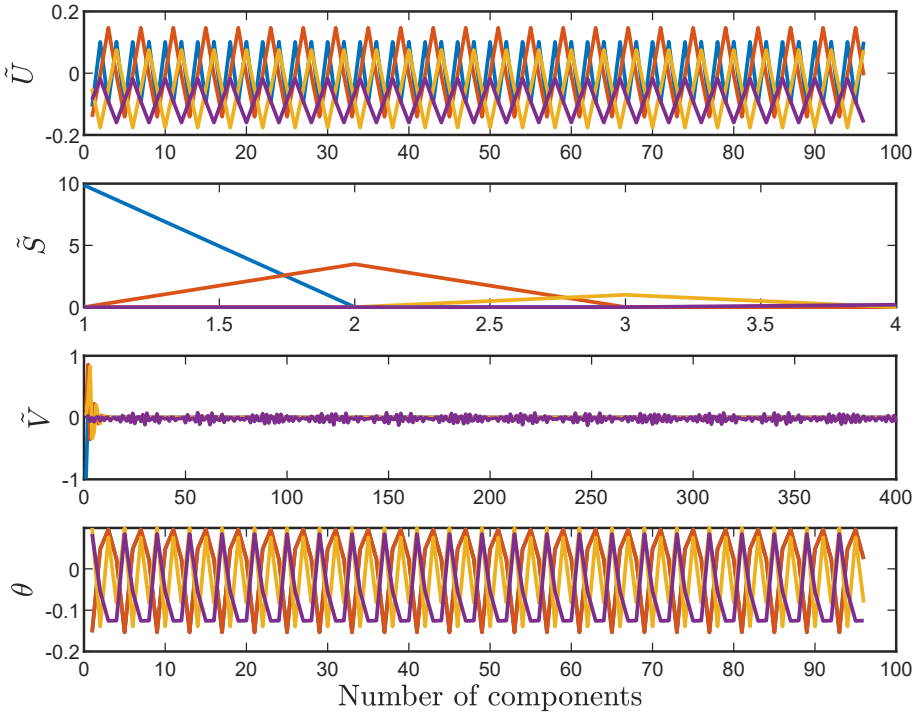


Figure 3.9. Essential modes (real part only) for the singular values in Fig. 3.8..

The state data of (3.66) is collected under $\tilde{u}_k$ in (3.8), i.e., $\tilde{x}_k$ over $k = \{0 \dots 400\}$ and probing noise $\varepsilon_k = 0.01sin(9.8k) + 0.01sin(10k) + 0.01cos(10k) + 0.01cos(10.2k)$. Figure 3.7 illustrates the total dynamic modes $\bar{\theta} \in \mathbb{R}^{96\times 96}$ of the original system with the SVD matrices where $U \in \mathbb{R}^{96\times 96}$, $S \in \mathbb{R}^{96\times 96}$, and $V \in \mathbb{R}^{400\times 96}$. Figure 3.8 shows that 4 dominant singular values need to be captured for the balanced truncation in DMD. Figure 3.9 presents 4 essential dynamic modes $\theta \in \mathbb{R}^{96\times 4}$ of decomposed system with the truncated SVD matrices where $\tilde{U} \in \mathbb{R}^{96\times 4}$, $\tilde{S} \in \mathbb{R}^{4\times 4}$, and $\tilde{V} \in \mathbb{R}^{400\times 4}$. Figure 3.10 shows convergence of feedback gain $\tilde{K}_r$ and performance cost P_r over iteration j. As seen in Fig. 3.11, state deviations of M-area power system i.e., $\Delta f_i, \Delta P_{mi}, \Delta P_{gi}, \Delta P_{tie,i}$ converges to zero under the proposed accelerated control scheme.

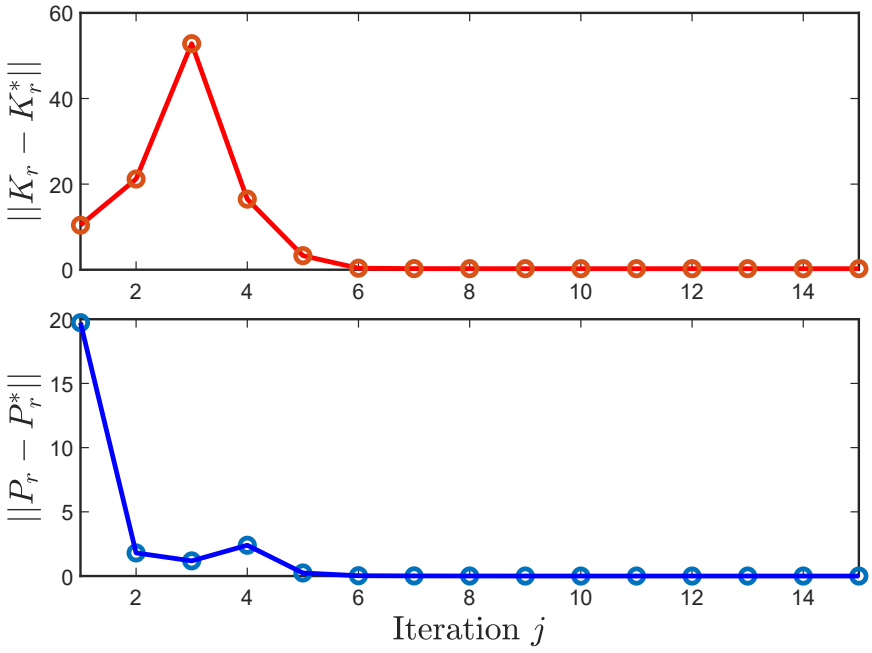


Figure 3.10. Convergence of K_r and P_r..

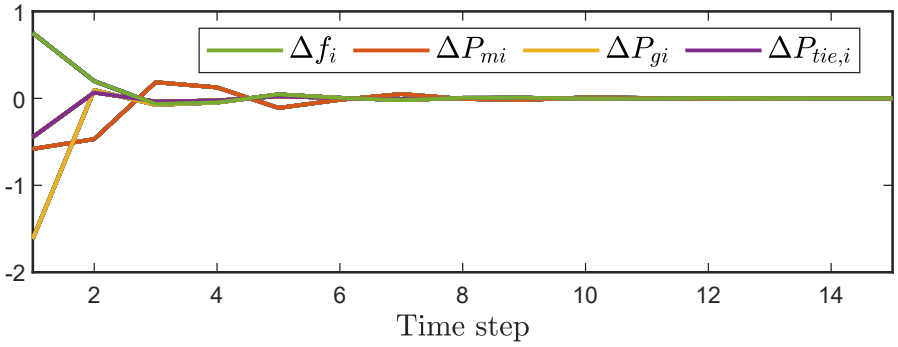


Figure 3.11. State trajectories, i.e., deviations of frequency, generator output, governor valve, and tie-line power of *i- th* area power system..

3.5.3 Comparative Studies

We compare the computational features of our Algorithm 5 and the RL algorithm in [119] for the LQR control problem of the large-scale networked system in (3.3). We consider the case study in Section 3.5.1 and use the same parameters. The system model in (3.61) with M equal to 20 and 12 are considered. We notice the difference in iteration

numbers (No.) and computational time (s). Note that Remark 11 is still satisfied. The original dynamic information for the system in (3.3) is not lost when using DMD for data vectors decomposition instead of SVD modes. Table 3.4 shows that our proposed algorithm has less computational time and iterations than those based on [119].

Table 3.4. Computation features of Algorithm 5 against [119]

Number of subsystems	Iteration No.		Computational Time (s)	
	Algorithm 5	[119]	Algorithm 5	[119]
20	15	18	1	1.2
12	12	15	0.24	0.29

3.6 Conclusion & Future Work

This paper proposes a model-free, off-policy discrete-time RL algorithm to solve the optimal control problem for large-scale systems. Using DMD, this approach reduces the efforts of RL control while retaining the dynamic information of the original large-scale system. Since DMD preconditioning is data-driven, the RL algorithm becomes entirely model-free. Potential path forward would treat DMD-based reinforcement learning, for high-dimensional systems having heterogeneous subsystems, extended LQR formulations and DMD-based inverse reinforcement learning [35, 152].

Chapter 4

Data-Efficient Reinforcement Learning for Complex Nonlinear Systems *

*This chapter, titled 'Data-Efficient Reinforcement Learning for Complex Nonlinear Systems,' was originally published in IEEE Transactions on Cybernetics, doi: 10.1109/TCYB.2023.3324601. It has been reprinted with permission from all co-authors and is used with permission from IEEE without any revisions. This version represents the authors' accepted manuscript.

CHAPTER 4

Data-Efficient Reinforcement Learning for Complex Nonlinear Systems

4.1 Introduction

Accurate modeling of complex nonlinear systems in various control applications can be challenging, necessitating data-driven control [132]. The indirect data-driven control approach addresses these complex systems by first identifying models from measured data and, then, applying a model-based control [78]. By contrast, direct methods design stabilizing controllers directly from measured data without explicit model identification. Recent progress in data-driven control allows for the development of model-free controllers [130]. Data-driven optimal control has been widely studied for nonlinear systems in the fields of robotics and power systems [29, 82, 149, 160]. Nonlinear optimal control problems need to solve the Hamilton Jacobi Bellman (HJB) equation and find a stabilizing controller. Various techniques have been designed to solve these tasks for specific types of systems [70, 79, 143], but this remains a challenging problem. Herein, we focus on the optimal control design via reinforcement learning (RL) for unknown discrete-time nonlinear systems.

We see RL through the lens of Koopman theory to handle complex nonlinear systems. This approach allows RL algorithms to iteratively learn a stabilizing controller and optimize a specific performance index. In model-based RL algorithms, model extraction could become strenuous for complex and high-dimensional systems [2,55,88]. The model-free, off-policy RL algorithm [69,80,84,90] can exploit the dataset generated from interactions with the environment to find optimal policies. Employing RL [22, 35, 81, 85, 89, 157]

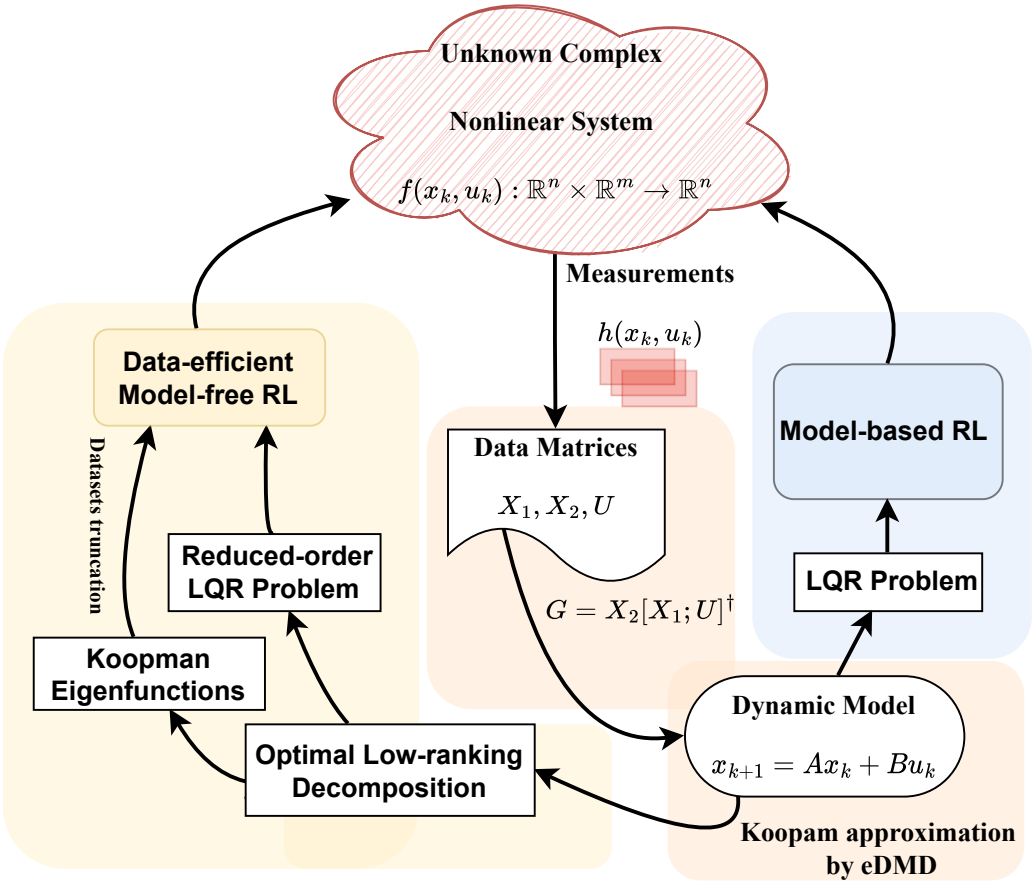


Figure 4.1. Overview of the proposed framework for complex nonlinear systems. The linear model is obtained with Koopman theory. eDMD approximates Koopman to a finite dimension. Model-based and data-efficient model-free RL algorithms solve LQR control problems for nonlinear systems with unknown models..

could result in over-fitting due to potentially near-singular rank matrices and longer run times for high-dimensional systems.

Data-driven optimal control is discussed in [50, 133], where a deep neural network is employed for system identification. Although the data-driven Koopman approach has produced promising results [48, 75, 92, 123, 127, 144], to our knowledge, it has not been considered in data-efficient model-free RL setting to handle high- dimensional nonlinear systems. We use Koopman theory [63, 74] to analyze symmetries in high-dimensional datasets. These symmetries are called Koopman eigenfunctions that oscillate at fixed natu-

ral frequencies. Koopman eigenfunctions serve as intrinsic coordinates along which large datasets can be effectively truncated [73]. These eigenfunctions can help reflect the exact behaviors of the high-dimensional system into lower dimensions [25]. We develop a lower-dimensional mapping of large datasets and study their temporal dynamics. Using this lower-dimension model significantly improves the efficacy of the data-driven RL algorithms. This approach does not rely on the system model, making it appropriate for model-free RL adaptation. The use of dimensionality reduction to accelerate RL has been recently proposed for both coupled and decoupled linear networked systems in [34]. Herein, we present an extension to [34] for complex nonlinear systems using Koopman eigenfunctions.

Datasets of the complex nonlinear system collected over space and time provide more relevant dynamic information compared to just spatial datasets [24]. Dynamic mode decomposition (DMD) is a computationally viable framework to analyze spatial-temporal datasets. It provides a finite approximation of the Koopman operator, which can be depicted as the best-fit linear dynamical model realizations [111, 125]. The modified framework of DMD, namely extended dynamic mode decomposition (eDMD) [10, 148], finds a linear model with the effect of control inputs. When singular values closer to zero are preserved, singular-value decomposition (SVD) approaches could involve near-singular matrices. The truncation step in eDMD selects a relatively small threshold and sets all eigenvalues below this threshold to zero. We show that eDMD can extract spatiotemporally coherent patterns from data. The off-policy RL is less efficient since it allows the agent to learn from a different behavior policy than the one it is currently improving. Experience replay improves data efficiency by storing and reusing past data, enhancing system excitability, and aiding learning [115]. In contrast, we use eDMD to identify the best-fit linear system approximation for the original nonlinear system, even with an unknown model. Subsequently, we integrate the off-policy RL with Koopman theory to design an efficient optimal control strategy. Salient contributions of this paper are:

1. To alleviate strenuous system modeling for a complex nonlinear system, we develop an entirely data-driven model-based RL algorithm using Koopman operators. This method avoids the need for an exact model, distinguishing it from RL approaches in [2, 55, 88, 155]. This feature makes it a more practical choice for control tasks in complex and uncertain environments.
2. We use Koopman eigenfunctions to truncate large datasets and develop a data-efficient model-free RL algorithm that reduces the required data for optimal control learning while retaining the dynamical information of the original nonlinear complex system. This algorithm significantly improves data efficiency in RL, for unknown complex systems, addressing longer learning times in [22, 81, 85, 89, 99, 157].
3. We provide convergence analysis for the proposed data-efficient model-free algorithm and validate our framework on a complex nonlinear power system.

Figure 4.1 illustrates an overview of the proposed approach. This paper is organized as follows: Section II introduces the data-driven Koopman operator and its finite approximation via eDMD. Sections III and IV present RL algorithms that are entirely data-driven and data-efficient, respectively. Section V offers simulation studies on nonlinear power system dynamics. Section VI concludes the paper.

Notations: $\otimes$ stands for the Kronecker product. For vectors a, b and matrix W, $a^\top W b = (b^\top \otimes a^\top) vec(W)$. t_k denotes the k^{th} discrete-time step. The n-dimensional Euclidean space is denoted by $\mathbb{R}^n$. I_n indicates the n-dimensional identity matrix. $\|\cdot\|_F$ and $\|\cdot\|$ denotes Frobenius norm and Euclidean norm of a vector or a matrix, respectively. $\mathbb{C}^-$ indicates an open left-half complex plane. $[l_{ij}]$ denotes a matrix whose ij-th element is l_{ij}. $S(L)$ denotes the spectrum of $L = \{\lambda_1(L), \ldots, \lambda_M(L)\}$, where λ_i is the i^{th} eigenvalue. L^ι stands for the complex conjugate transpose of L. $\langle \cdot, \cdot \rangle$ indicates the inner product operation. $L^\dagger$ denotes the pseudoinverse of L matrix. For a matrix $L = [l_{ij}] \in \mathbb{R}^{n \times n}$, $\text{vec}(L) = [l_{11}\ l_{12}\ \ldots\ l_{1n}\ l_{22}\ \ldots\ l_{(n-1)n}\ l_{nn}]^\top$.

4.2 Preliminaries and Problem Formulation

This section discusses the Koopman operator for an unknown nonlinear dynamical system. We use the eDMD method to find the best-fit linear model, given system trajectories, and design linear quadratic regulator (LQR) control.

4.2.1 Preliminaries of the Data-driven Koopman Operator

Consider a discrete-time, nonlinear, non-affine system

$$x_{k+1} = f(x_k, u_k), \tag{4.1}$$

where $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, and continuously differential dynamic map $f(x_k, u_k) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$.

Following the Koopman theory in [74], we consider an observable function as $h(x_k) : \mathbb{R}^n \to \mathbb{R}^n$, and the unknown system model in (4.1) produces trajectories as

$$h(x_k) = \begin{bmatrix} h_1(x_k), h_2(x_k), \ldots, h_n(x_k) \end{bmatrix}^\top, \tag{4.2}$$

where n is the number of spatial measurements.

Assumption 7. The trajectories $(h(x_k), u_k)$ of the complex system in (4.1) are known, but f is unknown.

Definition 3. The Koopman operator $\mathscr{K}$ for the dynamical system (4.1) acts on the observable function $h(x_k)$ and propagates it to the next time step [74, 117]

$$h(x_{k+1}) = h(f(x_k, u_k)) = \mathscr{K} h(x_k). \tag{4.3}$$

Remark 21. Note that we can convert a non-affine dynamical control system in (4.1) to an affine control system by defining extended state space. Define the $\bar{f}$ on the extended state space, where control input is added to the state vector, $y_k := [x_k^\top, u_k^\top]^\top \in \mathbb{R}^{(n+m)}$.

Similar to (4.2), consider $h(y_k) : \mathbb{R}^{(n+m)} \to \mathbb{R}^{(n+m)}$ on extended state space, and (4.1) produces trajectories as

$$h(y_k) = \Big[h_1(y_k), h_2(y_k), \ldots, h_{(n+m)}(y_k)\Big]^\top. \tag{4.4}$$

Given Remark 21 and Definition 3, operator $\mathscr{K}$ for (4.1) acts on the $h(y_k)$ and propagates it to the next time step on extended state space become

$$h(y_{k+1}) = h(\bar{f}(y_k)) = \mathscr{K} h(y_k). \tag{4.5}$$

Although the underlying system in (4.1) is nonlinear, the system (4.3) and (4.5) is propagating linearly in the space of observables. The operator $\mathscr{K}$ is linear but has infinite dimension [24, 74, 117], which makes computation costly. However, the linear nature of $\mathscr{K}$ enables to execute eigendecomposition of $\mathscr{K}$ as

$$\theta_i(x_{k+1}) = \mathscr{K}\theta_i(x_k) = \lambda_i \theta_i(x_k), \quad i = 1, 2, \ldots, n, \tag{4.6}$$

where $\theta_i(x_k)$ and λ_i are Koopman eigenfunctions and eigenvalues of $\mathscr{K}$ operator, respectively. Eigenanalysis finds the finite-dimensional approximation of $\mathscr{K}$ such that (4.3) propagates linearly in the subspace spanned by a finite set of observable functions $h(x_k)$. The m^{th} observable function can be written in terms of Koopman eigenfunctions $\theta_i(x_k)$ [24,148] as

$$h_m(x_k) = \sum_{i=1}^{n} \theta_{mi}(x_k) v_{mi}, \tag{4.7}$$

where $\theta_{mi}(x_k)$ and v_{mi} are m^{th} Koopman eigenfunction and Koopman mode for the m^{th} observable function, respectively.

Problem: Given Assumption 7, lift the nonlinear system in (4.1) into a best-fit linear model and find its optimal control input in terms of the linear feedback policy $u_k = -Kx_k$ such that the following quadratic performance index J is minimized

$$J\big(u_k, x_k\big) = \sum_{k=0}^{\infty} \big[x_k^\top Q x_k + u_k^\top R u_k\big], \tag{4.8}$$

where $Q = Q^\top \in \mathbb{R}^{n\times n} \geq 0$ and $R = R^\top \in \mathbb{R}^{m\times m} > 0$.

4.2.2 LQR Control via Koopman Approximation

We use eDMD [92, 148], where a higher-dimensional function space is projected to a lower-dimensional vector space, to obtain a finite approximation of $\mathscr{K}$. Suppose we have access to datasets from simulation experiments, where $\bar{x}_k = h(x_k)$ represents a vector of observables. Given Assumption 7, we apply eDMD to construct linear state and control matrices (A, B) using the system state and control measurements. The eDMD in [148] modifies the framework of DMD to include the effect of the inputs $u_k \in \mathbb{R}^m$,

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k, \tag{4.9}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$.

The behavioral trajectories $(h(x_k), u_k)$ are collected from experiments on the unknown system (4.1). Given the state measurement sampled at t_k, $h(x_k) \in \mathbb{R}^n$, the state data matrices $X_1, X_2 \in \mathbb{R}^{n\times(p-1)}$ are collected such that X_2 is the time-shifted matrix of X_1. They are expressed as

$$X_1 = \begin{bmatrix} | & | & \dots & | \\ \bar{x}_1 & \bar{x}_2 & \dots & \bar{x}_{p-1} \\ | & | & \dots & | \end{bmatrix}, \quad X_2 = \begin{bmatrix} | & | & \dots & | \\ \bar{x}_2 & \bar{x}_3 & \dots & \bar{x}_p \\ | & | & \dots & | \end{bmatrix}. \tag{4.10}$$

The data matrix of control inputs $U \in \mathbb{R}^{m\times(p-1)}$ is collected as

$$U = \begin{bmatrix} | & | & \dots & | \\ u_1 & u_2 & \dots & u_{p-1} \\ | & | & \dots & | \end{bmatrix}, \tag{4.11}$$

where p is the number of snapshots. Then, the data-driven dynamic model in matrix form becomes

$$X_2 = AX_1 + BU. \tag{4.12}$$

Given the measurements $y_k \in \mathbb{R}^{(n+m)}$ on the extended state space, $Y \in \mathbb{R}^{(n+m)\times(p-1)}$ is the concatenation of the state and control measurements. It is

$$Y = \begin{bmatrix} X_1 \\ U \end{bmatrix} = \begin{bmatrix} | & | & \dots & | \\ y_1 & y_2 & \dots & y_{p-1} \\ | & | & \dots & | \end{bmatrix}. \tag{4.13}$$

The system model in (4.12) can be written as

$$X_2 = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} X_1 \\ U \end{bmatrix} = GY, \tag{4.14}$$

where $G \in \mathbb{R}^{n\times(n+m)}$ is an augmented matrix. The least-squares (LS) solution to (4.14) can be obtained by minimizing the norm of $\|X_2 - GY\|_F$. This minimization problem can be efficiently solved via $G = X_2 Y^\dagger$, where SVD computes the pseudoinverse of Y by

$$Y = U_k S_k V_k^\iota = \begin{bmatrix} U_{k1} \\ U_{k2} \end{bmatrix} S_k V_k^\iota, \tag{4.15}$$

where $U_k \in \mathbb{R}^{(n+m)\times(n+m)}$, $S_k \in \mathbb{R}^{(n+m)\times(n+m)}$, $V_k \in \mathbb{R}^{(p-1)\times(n+m)}$. Split U_k into two components (U_{k1}, U_{k2}) to yield G as

$$G \approx \bar{G} = X_2 V_k S_k^{-1} \begin{bmatrix} U_{k\,1}^\iota \\ U_{k\,2}^\iota \end{bmatrix}. \tag{4.16}$$

where $U_{k1} \in \mathbb{R}^{n\times(n+m)}$ and $U_{k2} \in \mathbb{R}^{m\times(n+m)}$. Similarly, $X_2 = USV^\iota$, where $U \in \mathbb{R}^{n\times n}$, $S \in \mathbb{R}^{n\times(p-1)}$, $V \in \mathbb{R}^{(p-1)\times(p-1)}$. The system matrices (A,B) are computed as

$$A \approx \bar{A} = X_2 V_k S_k^{-1} U_{k\,1}^\iota, B \approx \bar{B} = X_2 V_k S_k^{-1} U_{k\,2}^\iota. \tag{4.17}$$

Provided (A,B) in (4.17), the LQR control finds the optimal control input u_k^*. The control input u_k^* minimizes $J(u_k, x_k)$ in (4.8) subject to (4.9) as

$$\begin{aligned} J^*(x_k) = \min_{u_k} J(u_k, x_k) &= \min_{u_k} \sum_{k=0}^{\infty} \left[x_k^\top Q x_k + u_k^\top R u_k \right] \\ &= x_k^\top P x_k, \end{aligned} \tag{4.18}$$

where $P = P^\top \in \mathbb{R}^{n\times n} > 0$ is a cost matrix. The optimal control input u_k^* [79] is given by

$$u_k^* = -K^* x_k, \tag{4.19}$$

where $K^* \in \mathbb{R}^{m\times n}$ is

$$K^* = (R + B^\top P B)^{-1} B^\top P A, \tag{4.20}$$

and P satisfies the discrete-time Algebraic Riccati Equation

$$P = A^\top P A - A^\top P B (R + B^\top P B)^{-1} B^\top P A + Q. \tag{4.21}$$

4.3 Data-driven Koopman-based RL Algorithm

This section develops Koopman-based Data-driven RL algorithms to solve the LQR control problem in (4.18). We leverage the Koopman theory to derive a linear dynamic model for the data-driven nonlinear system in (4.1).

4.3.1 Data-driven Model-based RL via Koopman Approximations

The solutions to the LQR problem in (4.18), i.e., (P, K^*), are computed using RL. The following lemma recalls the policy iteration (PI) for the discrete-time optimal LQR [52].

Lemma 2. See [52, 72]. Consider an initial stabilizing control gain matrix $K^0 \in \mathbb{R}^{m\times n}$. Then, find $P^j = (P^j)^\top \in \mathbb{R}^{n\times n} > 0$ for each $j = 0, 1, \ldots$, as the solution of the Lyapunov equation

$$P^j = (A^j)^\top P^j A^j + (K^j)^\top R K^j + Q, \tag{4.22}$$

where $A^j = A - BK^j$ and

$$K^{j+1} = (R + B^\top P^j B)^{-1} B^\top P^j A. \tag{4.23}$$

The matrix K^0 is selected such that the following claims hold.

1. $S(A^j) \subset \mathbb{C}^-$, i.e., the matrix A^j is Hurwitz,
2. $\lim_{j\to\infty} K^j = K^*$, $P \leq P^{j+1} \leq P^j$, $\lim_{j\to\infty} P^j = P$.

We present a model-based design in Algorithm 6 to find the optimal policy. Given Lemma 2, (4.22) and (4.23) can be iteratively solved using Algorithm 6.

Algorithm 6 Koopman-Based Discrete-Time RL Algorithm

1 **Require:** Collected behavioral trajectories (x_k, u_k) of (4.1).

2 **Initialization:** Set $j = 0$ and a small threshold e.

3 **Koopman:**

3.1 Input: Data matrices X_1, X_2, U.

3.2 Augment: $Y = \begin{bmatrix} X_1 \\ U \end{bmatrix}$, $X_2 = GY$.

3.3 Compute SVD: $(U_k, S_k, V_k) \leftarrow Y$.

3.4 Split: $U_{k1}, U_{k2} \leftarrow U_k$.

3.5 Output: Identify matrices (A, B) using (4.17).

4 Set stabilizing K^0 and K in Lemma 2.

5 Solve (4.22) and (4.23) for (P^j, K^{j+1}) simultaneously.

6 **Stop if** $\|K^{j+1} - K^j\| \leq e$; Otherwise, set $j = j+1$ and go to step 4.

Remark 22. It is clear from Lemma 2 that model-based PI solutions require knowledge of system matrices (A, B).

RL Algorithm 6 requires the complete knowledge of system matrices (A, B) found from Koopman theory. Moreover, the RL Algorithm 6 becomes infeasible for $n >> 1$, requiring the solution of high-dimensional Lyapunov equations at each iteration. We can design a data-driven model-free format to obtain (P, K^*), where the system model found by the Koopman approach and datasets learned iteratively can be reduced to lower dimensions.

4.3.2 Data-driven Model-free Off-policy RL

We present the model-free RL version based on [70, 128] to solve the LQR problem in (4.18) for the data-driven nonlinear system in (4.1). The system (4.9) is rewritten to present the off-policy RL approach as

$$x_{k+1} = A_f x_k + B(K^j x_k + u_k), \tag{4.24}$$

where $A_f = A - BK^j$. The fixed stabilizing policy u_k generates the behavioral trajectory data used in learning. The off-policy Bellman equation using (4.17), (4.18), and (4.24) can be expressed as

$$\begin{aligned} & x_k^\top P^{j+1} x_k - x_{k+1}^\top P^{j+1} x_{k+1} \\ =& x_k^\top Q x_k + x_k^\top (K^j)^\top R K^j x_k - (u_k + K^j x_k)^\top B^\top \\ & \times P^{j+1} x_{k+1} - (u_k + K^j x_k)^\top B^\top P^{j+1} A_f x_k. \end{aligned} \tag{4.25}$$

The off-policy Bellman equation in (4.25) is rewritten as

$$\begin{aligned} & (x_k^\top \otimes x_k^\top)\mathrm{vec}(P^{j+1}) - (x_{k+1}^\top \otimes x_{k+1}^\top)\mathrm{vec}(P^{j+1}) \\ & + 2(x_k^\top \otimes (u_k + K^j x_k)^\top)\mathrm{vec}(B^\top P^{j+1} A) \\ & - ((K^j x_k - u_k)^\top \otimes (u_k + K^j x_k)^\top)\mathrm{vec}(B^\top P^{j+1} B) \\ & = x_k^\top Q x_k + x_k^\top (K^j)^\top R K^j x_k. \end{aligned} \tag{4.26}$$

The LS solution of (4.26) for P^{j+1} and K^{j+1} can be found without using system dynamics. One defines (δ^j, κ^j) as

$$\delta^j = \begin{bmatrix} x_k^\top Q x_k + x_k^\top (K^j)^\top R K^j x_k \\ \vdots \\ x_{k+\zeta-1}^\top Q x_{k+\zeta-1} + x_{k+\zeta-1}^\top (K^j)^\top R K^j x_{k+\zeta-1} \end{bmatrix}, \tag{4.27}$$

$$\kappa^j = \begin{bmatrix} \rho_{xx1} & \rho_{xu1} & \rho_{uu1} \\ \vdots & \vdots & \vdots \\ \rho_{zz\zeta} & \rho_{zu\zeta} & \rho_{uu\zeta} \end{bmatrix}, \tag{4.28}$$

where $z = 1, 2, \ldots, \zeta$, and

$$\rho_{xx_z} = (x_{k+z-1}^\top \otimes x_{k+z-1}^\top) - (x_{k+z}^\top \otimes x_{k+z}^\top), \tag{4.29a}$$

$$\rho_{xu_z} = 2(x_{k+z-1}^\top \otimes (u_{k+z-1} + K^j x_{k+z-1})^\top), \tag{4.29b}$$

$$\rho_{uu_z} = -(K^j x_{k+z-1} - u_{k+z-1})^\top \otimes (u_{k+z-1} + K^j x_{k+z-1})^\top. \tag{4.29c}$$

Note that in (4.29), $\rho_{xx_z} \in \mathbb{R}^{1\times n^2}, \rho_{xu_z} \in \mathbb{R}^{1\times nm}$, and $\rho_{uu_z} \in \mathbb{R}^{1\times m^2}$. Moreover,

$$((\kappa^j)^\top \kappa^j)^{-1} (\kappa^j)^\top \delta^j = \begin{bmatrix} \mathrm{vec}(P^{j+1})^\top \\ \mathrm{vec}(B^\top P^{j+1} A)^\top \\ \mathrm{vec}(B^\top P^{j+1} B)^\top \end{bmatrix}. \tag{4.30}$$

$$K^{j+1} = \left(R + B^\top P^{j+1} B\right)^{-1} B^\top P^{j+1} A. \tag{4.31}$$

Note that (4.30) has $\Xi = n^2 + m^2 + nm$ unknown parameters. The matrix P^{j+1} can be uniquely solved with LS in (4.30) while satisfying the full-rank condition, i.e., $\mathrm{rank}((\kappa^j)^\top \kappa^j) = \Xi$ and a minimum of $\eta \geq \Xi$ samples are needed at each iteration j. Using solutions in (4.30), the feedback gain K^{j+1} can be found by policy improvement in (4.31). The system (4.24) should be persistently excited [69, 84], which is guaranteed by adding probing noises to the control input u_k in (4.24).

Lemma 3. See [69]. Consider a discrete-time system in (4.9) and J in (4.8). Assume the LS solution to policy evaluation (4.30) has a full rank of $((\kappa^j)^\top \kappa^j)$ and, at a minimum, has $\eta \geq \Xi$ samples at each iteration. Then, the following claims hold.

1. $A - BK^j$ is Hurwitz for each $j = 0, 1, \ldots,$ where K^j follows from the solution of (4.31).
2. $\lim_{j\to\infty} P^j = P$, $\lim_{j\to\infty} K^j = K^*$, and control $u_k^* = -K^* x_k$ minimizes J.

Remark 23. Initial control gain matrix K^0 for model-free implementation of PI by merely following (4.20) and (4.21) is identified through experience as there could be infinitely many stabilizing policies [22, 35, 57]. Thus, one should experiment by trying different matrices Q and R in (4.21).

Model-free RL version is presented to show datasets (δ^j, κ^j) generated from interactions with the environment are exploited to find the optimal policy u_k^*. Next, we propose that iteratively-learned datasets (δ^j, κ^j) in (4.27) and (4.28) can be decomposed to reduce the computing burden on the RL. This enhances the RL algorithm's data efficiency to provide optimal control solutions for large-scale systems.

4.4 Data-efficient Discrete-Time RL Algorithm

We determine the Koopman eigenfunctions for effective decomposition in the model-free version presented in Section 4.3.2. This reduces the computational complexity of the RL algorithm for the nonlinear system (4.1).

4.4.1 Lossless Model Reduction

The lossless model reduction technique finds optimal low-ranking decomposition for linear systems [97]. In performing decomposition, we find a projection matrix that transforms a high-dimensional space into a lower one by seeking a balanced realization of (4.9), in which the system is controllable.

The discrete controllability Gramian $\Phi_c = \sum_{m=0}^{\infty} A^m B B^\top (A^\top)^m$ provides a measure of the control $u(\cdot)$ propagating the state $x(\cdot)$ in (4.9) [136]. The balanced proper orthogonal

decomposition (POD) [116] is developed to avoid the computation of Φ_c. Alternatively, it uses state data matrices in (4.10) for the empirical controllability Gramian, making balanced POD suitable for high-dimensional systems.

Given the discretized measurement $x_k \in \mathbb{R}^n$, a lower-dimensional state measurement $\zeta_k \in \mathbb{R}^r$ is induced by projecting x_k through a projection matrix P_r

$$\zeta_k \approx P_r x_k, \tag{4.32}$$

where the projection matrix $P_r \in \mathbb{R}^{r\times n}$ can be computed by the following standard minimization problem [67],

$$\min_{P_r\in\mathbb{R}^{r\times n}} \sum_{j=0}^{N-1} \left\| (I - P_r^{\dagger} P_r) x_k \right\|^2, \quad r << n. \tag{4.33}$$

The computation of Koopman modes without the optimal low-ranking reduction is costly. Approximating the $\mathscr{K}$ with eDMD is possible. eDMD identifies spatiotemporal patterns in measurements that help obtain Koopman eigenfunctions. These eigenfunctions decompose the data vectors in (4.29) onto a lower-dimensional space, while retaining the complete dynamical information of the original system. The solution to (4.33) with the balanced POD modes can be found using eDMD, where $P_r^{\dagger} = \hat{U}$. The matrix $\hat{U}$ is specified by the first r columns of the left-singular matrix in SVD of X_2. These columns are POD modes that are orthonormal [54]. Thus, $\hat{U}$ is the unitary matrix, where $\hat{U}^{\iota}\hat{U} = I$. First, SVD reduces Y in (4.15) to rank r,

$$Y \approx \tilde{Y} = \tilde{U}\tilde{S}\tilde{V}^{\iota}, \tag{4.34}$$

where $\tilde{U} \in \mathbb{R}^{(n+m)\times r}$, $\tilde{S} \in \mathbb{R}^{r\times r}$, and $\tilde{V} \in \mathbb{R}^{(p-1)\times r}$. Note that the diagonal matrix S_k in (4.15) holds singular values (σ_i) of Y.

To ensure that a specific amount of energy from the original data is captured in the low-ranking approximation in subspace $\mathbb{R}^r \subset \mathbb{R}^{(n+m)}$, one can apply a strict threshold μ on S_k to retain the first singular elements [38]. The percentage energy of each σ is given by

$$E_{\sigma_i} = \frac{\sigma_i}{\sum_{i=1}^{n+m} \sigma_i}. \tag{4.35}$$

The value of r can be any positive integer between 1 to $(n+m)$ for the matrix S_k, determined by a strict threshold $\mu \geq 0$, with the constraint that $\sigma_i \geq \mu \sum_{i=1}^{n+m} \sigma_i$. This thresholding keeps r leading σs with high energy and eliminates remaining σs with lower energy, i.e., $r+1$ to $(n+m)$. The SVD in (4.34) provides a truncated SVD of (4.15) by eliminating nonessential singular values in S_k such that $\tilde{S}$ is a square matrix. This enables pseudoinverse of $\tilde{Y}$ to modify G in (4.16) as

$$\tilde{G} = X_2 \tilde{V} \tilde{S}^{-1} \tilde{U}^\iota = X_2 \tilde{V} \tilde{S}^{-1} \begin{bmatrix} U_1^\iota \\ U_2^\iota \end{bmatrix}, \tag{4.36}$$

where $U_1 \in \mathbb{R}^{n \times r}$ and $U_2 \in \mathbb{R}^{m \times r}$. Note that $\tilde{G} \in \mathbb{R}^{n \times (n+m)}$ has the same dimension as G. The SVD of X_2 is

$$X_2 \approx \tilde{X_2} = \hat{U} \hat{S} \hat{V}^\iota, \tag{4.37}$$

where $\hat{U} \in \mathbb{R}^{n \times r}$, $\hat{S} \in \mathbb{R}^{r \times r}$, and $\hat{V} \in \mathbb{R}^{(p-1) \times r}$.

The dynamic model using matrices in (4.17) provides a high dimensional system, where $n >> 1$. The data-efficient model of rank $r << n$ can be formed via projection as shown in [67] that holds the same form as (4.32), where $P_r = \hat{U}^\iota$. Then, the reduced-order system matrices become

$$\tilde{A} = P_r A P_r^\dagger = \hat{U}^\iota X_2 \tilde{V} \tilde{S}^{-1} U_1^\iota \hat{U} \tag{4.38a}$$

$$\tilde{B} = P_r B = \hat{U}^\iota X_2 \tilde{V} \tilde{S}^{-1} U_2^\iota, \tag{4.38b}$$

where $\tilde{A} \in \mathbb{R}^{r\times r}$ is Hurwitz and $\tilde{B} \in \mathbb{R}^{r\times m}$. The resulting data-driven dynamic model is

$$\tilde{X}_2 = \tilde{G}\tilde{Y} = \begin{bmatrix} \tilde{A} & \tilde{B} \end{bmatrix} \tilde{Y}. \tag{4.39}$$

The following lemma implies that the optimal control learning for the system in (4.9) with J in (4.8) is analogous to the learning for r-dimensional dynamic model using ζ_k.

Lemma 4. [136] The r-dimensional dynamic model, satisfied by system dynamics in (4.9), and ζ_k in (4.32), is

$$\zeta_{k+1} = \tilde{A}\zeta_k + \tilde{B}u_k. \tag{4.40}$$

Also, $x_k \approx P_r^{\dagger}\zeta_k$ holds for any u_k, x_k. The r-dimensional state ζ_k retains the behavior of $x_k \in \mathbb{R}^n$, if P_r satisfies (4.33). Then, the performance index concerning ζ_k becomes

$$\bar{J}(u_k, \zeta_k) = \sum_{k=0}^{\infty} \left[\zeta_k^{\top}\bar{Q}\zeta_k + u_k^{\top}Ru_k\right], \tag{4.41}$$

where

$$\bar{Q} = (P_r^{\dagger})^{\top} Q P_r^{\dagger} \geq 0 \in \mathbb{R}^{r\times r}. \tag{4.42}$$

4.4.2 Extraction of Koopman Eigenfunctions

We use eDMD to find Koopman eigenfunctions of the original system from the reduced-dimensional dynamic model. It is easy to see from (4.6) and (4.7) that a data-driven system can be decomposed into the Koopman modes, where Koopman eigenfunctions characterize the temporal nature of respective Koopman modes. Eigendecomposition of $\tilde{A}$ generates eigenvalues λ and eigenvectors v that can examine essential characteristics of the system [136]. Eigenvalue analysis is vital to fully extract dynamic information from system measurements in the extended state space.

Remark 24. It is clear from (4.39) that eigenvalues and eigenvectors of $\tilde{A}$ coincide with $\tilde{G}$. Eigenvalues of $\tilde{G}$ and G are analogous. Also, the eigenvalue analysis of $\tilde{A}$ is more feasible than that of $\bar{A}$, where $r << n$.

Lemma 5. See [117]. Given the linear system in (4.14) and (4.39), eigenvalues of the linear operator $\mathscr{K}$ are eigenvalues of $\tilde{A}$, and eigenfunctions of $\mathscr{K}$ coincide with eigenvectors of $\tilde{A}$.

Proof. Consider the eigendecomposition of $\tilde{A}$,

$$\tilde{A}v_i = \lambda_i v_i, \quad i = 1, \ldots, n, \tag{4.43}$$

where λ_i and v_i are eigenvalues and eigenvectors of $\tilde{A}$. Let w_i denotes eigenfunctions of the adjoint $\tilde{A}^\iota$ defined on $\mathbb{R}^n$ linear space so that $\langle v_i, w_h \rangle = \rho_{ih}$,

$$\tilde{A}^\iota w_i = \bar{\lambda}_i w_i, \quad i, h = 1, \ldots, n. \tag{4.44}$$

Given Remark 24, one defines eigenfunctions as $\theta_i(x_k) = \langle x_k, w_i \rangle$, where

$$\begin{aligned} \mathscr{K}\theta_i(x_k) &= \theta_i(\tilde{A}x_k) = \langle \tilde{A}x_k, w_i \rangle = \langle x_k, \tilde{A}^\iota w_i \rangle \\ &= \lambda_i \langle x_k, w_i \rangle = \lambda_i \theta_i(x_k). \end{aligned} \tag{4.45}$$

Given that $\tilde{A}$ has a full set of eigenvectors for any x_k, one has

$$h(x_k) = \sum_{i=1}^{n} \langle x_k, w_i \rangle v_i = \sum_{i=1}^{n} \theta_i(x_k) v_i, \tag{4.46}$$

$$\begin{aligned} \mathscr{K}h(x_k) &= \mathscr{K} \sum_{i=1}^{n} \theta_i(x_k) v_i = \sum_{i=1}^{n} \mathscr{K}\theta_i(x_k) v_i \\ &= \sum_{i=1}^{n} \lambda_i \theta_i(x_k) v_i. \end{aligned} \tag{4.47}$$

Thus, the eigenfunctions of $\mathscr{K}$ coincide with the eigenvectors of $\tilde{A}$. □

Remark 25. Note that the infinite-dimensional Koopman operator $\mathscr{K}$ has an infinite number of eigenvalues since λ_{mi} is also an eigenvalue corresponding to the eigenfunction $\theta_{mi}(x_k, u_k)$.

Given the sequences in (4.10), (4.11), and (4.13) having $(p-1)$ snapshots sampled at interval t_k, the linear operator G in (4.14) drives the evolution of system forward in time. It is clear from Lemma 5 and Remark 24 that the eigenvalues of the operator A are also the eigenvalues of the operator $\mathscr{K}$. Moreover, for observable $h(x_k) = x_k$, the eigenvectors of A coincide with the Koopman modes. Given that X_2 in (4.10) and Y in (4.13) are sufficiently long, the last snapshot x_p can be approximated by a linear combination of snapshots up to y_{p-1} using (4.14)

$$X_2^p = GY_1^{p-1} = Y_1^{p-1}C + se_{p-1}^\top, \tag{4.48}$$

where C is a companion matrix, $e_{p-1} \in \mathbb{R}^{p-1}$ is $(p-1)$th unit vector, and s is surplus such that $s \neq 0$ if the linear character is not satisfied [138]. Thus, C satisfying (4.48) is not unique and one solution will be $C = X_2^p (Y_1^{p-1})^\dagger$ if Y_1^{p-1} has linearly independent columns, i.e., $s = 0$. Then, $S(C)$ coincides with $S(G)$ and $S(\mathscr{K})$.

The columns of θ are approximated eigenvectors of $A \in \mathbb{R}^{n\times n}$ [125]. Koopman eigenfunctions $\theta \in \mathbb{R}^{n\times r}$ are

$$\theta = X_2 \tilde{V} \tilde{S}^{-1} U_1^l \hat{U} v = X_2 \tilde{V} \tilde{S}^{-1} U_1^l P_r^\dagger v. \tag{4.49}$$

The eDMD constructs Koopman eigenfunctions θ using a finite-rank approximation in Remark 24. It can be used for data preconditioning in model-free RL. This reduces the dimensionality of data vectors. The updated vectors are computed from (4.29) and (4.49) as

$$\hat{\rho}_{xx} = \rho_{xx_z}(\theta^\top \otimes \theta^\top) \in \mathbb{R}^{1\times r^2}, \tag{4.50a}$$

$$\hat{\rho}_{xu} = \rho_{xu_z}(I \otimes \theta^\top) \in \mathbb{R}^{1\times rm}, \tag{4.50b}$$

$$\hat{\rho}_{uu} = \rho_{uu_z}(I \otimes I) \in \mathbb{R}^{1\times m^2}. \tag{4.50c}$$

In the following section, we propose a model-free RL algorithm for (4.1), that is data-efficient, via Koopman-preconditioning of data vectors.

4.4.3 Data-efficient Model-free Discrete-time RL Algorithm

In contrast to the model-free version in Section 4.3.2, Algorithm 7 additionally computes Koopman eigenfunction to transform data vectors to lower dimensions such that dynamic information of the original system is conserved. The evaluation and improvement of policy at each iteration are identical to the model-free version in Section 4.3.2. However, we are using the lower-dimensional state ζ_k shown in (4.32), which improves data efficiency.

The operator κ_r^j in (4.52) is updated from κ^j in (4.28) using θ in (4.49). At each iteration j, collected operators δ_r^j and κ_r^j, from (4.51) and (4.52), give P^{j+1} in (4.53).

Remark 26. Note that in (4.52), $\hat{\rho}_{xx} \in \mathbb{R}^{1\times r^2}, \hat{\rho}_{xu} \in \mathbb{R}^{1\times rm}$, and $\hat{\rho}_{uu} \in \mathbb{R}^{1\times m^2}$. Thus, (4.53) has $\Xi_r = r^2 + m^2 + rm$ unknown parameters, where $r << n$. The matrix $\bar{P}^{j+1}$ can be uniquely solved with LS in (4.53) while satisfying the full-rank condition of $((\kappa_r^j)^\top \kappa_r^j)$, and a minimum of $\eta \geq \Xi_r$ samples are needed at each iteration j.

Given Remark 26, it is clear that Algorithm 7 requires fewer data samples η, i.e., $\Xi_r << \Xi$, at each iteration j, than the model-free version presented in Section 4.3.2. This shows the data efficiency of the proposed RL Algorithm 7. Using solutions in (4.53), $\bar{K}^{j+1}$ can be found by policy improvement in (4.54) and feedback gain K^{j+1} computed in (4.55). Similar to Lemma 3, the convergence of Algorithm 7 is shown as follows.

Theorem 9. *Consider a discrete-time system in* (4.9)*, J in* (4.8)*, and a projection that satisfies* (4.33)*. Assume that the LS solution to policy evaluation in* (4.53) *has a full rank of $((\kappa_r^j)^\top \kappa_r^j)$ and, at a minimum, has $\eta \geq \Xi_r$ samples at each iteration. Then, the following claims hold.*

1. *$A - BK^j$ is Hurwitz for each $j = 0, 1, \ldots$, where K^j follows from the solution of* (4.55).
2. *$\lim_{j\to\infty} P^j = P$, $\lim_{j\to\infty} K^j = K^*$, and control $u_k^* = -K^* x_k$ minimizes J.*

Algorithm 7 Data-Efficient Model-Free Discrete-Time RL Algorithm

1: **Require:** Collected behavioral trajectories (x_k, u_k) of (4.1).

2: **Initialization:** Set $j = 0$ and small threshold e.

3: **Koopman:**

3.1 Input: Data matrices X_1, X_2, U and strict threshold μ.

3.2 Augment: Same as step 3.2 in Algorithm 6.

3.3 Compute SVD: $(U_k, S_k, V_k) \leftarrow Y$ & $(U, S, V) \leftarrow X_2$.

3.4 Truncation: $(\tilde{U}, \tilde{S}, \tilde{V}) \leftarrow \tilde{Y}$ & $(\hat{U}, \hat{S}, \hat{V}) \leftarrow \tilde{X}_2$.

3.5 Split: $U_1, U_2 \leftarrow \tilde{U}$.

3.6 Output: Identify θ as (4.49).

4: Select stabilizing K^0 and control input $u_k = -Kx_k + \varepsilon_k$, where ε_k is a probing noise.

5: **Datasets Collection**: Collect $(\rho_{xx_z}, \rho_{xu_z}, \rho_{uu_z})$ in (4.29).

6: **Datasets truncation**: Compute reduced-order data vectors $(\hat{\rho}_{xx}, \hat{\rho}_{xu}, \hat{\rho}_{uu})$ as (4.50), and operators (δ_r^j, κ_r^j) given by

$$\delta_r^j = \begin{bmatrix} \zeta_k^\top \bar{Q} \zeta_k + \zeta_k^\top (\bar{K}^j)^\top R \bar{K}^j \zeta_k \\ \vdots \\ \zeta_{k+\eta-1}^\top \bar{Q} \zeta_{k+\eta-1} + \zeta_{k+\eta-1}^\top (\bar{K}^j)^\top R \bar{K}^j \zeta_{k+\eta-1} \end{bmatrix}, \tag{4.51}$$

$$\kappa_r^j = \begin{bmatrix} \hat{\rho}_{xx} & \hat{\rho}_{xu} & \hat{\rho}_{uu} \end{bmatrix}. \tag{4.52}$$

7: **Policy Evaluation**: Compute $\bar{P}^{j+1}$ by

$$((\kappa_r^j)^\top \kappa_r^j)^{-1} (\kappa_r^j)^\top \delta_r^j = \begin{bmatrix} \mathrm{vec}(\bar{P}^{j+1})^\top \\ \mathrm{vec}((P_r B)^\top \bar{P}^{j+1} P_r A P_r^\dagger)^\top \\ \mathrm{vec}((P_r B)^\top \bar{P}^{j+1} P_r B)^\top \end{bmatrix}. \tag{4.53}$$

8: **Policy Improvement**: Compute $\bar{K}^{j+1}$ by

$$\bar{K}^{j+1} = \left(R + (P_r B)^\top \bar{P}^{j+1} P_r B\right)^{-1} (P_r B)^\top \bar{P}^{j+1} (P_r A). \tag{4.54}$$

9: **Stop if** $\|\bar{K}^{j+1} - \bar{K}^j\| \leq e$; Otherwise, set $j = j+1$, and go to step 2.

10: **Return**:

$$K^{j+1} = P_r \bar{K}^{j+1}. \tag{4.55}$$

Proof. From Lemma 4, it is seen that the unique stabilizing control policy $u_k = -\bar{K}\zeta_k$ minimizes $\bar{J}(u_k, \zeta_k)$ in (4.41) as

$$\begin{aligned} \bar{J}^*(\zeta_k) = \min_{u_k} \bar{J}(u_k, \zeta_k) &= \sum_{k=0}^{\infty} \left[\zeta_k^\top \bar{Q} \zeta_k + u_k^\top R u_k\right] \\ &= \zeta_k^\top \bar{P} \zeta_k, \end{aligned} \tag{4.56}$$

where $\bar{P} \in \mathbb{R}^{r \times r} > 0$ is a truncated cost matrix. The matrix $\bar{P}$ spans analogous eigenvalues as P, i.e., $S(\bar{P}) = S(P_r P P_r^\dagger)$. The (4.56) provides that $P_r A P_r^\dagger - P_r B \bar{K}^j$ for each $j = 0, 1, \ldots$ is Hurwitz. Since a P_r satisfies (4.33), then, $A - BK^j$ is Hurwitz.

Given Lemma 3, the lossless reduced-dimensional off-policy RL can be obtained, where $\bar{K} = P_r^\dagger K \in \mathbb{R}^{m \times r}$ is

$$\bar{K}^* = (R + (P_r B)^\top \bar{P}^* (P_r B))^{-1} (P_r B)^\top \bar{P}^* (P_r A), \tag{4.57}$$

and $\bar{P} = P_r P P_r^\dagger > 0 \in \mathbb{R}^{r \times r}$ satisfies

$$\begin{aligned} \bar{P}^* = (P_r A)^\top \bar{P}^* (P_r A) &- (P_r A)^\top \bar{P}^* (P_r B) \\ &\times (R + (P_r B)^\top \bar{P}^* (P_r B)^{-1}) (P_r B)^\top \bar{P}^* (P_r A) + \bar{Q}. \end{aligned} \tag{4.58}$$

The LS solution of (4.53) for $\bar{P}^{j+1}$ and $\bar{K}^{j+1}$ can be found without knowing system dynamics by defining (δ_r^j, κ_r^j) as in (4.51) and (4.52). Given Remark 23, by considering $\bar{K}^0$ as an

initial gain matrix, $(\bar{P}^{j+1}, \bar{K}^{j+1})$ is uniquely obtained by LS in (4.53), while satisfying the full-rank condition of $((\kappa_r^j)^\top \kappa_r^j)$, which is derived from data matrices in (4.50). □

As shown in Section 4.3.2, off-policy RL becomes computationally intractable as it requires learning from different behavior policies obtained from datasets generated during interactions with the environment. To address this challenge, we have combined eDMD for Koopman approximation and off-policy RL to design an efficient optimal control strategy for complex nonlinear systems with unknown models.

Remark 27. Note that we use the eDMD method to find the best-fit linear model of an unknown nonlinear system in (4.1). Provided with the derived linear model, the LQR control of (4.1) finds the optimal control input u_k^* in (4.19). Then, in Lemma 3 and Theorem 9, claim 1) demonstrates stability, while claim 2) establishes the optimality of the RL algorithm solving the LQR control of (4.1).

Remark 28. According to Section 4.3.2, model-free RL requires n^2+m^2+nm data samples to obtain a unique solution, whereas Algorithm 7 needs only r^2+m^2+rm samples. The computational complexity of LS, with η as data samples and Ξ representing unknown parameters, for $\eta \geq \Xi$, is of the order $O(\Xi^2\eta)$. When applying policy iteration for optimal control, the model-free version in Section 4.3.2 has a complexity of $O\big((n^2+m^2+nm)^2\eta\big)$, whereas Algorithm 7 has lower complexity of $O\big((r^2+m^2+rm)^2\eta\big)$, given $r << n$. Thus, Algorithm 7 has more computational tractability and conserves a significant learning time when r is small enough.

4.5 Simulation Studies

We evaluate Algorithm 7 using an example of excitation control of a power grid under a three-phase fault [48]. The grid dynamics can be expressed by (4.1), where x_k denotes

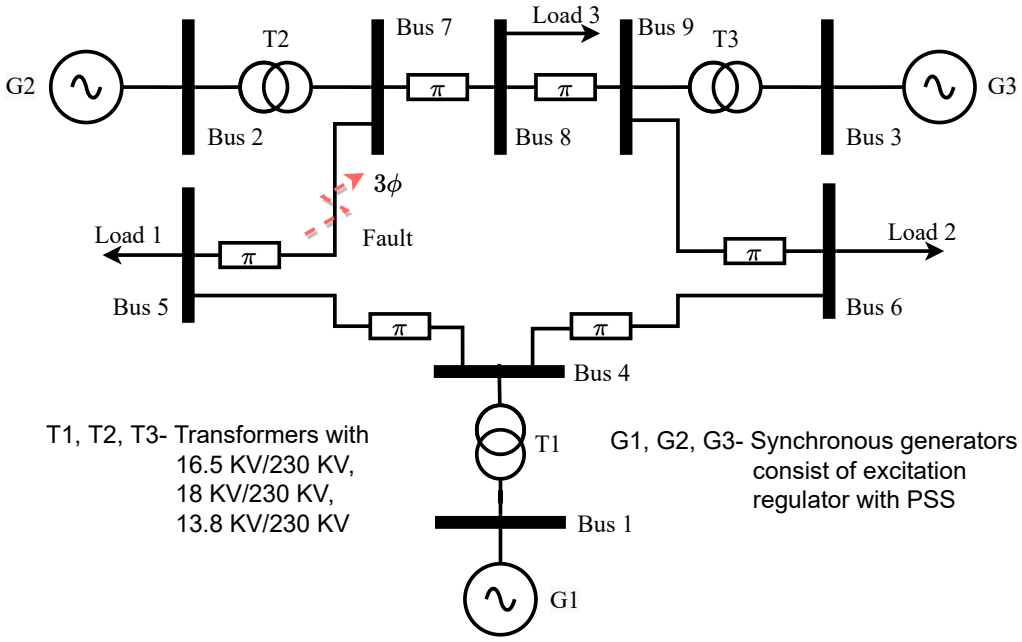


Figure 4.2. IEEE 9-bus network schematic is modified from [7] with three synchronous generators, with a three-phase fault happening between buses 5 and 7..

states and u_k denotes control inputs of each generator. Similar to the example in [48], we consider the state and control data trajectories as

$$h(x_k) = \left[\delta,\ \Delta\omega,\ V_t,\ P_t,\ Q_t\right]^\top, u_k = \left[P_{ref},\ E_f\right]^\top, \tag{4.59}$$

where $\Delta\omega$ is the generators' rotor speed deviation in per-unit (pu). V_t, P_t, Q_t are terminal voltage, active power, and reactive power in pu, respectively. δ in degrees is the rotor angle, P_{ref} is the reference power of turbine-governor in pu, and E_f is the excitation field in pu.

Figure 4.2 shows the schematic diagram of the modified IEEE 9-bus system, where all generators are equipped with an automatic voltage regulator (AVR) to maintain E_f, regulate the terminal voltage V_t at V_{ref}, and $\Delta\omega$-power system stabilizer (PSS) provides extra damping [1]. The buses are connected through a 100 Km, three-phase π-section transmission line. All computations are executed on an Intel(R) Xeon(R) W-10855M 2.80 GHz, 32 GB RAM, with MATLAB 2021a. We collect data trajectories $(h(x_k), u_k)$, by implementing

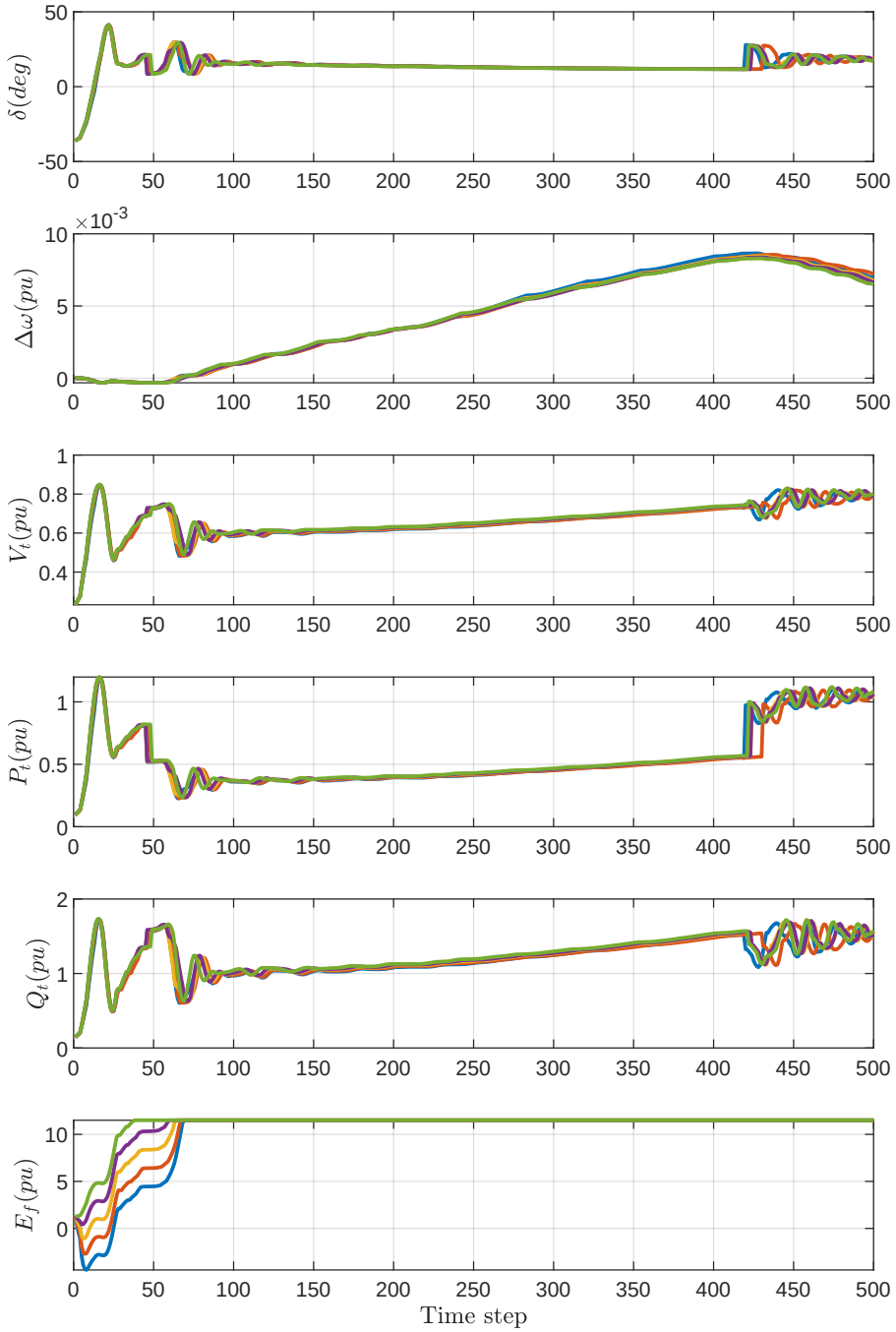


Figure 4.3. Data trajectories of generator 1 under three-phase fault..

the system in Figure 4.2 in the Simulink environment, to evaluate the proposed Algorithm 7. Therein, $(h(x_k), u_k)$ are directly accessible from simulation experiments as a vector of

observables. Therefore, there is no need to explicitly define the specific form of $h(x)$. The trajectories are collected by randomly selecting V_{ref} in the interval $[0.9, 1]$ and $P_{ref} = 1$, over 5 spaced points under a three-phase fault between buses 5 and 7.

Figure 4.3, 4.4, and 4.5 display collected data trajectories of generators (G1, G2, G3). The breaker closes after 18 cycles to clear the fault. As shown in Section 4.2.2, eDMD operates on collected data of three generators, each having 5 state and 2 control data trajectories with 5 spaced points over $k = \{0, \ldots, 842\}$, i.e., $X_2, X_1 \in \mathbb{R}^{75\times 842}$, $U \in \mathbb{R}^{30\times 842}$ and $Y \in \mathbb{R}^{105\times 842}$.

Figure 4.6 shows singular values captured by the lossless model reduction. As shown in Section 4.4.1, strict threshold μ on singular values for a balanced POD via eDMD captures the first 20 dominant modes. The low-dimensional approximation $\tilde{Y}$ to rank 20 in (4.34) gives $\tilde{U} \in \mathbb{R}^{105\times 20}$, $\tilde{S} \in \mathbb{R}^{20\times 20}$, and $\tilde{V} \in \mathbb{R}^{842\times 20}$. Similarly, low-dimensional approximation $\tilde{X_2}$ to rank 20 in (4.37) gives $\hat{U} \in \mathbb{R}^{75\times 20}$, $\hat{S} \in \mathbb{R}^{20\times 20}$, and $\hat{V} \in \mathbb{R}^{842\times 20}$. This provides matrix $\tilde{G} \in \mathbb{R}^{75\times 105}$ as in (4.36) where $U_1 \in \mathbb{R}^{75\times 20}$ and $U_2 \in \mathbb{R}^{30\times 20}$. Koopman eigenfunction $\theta \in \mathbb{R}^{75\times 30}$ is computed as in (4.49). The eigenvalues of the Koopman operators are shown in Fig. 4.7. Note that the computed Koopman operator approximations are verified by Fig. 4.7. Dominant eigenvalues are consistent throughout the $G, \bar{A}, \tilde{A}$ matrices. This implies that the long-term behavior will be the same for approximations of the Koopman operators.

One can select a penalizing state weight $Q = 0.01 \times I_{75} \in \mathbb{R}^{75\times 75}$ for (4.8). The resulting lower-dimensional state weight in (4.42) becomes $\bar{Q} \in \mathbb{R}^{20\times 20}$. In step 3 of Algorithm 7, consider the probing noise $\varepsilon_k = 0.001 \times rand(1)$. Compute data vectors ($\rho_{xx_z} \in \mathbb{R}^{1\times 5625}, \rho_{xu_z} \in \mathbb{R}^{1\times 2250}, \rho_{uu_z} \in \mathbb{R}^{1\times 900}$) in step 4 of Algorithm 7 as in (4.29). We use Koopman eigenfunction θ in (4.49) to update $(\rho_{xx_z}, \rho_{xu_z}, \rho_{uu_z})$ to lower dimensions, i.e., $(\hat{\rho}_{xx}, \hat{\rho}_{xu}, \hat{\rho}_{uu})$ in (4.50), to conclude Koopman-preconditioning in step 5 of Algorithm 7.

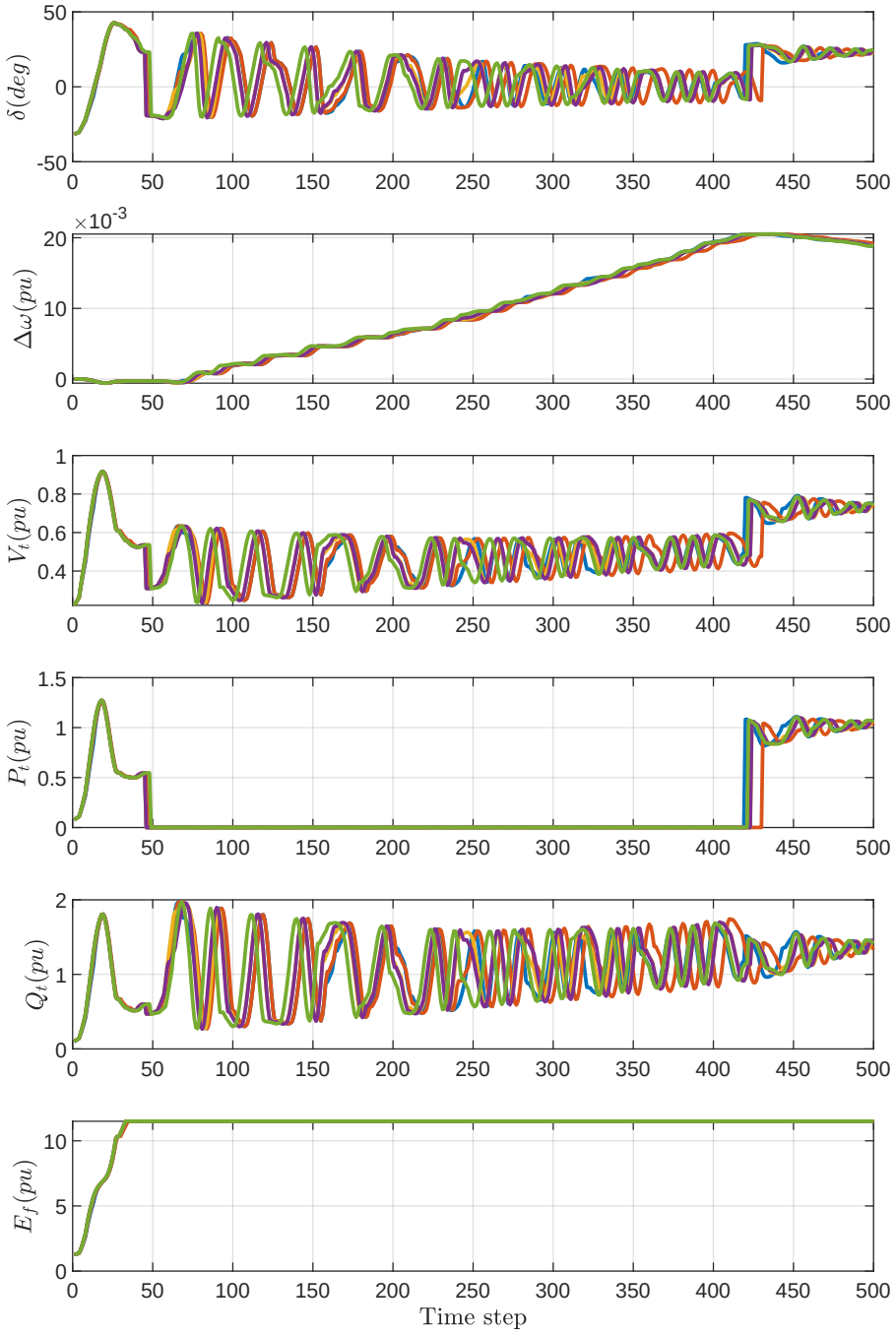


Figure 4.4. Data trajectories of generator 2 under the three-phase fault..

Figure 4.9 shows the convergence of performance index J in (4.8) and $\bar{J}$ in (4.41) for state matrices of dimensions 75 and 20, respectively. The feedback gain $\bar{K}$ for the lower

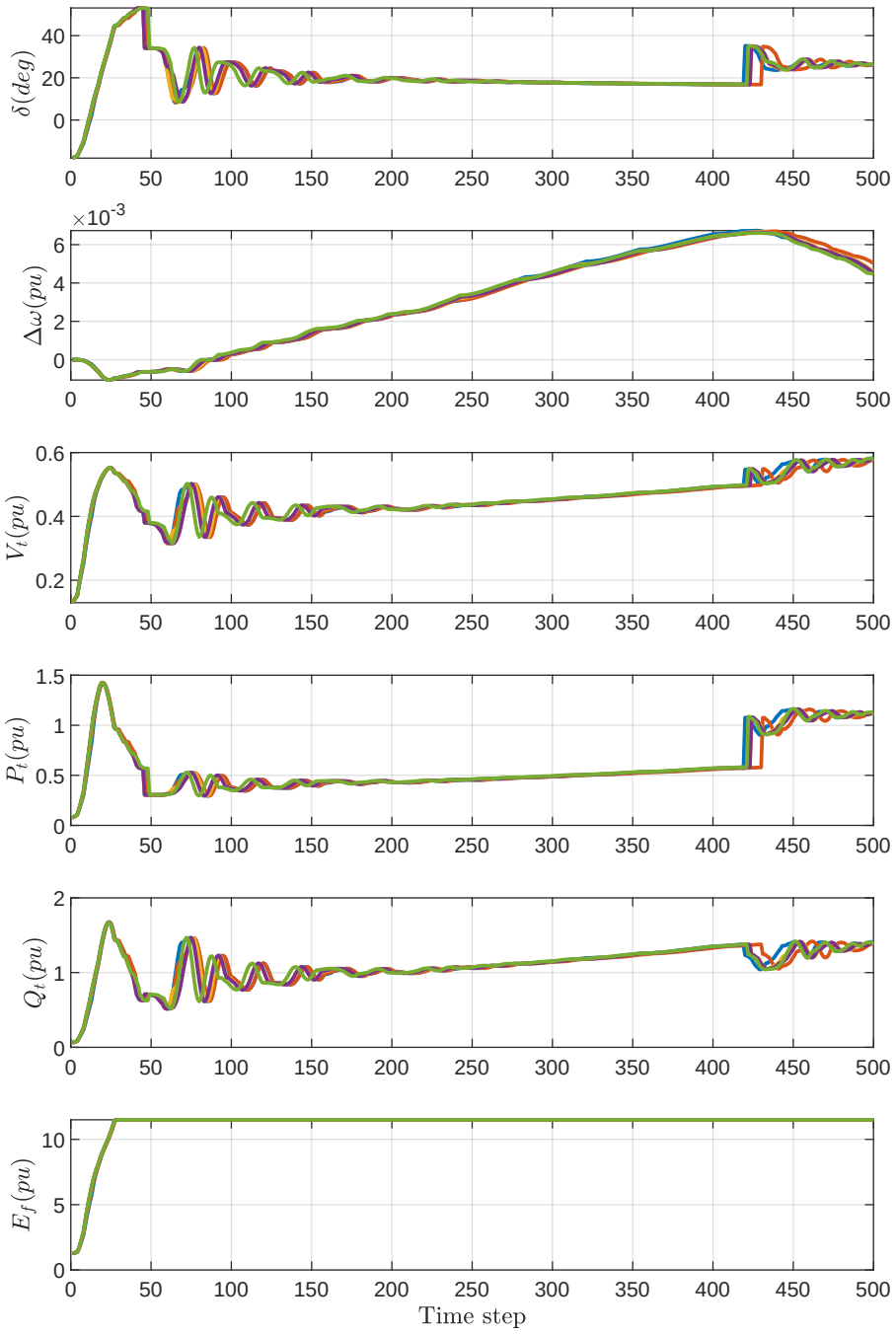


Figure 4.5. Data trajectories of generator 3 under the three-phase fault..

dimension is computed in (4.54). Figure 4.8 shows the convergence of feedback gain $\bar{K}$. Finally, $\bar{K}$ converges to

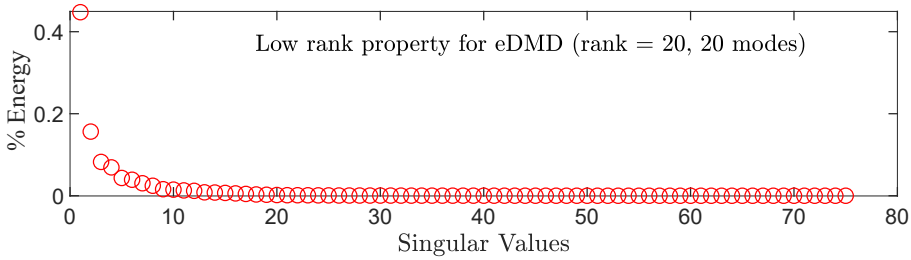


Figure 4.6. Singular values of X_2 with percentage of energy in each mode..

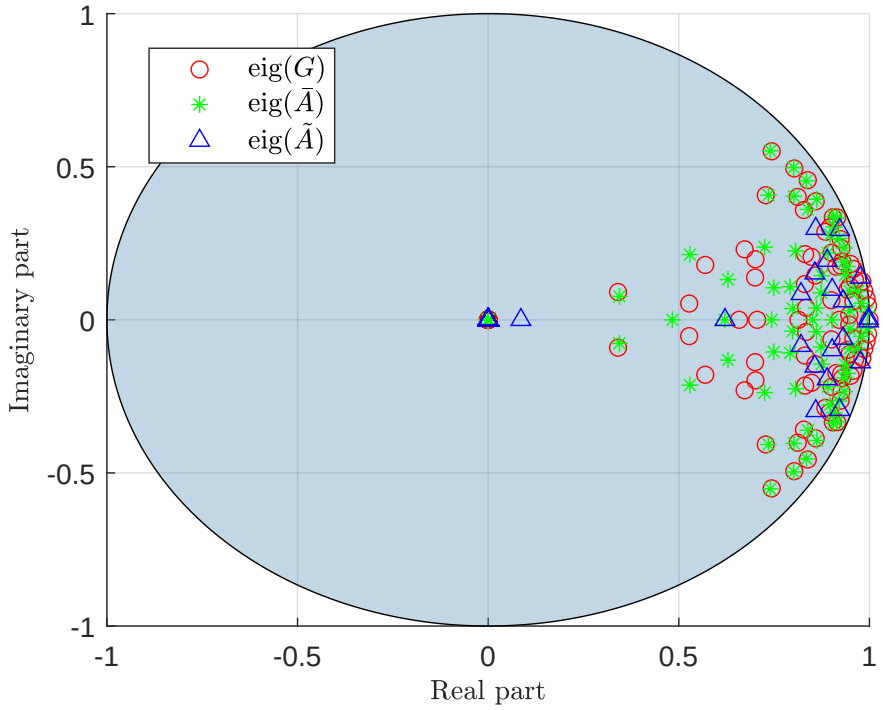


Figure 4.7. Eigenvalues of $G, \bar{A}, \tilde{A}$ matrices..

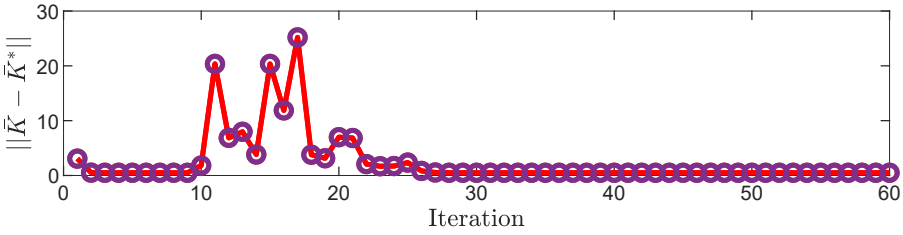


Figure 4.8. Convergence of the feedback gain $\bar{K}$..

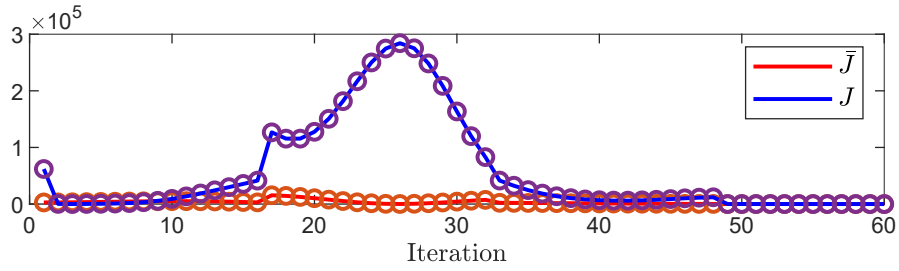


Figure 4.9. Convergence of the performance index J..

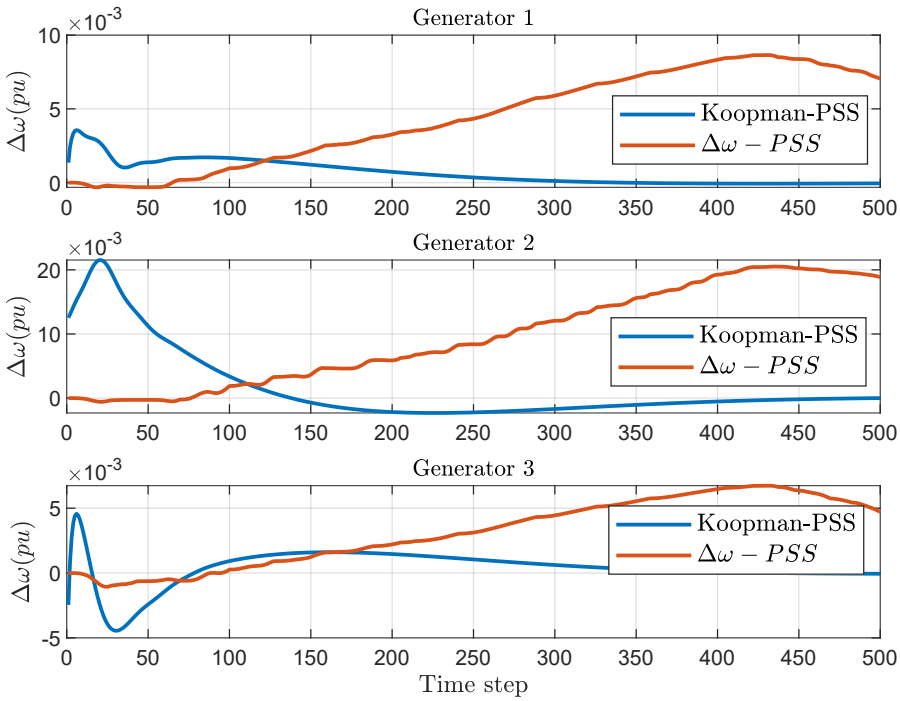


Figure 4.10. Speed deviations of generators under three-phase fault..

$$\bar{K}^* = \Big[0.0022 \quad -0.0056 \quad -0.0033 \quad -0.0020 \quad 0.0071 \quad -0.0216$$

$$-0.0065 \quad 0.0495 \quad 0.0052 \quad 0.0027 \quad 0.0309 \quad 0.0297 \quad -0.0079$$

$$-0.0087 \quad -0.0588 \quad -0.0438 \quad 0.0839 \quad -0.0629 \quad 0.0625 \quad 0.0496\Big] .$$

In both cases, a minimum of 50 iterations are required. On the contrary, given Q and R, the computation takes 15 seconds for $r = 20$, and for $n = 75$ results in over-fitting due to

potentially near-singular rank matrices for LS in step 7 of Algorithm 7. Alternatively, selecting a different Q and R for larger systems, as shown in Remark 23, is difficult. This results in a longer learning time of three to four minutes for $n = 75$. These results support that the low-ranking decomposition significantly reduces learning time and avoids over-fitting. The model-free RL algorithms in [69, 80, 90] do not address nonlinear multiagent systems. However, our approach overcomes this limitation and finds optimal policies, while maintaining data efficiency and preserving dynamic information for such systems. Figure 4.10 shows the control performance when a three-phase fault occurs. The improved damping effects can be seen in Figure 4.10 with the speed deviations of each generator, where Koopman-based controllers have smoother trajectories with fewer oscillations than $\Delta\omega$-PSS.

4.6 Conclusion & Future Work

The integration of Koopman theory with RL for nonlinear optimal control reduces strenuous system modeling and improves data efficiency. Data-driven Koopman preconditioning transforms nonlinear system models into linear representations. This allows linear optimal control via an entirely data-driven RL algorithm. Simulation studies on nonlinear power system dynamics show the effectiveness of the proposed approach in achieving data efficiency by requiring less data. In future work, incorporating external uncontrolled noise into the data collection process could be explored to enhance the robustness of the proposed approach.

Chapter 5

Efficient Reward Shaping for Multiagent Systems *

*This chapter has been used with permission from all co-authors.

CHAPTER 5

Efficient Reward Shaping for Multiagent Systems

5.1 Introduction

Large-scale multiagent systems (MASs) are becoming prevalent in telecommunication networks, traffic systems, or electrical grids [105, 120, 122]. Linear quadratic regulator (LQR) design for such systems is discussed in [21, 140]. Reinforcement learning (RL) algorithms, such as actor-critic learning [32], Q-learning [56, 91, 100, 145], adaptive dynamic programming [23, 58], or integral learning [130], can solve the resulting LQR design. These RL methods optimize reward functions to learn optimal control policies through interactions with the environment. The reward function guides the learning process by measuring the utility of taking a particular action in a given system state, while the policy determines decision-making strategy based on observed states. RL methods typically assume known reward functions. However, the manual selection of suitable reward functions could become challenging for a large-scale MAS.

Reward shaping introduces additional rewards to guide the learning process beyond the rewards obtained from the underlying dynamics [44, 103, 114]. Among widely known reward-shaping methods are inverse optimal control (IOC) [37, 65] and inverse RL [5, 35, 152]. IOC reshapes the reward function using state and control trajectories assuming a stable control system. Unlike IOC, inverse RL reshapes the reward function by observing a demonstrated optimal behavior without the knowledge of system dynamics. However, the inverse RL for reward-shaping becomes infeasible as the system size grows. Finding a low-dimensional projection of the large-scale MAS dataset, which maintains its fundamental dynamic information within an iterative framework, is desirable. Maintaining

MAS stability while retrieving target performance measures is crucial. One way to generate a reward function that guarantees stability is by using the IOC based on the system dynamics and demonstrations. Inverse RL in [86] incorporates the IOC to ensure stability in adversarial apprentice games.

In inverse RL, the system is excited with exploratory noise to collect an adequate number of data samples for policy and reward updates [152]. For the LQR problem, these updates are performed using the least squares method. The estimate is iteratively refined until convergence to an optimal feedback gain derived from solving the Algebraic Riccati Equation. A minimum of $n(n+1)/2$ data samples, where n represents the system's order, are needed for a unique solution [58]. This issue significantly affects reward-shaping and control in large-scale MASs. The studies in [30, 101] investigate RL algorithms for decomposed optimal control problems, while [59, 118] explore RL for MASs, focusing on dimension reduction using controllability and observability gramians. However, they do not specifically address reward-shaping.

We propose accelerated and data-efficient design of inverse RL controllers through dimensionality reduction by leveraging the balanced proper orthogonal decomposition (POD) [116]. The LQR controller can be learned by projecting the measured states into a lower-dimensional space and capturing the dominant modes identified through the POD [76]. This approach harnesses the lossless dimensionality reduction property, controlling the dominant behavior of the MAS states to achieve the desired mission. The inverse RL algorithm is then designed by using low-dimensional data, which significantly reduces the computational complexity involved in the learning process. The dynamic mode decomposition (DMD) in [111, 125] can reduce the data dimensions and analyzes spatial-temporal data without relying on the system model. It retains the dynamic information of the original MAS [25], and imitates target MAS trajectories with less computational burden. DMD provides balanced POD and identifies dynamic modes used to project data vectors in the

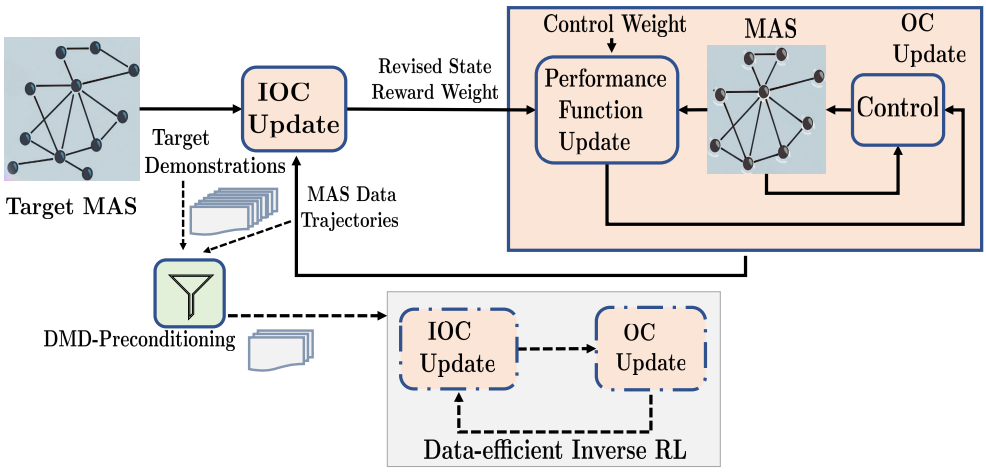


Figure 5.1. Data-efficient Inverse RL framework for solving the large-scale reward-shaping problem: 1) The learning MAS utilizes optimal control and IOC updates to infer the unknown performance function from optimal demonstrations of the target MAS, and 2) DMD is employed for lossless dimensionality reduction, projecting the necessary datasets for optimal control and IOC updates to lower dimensions. .

least squares of policy and reward updates onto a lower-dimensional space. We have recently proposed dimensionality reduction to accelerate off-policy RL for linear networked systems [34]. In this paper, we extend our approach to incorporate reward-shaping for large-scale MASs. Figure 1 shows the proposed framework of data-efficient inverse RL for large-scale MASs. This article takes the following steps and makes the following salient contributions:

1. We formulate a reward-shaping problem with dynamically decoupled linear agents. Therein, the MAS with LQR control tries to imitate the target MAS's behavior.
2. We leverage DMD for lossless dimensionality reduction, extract dynamic modes from state measurements, and build a projection matrix to preserve essential dynamics.

3. We develop a scalable model-free inverse RL algorithm to shape an unknown reward function by solving optimal control and IOC as sub-tasks.
4. The proposed algorithm is analyzed for its stability and convergence. We also quantify the non-uniqueness of state reward weights.

5.2 Preliminaries

5.2.1 Notations and Graph Theory

Notations: $\mathbb{R}^n$ denotes the n-dimensional Euclidean space. The n-by-n identity matrix is denoted by I_n. The Frobenius norm of a vector or a matrix is represented by $\|\cdot\|_F$, while the Euclidean norm is represented by $\|\cdot\|$. The complex conjugate transpose of D is denoted by D^t, and the pseudoinverse of D is represented by $D^\dagger$. $\otimes$ denotes the Kronecker product. For $y = [y_1,\ y_2,\ \ldots,\ y_n]^\top \in \mathbb{R}^n$, $y \otimes y \triangleq [y_1^2, \ldots, y_1y_n, y_2y_1, \ldots, y_2y_n, \ldots, y_n^2]^\top \in \mathbb{R}^{n^2}$ and $y \bar{\otimes} y \triangleq [y_1^2,\ 2y_1y_2,\ \ldots,\ 2y_1y_n,\ y_2^2,\ 2y_2y_3,\ \ldots,\ 2y_{n-1}y_n,\ y_n^2]^\top \in \mathbb{R}^{n(n+1)/2}$. For a symmetric matrix $W \in \mathbb{R}^{n\times n}$, $\text{vem}(W) \triangleq [W_{11},\ W_{12},\ \ldots,\ W_{1n},\ W_{22},\ \ldots,\ W_{2n},\ \ldots,\ W_{nn}]^\top$.

Graph Theory: The graph topology $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ for MAS with $\mathscr{N} = \{1, \ldots, N\}$ vertices, where $v_i \in \mathscr{V}$. The connectivity weights e_{ij} for the edges $\mathscr{E} \subseteq \mathscr{V} \times \mathscr{V}$ connect the vertices, and if $(v_j, v_i) \in \mathscr{E}$ then $e_{ij} = 1$; Otherwise, it is set to 0. The set of vertices that are connected to vertex v_i is denoted as $\mathscr{N}_i \cong \{v_j : e_{ij} > 0\}$, and there is no self-loops $(e_{ii} = 0)$. A directed path in the graph from vertex v_{i_1} to vertex v_{i_j} is defined as the series of edges $\mathscr{E}$ that pass through vertices $(v_{i_{z-1}}, v_{i_z}) \in \mathscr{E}$ for $z \in (2, \ldots, j)$. The in-degree and out-degree of vertex i are $d_i^i = \sum_{j=1}^N e_{ij}$ and $d_i^o = \sum_{j=1}^N e_{ji}$, respectively. The graph topology is defined as balanced, bi-directional, and undirected when $e_{ij} = e_{ji}$ and $d_i^i = d_i^o$ for all (v_i, v_j). Graph Laplacian matrix is $\mathscr{L} = \mathscr{D} - \mathscr{A} \in \mathbb{R}^{N\times N}$, where $\mathscr{A} = [e_{ij}], \mathscr{A} = \mathscr{A}^{\mathrm{T}}$ is the adjacency matrix, and $\mathscr{D} = \text{diag}\{d_i\}$ is the degree matrix.

5.2.2 Large-scale Reward-shaping Problem

This section formulates a reward-shaping problem for large-scale MAS in the context of discrete time. Consider the MAS with dynamically decoupled linear agents $i \in \mathcal{N}$,

$$x_{i_{k+1}} = Ax_{i_k} + Bu_{i_k}, \tag{5.1}$$

where $x_i \in \mathbb{R}^n$ denotes the state and $u_i \in \mathbb{R}^p$ denotes the control input of agent i at time instant k. Each agent has identical system matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times p}$. Assume (A,B) to be stabilizable. The aggregated dynamics of (5.1) is

$$\bar{x}_{k+1} = \bar{A}\bar{x}_k + \bar{B}\bar{u}_k, \tag{5.2}$$

where $\bar{x} = (x_1^\top, \ldots, x_N^\top)^\top \in \mathbb{R}^{nN}$, $\bar{u} = (u_1^\top, \ldots, u_N^\top)^\top \in \mathbb{R}^{pN}$, $\bar{A} = I_N \otimes A \in \mathbb{R}^{nN\times nN}$, and $\bar{B} = I_N \otimes B \in \mathbb{R}^{nN\times pN}$.

Consider target MAS that has dynamically decoupled linear agents $i \in \mathcal{N}$,

$$x_{ie_{k+1}} = Ax_{ie_k} + Bu_{ie_k}, \tag{5.3}$$

where $x_{i_e} \in \mathbb{R}^n$ and $u_{i_e} \in \mathbb{R}^p$ denote the state and the control input of the target agent i, respectively. The system matrices (A,B) are the same as that in (5.1). Define aggregated vectors $\bar{x}_e = (x_{1_e}^\top, \ldots, x_N^\top)^\top \in \mathbb{R}^{nN}$ and $\bar{u}_e = (u_{1_e}^\top, \ldots, u_N^\top)^\top \in \mathbb{R}^{pN}$. The aggregated target MAS of (5.3) is

$$\bar{x}_{e_{k+1}} = \bar{A}\bar{x}_{e_k} + \bar{B}\bar{u}_{e_k}, \tag{5.4}$$

where $\bar{A} = I_N \otimes A \in \mathbb{R}^{nN\times nN}$ and $\bar{B} = I_N \otimes B \in \mathbb{R}^{nN\times pN}$.

Remark 29. The target (5.4) and controlled MAS (5.2) are distinct systems. Respective agents are not adjacent on the same graph.

Consider each target agent i has $R_e = R_e^\top > 0 \in \mathbb{R}^{p\times p}$, $Q_{e_1} = Q_{e_1}^\top \geq 0, \forall i$, and $Q_{e_2} = Q_{e_2}^\top \geq 0, \forall i \neq j$, where $Q_{e_1}, Q_{e_2} \in \mathbb{R}^{n\times n}$ are identical local and relative state reward weights

for each target agent, respectively. The aggregated form of the target performance function that combines the dynamic characteristic of target agents [21, 34] is

$$J_e\big(\bar{u}_{e_k},\bar{x}_{e_k}\big) = \sum_{k=0}^{\infty}\big[\bar{x}_{e_k}^{\top}\bar{Q}_e\bar{x}_{e_k} + \bar{u}_{e_k}^{\top}\bar{R}_e\bar{u}_{e_k}\big], \tag{5.5}$$

where the matrix $\bar{Q}_e \geq 0$ is strictly diagonally-dominant, and $\bar{R}_e > 0$ is a block-diagonal matrix given as

$$\bar{R}_e = I_N \otimes R_e \in \mathbb{R}^{pN\times pN}, \tag{5.6a}$$

$$\bar{Q}_e = (I_N \otimes Q_{e_1} + \mathscr{L} \otimes Q_{e_2}) \in \mathbb{R}^{nN\times nN}. \tag{5.6b}$$

The optimal $\bar{u}_{e_k}^*$ is found by minimizing the aggregated $J_e(\bar{u}_{e_k},\bar{x}_{e_k})$ in (5.5) as

$$\begin{aligned} J_e^*(\bar{x}_{e_k}) &= \min_{\bar{u}_{e_k}} J_e(\bar{u}_{e_k},\bar{x}_{e_k}) \\ &= \bar{x}_{e_k}^{\top}P_e\bar{x}_{e_k}, \quad P_e = P_e^{\top} \in \mathbb{R}^{nN\times nN} > 0. \end{aligned} \tag{5.7}$$

Assumption 8. The matrices $\bar{Q}_e$ and $\bar{R}_e$ are unknown to the MAS in (5.2), but can observe target trajectories of $(\bar{x}_{e_k},\bar{u}_{e_k}^*)$.

Based on the optimal control theory [79], the optimal control policy $\bar{u}_{e_k}^*$, feedback control gain K_e, and target cost matrix P_e, are given by

$$\bar{u}_{e_k}^* = -K_e^*\bar{x}_{e_k}, \quad K_e \in \mathbb{R}^{pN\times nN}, \tag{5.8a}$$

$$K_e^* = (\bar{R}_e + \bar{B}^{\top}P_e\bar{B})^{-1}\bar{B}^{\top}P_e\bar{A}, \tag{5.8b}$$

$$P_e = \bar{A}^{\top}P_e\bar{A} - \bar{A}^{\top}P_e\bar{B}(\bar{R}_e + \bar{B}^{\top}P_e\bar{B})^{-1}\bar{B}^{\top}P_e\bar{A} + \bar{Q}_e. \tag{5.8c}$$

Define a performance function that couples the dynamic behavior for agents the same as the target MAS. Each agent i has where $R = R^{\top} > 0 \in \mathbb{R}^{p\times p}$, $Q_1 = Q_1^{\top} \geq 0, \forall i$, and $Q_2 = Q_2^{\top} \geq 0, \forall i \neq j$. The aggregated form becomes

$$J\big(\bar{u}_k,\bar{x}_k\big) = \sum_{k=0}^{\infty}\big[\bar{x}_k^{\top}\bar{Q}\bar{x}_k + \bar{u}_k^{\top}\bar{R}\bar{u}_k\big], \tag{5.9}$$

where matrices $\bar{Q} \geq 0$ and $\bar{R} > 0$ are given as (by referring to the generation of $\bar{Q}_e$ and $\bar{R}_e$ in (5.6)),

$$\bar{R} = I_N \otimes R \in \mathbb{R}^{pN \times pN}, \tag{5.10a}$$

$$\bar{Q} = (I_N \otimes Q_1 + \mathscr{L} \otimes Q_2) \in \mathbb{R}^{nN \times nN}. \tag{5.10b}$$

The matrices $Q_1, Q_2 \in \mathbb{R}^{n \times n}$ penalize the local and relative state differences between agents, respectively.

Remark 30. MAS in (5.2) knows its own aggregated state reward weight $\bar{Q}$ and control weight $\bar{R}$. It is worth noting that $\bar{R}$ can differ from $\bar{R}_e$.

Definition 4. (**Equivalent state reward weight**) Given aggregated system matrices $\bar{A}$ and $\bar{B}$, the weights $\bar{Q}_e$ and $\bar{R}_e$ in (5.5), and $\bar{u}^*_{e_k}$ in (5.8a), one can choose $\bar{R} > 0$ arbitrarily and find $\bar{Q}$ such that there is P^∞ in the MAS that solve $\bar{u}_k = \bar{u}^*_{e_k}$. Then, $\bar{Q}$ is called an *equivalent weight* to $\bar{Q}_e$.

Large-scale Reward-shaping Problem: Given trajectories of the target MAS $(\bar{x}_{e_k}, \bar{u}^*_{e_k})$, MAS in (5.2) aims to learn a state reward weight $\bar{Q}$ that is equivalent to $\bar{Q}_e$ and satisfies (5.8c), replicating the target's behavior, i.e., $(\bar{x}_k, \bar{u}^*_k) = (\bar{x}_{e_k}, \bar{u}^*_{e_k})$.

5.3 Efficient Inverse Reinforcement Learning

This section begins with model-free inverse RL for large-scale reward shaping with optimal control and IOC learning as sub-tasks. We then develop a computationally tractable formulation using DMD for the inverse RL.

5.3.1 Model-free Inverse RL

Find a unique optimal control $\bar{u}_k^*$ for (5.2) which minimizes the aggregated $J(\bar{u}_k, \bar{x}_k)$ in (5.9) as

$$\begin{aligned} J^*(\bar{x}_k) &= \min_{\bar{u}_k} J(\bar{u}_k, \bar{x}_k) \\ &= \bar{x}_k^\top P \bar{x}_k, \quad P = P^\top \in \mathbb{R}^{nN \times nN} > 0. \end{aligned} \tag{5.11}$$

If the cost matrix P and feedback gain K satisfy

$$K^* = (\bar{R} + \bar{B}^\top P \bar{B})^{-1} \bar{B}^\top P \bar{A}, \tag{5.12a}$$

$$P = \bar{A}^\top P \bar{A} - \bar{A}^\top P \bar{B} (\bar{R} + \bar{B}^\top P \bar{B})^{-1} \bar{B}^\top P \bar{A} + \bar{Q}, \tag{5.12b}$$

then, the optimal control policy $\bar{u}_k^*$ can be represented as

$$\bar{u}_k^* = -K^* \bar{x}_k, \quad K \in \mathbb{R}^{pN \times nN}. \tag{5.13}$$

Given an estimate $\bar{Q}$ of $\bar{Q}_e$, MAS in (5.2) solves (5.12a) and (5.12b) using RL to derive an optimal solution (P, K^*). The reward $\bar{Q}$ is revised using IOC given $\bar{x}_{e_k}, \bar{u}_{e_k}^*$. The MAS seeks $\bar{Q}$ satisfying (5.12b) to imitate the target's behavior, i.e., $(\bar{x}_k, \bar{u}_k^*) = (\bar{x}_{e_k}, \bar{u}_{e_k}^*)$.

Lemma 6. If P satisfies the (5.12b) and

$$P = (\bar{A} - \bar{B}K_e)^\top P (\bar{A} - \bar{B}K_e) + K_e^\top \bar{R} K_e + \bar{Q}. \tag{5.14}$$

Then, given $(\bar{R} + \bar{B}^\top P \bar{B}) > 0$, $K = K_e$. Thus, $\bar{u}_k^*$ in (5.13) corroesponding to P is equal to $\bar{u}_{e_k}^*$ and MAS imitates the target's behavior i.e., $(\bar{x}_k, \bar{u}_k^*) = (\bar{x}_{e_k}, \bar{u}_{e_k}^*)$.

Proof. From (5.12), we can write

$$P = (\bar{A} - \bar{B}K)^\top P (\bar{A} - \bar{B}K) + K^\top \bar{R} K + \bar{Q}. \tag{5.15}$$

Subtract (5.14) from (5.15),

$$\begin{aligned} &K^\top \bar{R} K - K_e^\top \bar{R} K_e + (K_e^\top - K^\top) \bar{B}^\top P \bar{A} + \bar{A}^\top P \bar{B} \\ &\times (K_e - K) + K^\top \bar{B}^\top P \bar{B} K - K_e^\top \bar{B}^\top P \bar{B} K_e = 0. \end{aligned} \tag{5.16}$$

From (5.12a), we have $\bar{B}^\top P\bar{A} = (\bar{R}+\bar{B}^\top P\bar{B})K$. Then, rewrite (5.16) as

$$\begin{aligned}(K_e^\top - K^\top)(\bar{R}+\bar{B}^\top P\bar{B})K + K^\top(\bar{R}+\bar{B}^\top P\bar{B})K - K_e^\top \\ \times(\bar{R}+\bar{B}^\top P\bar{B})K_e + K^\top(\bar{R}+\bar{B}^\top P\bar{B})(K_e-K) = 0,\end{aligned} \tag{5.17}$$

which infers

$$(K_e-K)^\top(\bar{R}+\bar{B}^\top P\bar{B})(K-K_e) = 0. \tag{5.18}$$

Given $(\bar{R}+\bar{B}^\top P\bar{B}) > 0$, (5.18) implies $K = K_e$ and MAS imitates the target's behavior, i.e., $(\bar{x}_k, \bar{u}_k^*) = (\bar{x}_{e_k}, \bar{u}_{e_k}^*)$. □

In the inverse RL algorithm [152], the updates of $(P^{h+1}, \bar{Q}^{h+1}, K^{h+1})$ can be summarized as follows:

Step 1: Update P^{h+1} to align K in (5.12) with K_e in (5.8b). Given the target MAS (5.4) and $\bar{u}_{e_k}^*$ in (5.8a), we define the $\mathscr{Q}$-function as

$$\begin{aligned}\mathscr{Q} &= \begin{bmatrix}\bar{x}_{e_k}\\ \bar{u}_{e_k}\end{bmatrix}^\top \begin{bmatrix}\bar{A}^\top P^{h+1}\bar{A}+\bar{Q}^h & \bar{A}^\top P^{h+1}\bar{B}\\ \bar{B}^\top P^{h+1}\bar{A} & \bar{B}^\top P^{h+1}\bar{B}+\bar{R}\end{bmatrix}\begin{bmatrix}\bar{x}_{e_k}\\ \bar{u}_{e_k}\end{bmatrix}\\ &= \begin{bmatrix}\bar{x}_{e_k}\\ \bar{u}_{e_k}\end{bmatrix}^\top \begin{bmatrix}L_{\bar{x}\bar{x}}^{h+1} & L_{\bar{x}\bar{u}}^{h+1}\\ L_{\bar{u}\bar{x}}^{h+1} & L_{\bar{u}\bar{u}}^{h+1}\end{bmatrix}\begin{bmatrix}\bar{x}_{e_k}\\ \bar{u}_{e_k}\end{bmatrix}.\end{aligned} \tag{5.19}$$

Then, $L^{h+1} = \begin{bmatrix}L_{\bar{x}\bar{x}}^{h+1} & L_{\bar{x}\bar{u}}^{h+1}\\ L_{\bar{u}\bar{x}}^{h+1} & L_{\bar{u}\bar{u}}^{h+1}\end{bmatrix}$ in (5.19) can be computed by $(\bar{x}_{e_k}, \bar{u}_{e_k})$, with no need for $(\bar{A}, \bar{B})$, given by,

$$\begin{aligned}\Big(\begin{bmatrix}\bar{x}_{e_{k+1}}^\top & \bar{u}_{e_{k+1}}^\top\end{bmatrix}\bar{\otimes}\begin{bmatrix}\bar{x}_{e_{k+1}}^\top & \bar{u}_{e_{k+1}}^\top\end{bmatrix} - \begin{bmatrix}\bar{x}_{e_k}^\top & \bar{u}_{e_k}^\top\end{bmatrix}\bar{\otimes} \\ \begin{bmatrix}\bar{x}_{e_k}^\top & \bar{u}_{e_k}^\top\end{bmatrix}\Big)\mathrm{vem}(L^{h+1}) = -[\bar{x}_{e_k}^\top\bar{Q}^h\bar{x}_{e_k} + \bar{u}_{e_k}^\top\bar{R}\bar{u}_{e_k}],\end{aligned} \tag{5.20}$$

Define the operators $(\rho_k, \phi_k, \rho, \Phi)$ as

$$\rho_k = \Big(\begin{bmatrix} \bar{x}_{e_{k+1}}^\top & \bar{u}_{e_{k+1}}^\top \end{bmatrix} \bar{\otimes} \begin{bmatrix} \bar{x}_{e_{k+1}}^\top & \bar{u}_{e_{k+1}}^\top \end{bmatrix} - \begin{bmatrix} \bar{x}_{e_k}^\top & \bar{u}_{e_k}^\top \end{bmatrix} \bar{\otimes} \begin{bmatrix} \bar{x}_{e_k}^\top & \bar{u}_{e_k}^\top \end{bmatrix} \Big)^\top \in \mathbb{R}^{1\times \frac{(nN+pN)(nN+pN+1)}{2}}, \tag{5.21a}$$

$$\phi_k = -[\bar{x}_{e_k}^\top \bar{Q}^h \bar{x}_{e_k} + \bar{u}_{e_k}^\top \bar{R} \bar{u}_{e_k}], \tag{5.21b}$$

$$\rho = [\rho_k, \rho_{k-1}, \ldots, \rho_{k-s+1}]^\top, \tag{5.21c}$$

$$\Phi = [\phi_k, \phi_{k-1}, \ldots, \phi_{k-s+1}]^\top, \tag{5.21d}$$

where s is data sample number. The operator ρ_k in (5.21a) can be simplified as

$$\begin{aligned} \rho_k &= \Bigg(\begin{bmatrix} \bar{x}_{e_{k+1}}^\top \bar{\otimes} \bar{x}_{e_{k+1}}^\top & \bar{x}_{e_{k+1}}^\top \bar{\otimes} \bar{u}_{e_{k+1}}^\top \\ \bar{u}_{e_{k+1}}^\top \bar{\otimes} \bar{x}_{e_{k+1}}^\top & \bar{u}_{e_{k+1}}^\top \bar{\otimes} \bar{u}_{e_{k+1}}^\top \end{bmatrix} \\ &\quad - \begin{bmatrix} \bar{x}_{e_k}^\top \bar{\otimes} \bar{x}_{e_k}^\top & \bar{x}_{e_k}^\top \bar{\otimes} \bar{u}_{e_k}^\top \\ \bar{u}_{e_k}^\top \bar{\otimes} \bar{x}_{e_k}^\top & \bar{u}_{e_k}^\top \bar{\otimes} \bar{u}_{e_k}^\top \end{bmatrix} \Bigg) \\ &= \begin{bmatrix} \rho_{xx} \in \mathbb{R}^{1\times \frac{(nN)(nN+1)}{2}} & \rho_{xu} \in \mathbb{R}^{1\times npN^2} \\ \rho_{ux} \in \mathbb{R}^{1\times npN^2} & \rho_{uu} \in \mathbb{R}^{1\times \frac{(pN)(pN+1)}{2}} \end{bmatrix}. \end{aligned} \tag{5.22}$$

Arranging the submatrices of this symmetric matrix into the vectors to find its unknowns results in

$$\rho_k = \begin{bmatrix} \rho_{xx} & \rho_{xu} & \rho_{uu} \end{bmatrix} \in \mathbb{R}^{1\times \frac{(nN+pN)(nN+pN+1)}{2}}. \tag{5.23}$$

Compute L^{h+1} by the batch least squares (BLS) method

$$\mathrm{vem}(L^{h+1}) = (\rho^\top \rho)^{-1} \rho^\top \Phi. \tag{5.24}$$

Step 2: Update $\bar{Q}^{h+1}$ of (5.2) in (5.12b) by a model-free solution for IOC with a learning rate $\delta \in (0,1]$. Given the system in (5.2), $\bar{u}_k$ in (5.13), and $M^{h+1} = L^{h+1} - \begin{bmatrix} \bar{Q}^h & 0 \\ 0 & \bar{R} \end{bmatrix}$, we compute $\bar{Q}^{h+1}$ using $(\bar{x}_k, \bar{u}_k)$,

$$\begin{aligned}(\bar{x}_k^\top \bar{\otimes} \bar{x}_k^\top)\mathrm{vem}(\bar{Q}^{h+1}) = (1-\delta)\bar{x}_k^\top \bar{Q}^h \bar{x}_k - \delta \bar{u}_k^\top \bar{R} \bar{u}_k \\ + \delta \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix} \\ - \delta \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}. \end{aligned} \tag{5.25}$$

Define the operators $(\theta_k, \omega_k, \Theta, \Omega)$ as

$$\theta_k = [\bar{x}_k^\top \bar{\otimes} \bar{x}_k^\top]^\top \in \mathbb{R}^{1\times((nN+1)nN/2)}, \tag{5.26a}$$

$$\begin{aligned}\omega_k = \delta \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix} - \delta \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \\ + (1-\delta)\bar{x}_k^\top \bar{Q}^h \bar{x}_k - \delta \bar{u}_k^\top \bar{R} \bar{u}_k, \end{aligned} \tag{5.26b}$$

$$\Theta = [\theta_k, \theta_{k-1}, \ldots, \theta_{k-s+1}]^\top, \tag{5.26c}$$

$$\Omega = [\omega_k, \omega_{k-1}, \ldots, \omega_{k-s+1}]^\top, \tag{5.26d}$$

and $\bar{Q}^{h+1}$ computed by BLS, given as

$$\mathrm{vem}(\bar{Q}^{h+1}) = (\Theta^\top \Theta)^{-1} \Theta^\top \Omega. \tag{5.27}$$

Step 3: Update $\bar{u}_k$ using the updated P^{h+1}

$$K^{h+1} = (L_{\bar{u}\bar{u}}^{h+1})^{-1} L_{\bar{u}\bar{x}}^{h+1}, \tag{5.28}$$

$$\bar{u}_k = -K^{h+1}\bar{x}_k = -\big((L_{\bar{u}\bar{u}}^{h+1})^{-1} L_{\bar{u}\bar{x}}^{h+1}\big)\bar{x}_k. \tag{5.29}$$

Note that (5.24) has $d_p = ((nN + pN)(nN + pN + 1)/2)$ unknown parameters. At each iteration, the BLS solution to (5.24) requires the full rank of $(\rho^\top \rho)$ and, at least, $s \geq d_p$ data samples. Similarly, (5.27) has $d_q = (nN(nN+1)/2)$ unknown parameters. BLS solution to (5.27) requires the full rank of $(\Theta^\top \Theta)$ and, at least, $s \geq d_q$ data samples at every iteration.

Remark 31. BLS needs $d_p = ((nN + pN)(nN + pN + 1)/2)$ data samples in (5.24) and $d_q = (nN(nN+1)/2)$ data samples in (5.27) to find a unique solution. Given $N >> 1$, it is computationally intractable to find the optimal control and reward functions of large-scale systems using inverse RL.

Next, we develop a scalable framework to address the computational issues in Remark 31. Our algorithm minimizes the performance function J of large-scale MAS in (5.9) while reducing the time required to imitate the target's trajectories.

5.3.2 Lossless Dimensionality Reduction

This section proposes a dimensionality reduction that allows for training the controller using a lower-dimensional state without loss of information. We construct lower dimensional state vectors η_k and η_{e_k} using collected state measurements $\bar{x}_k \in \mathbb{R}^{nN}$ and $\bar{x}_{e_k} \in \mathbb{R}^{nN}$. We consider projection matrices $D \in \mathbb{R}^{r \times nN}$ and $D_e \in \mathbb{R}^{r \times nN}$ given by

$$\eta_k \approx D\bar{x}_k, \tag{5.30}$$

$$\eta_{e_k} \approx D_e \bar{x}_{e_k}. \tag{5.31}$$

For the model-free formulation, matrix D will be solely solved using $\bar{x}_k$. The matrix D captures the redundant terms that are not required for the controllability of (5.2). One can use a balanced truncation to build D for a large-scale MAS, which could otherwise become impractical as it would require solving high-dimensional Lyapunov equations [135, 136].

An alternative method is balanced POD [116] which approximates the balanced truncation [97].

Considering (5.30), the lower dimensional state η_k can be used to retrieve $\bar{x}_k$ without any loss of information. Then, one can train the controller (5.13) using η_k, resulting in an optimal controller that minimizes J in (5.9) and replicates the target's behavior, i.e., $(\bar{x}_k, \bar{u}_k^*) = (\bar{x}_{e_k}, \bar{u}_{e_k}^*)$. We provide the following Lemma based on [136] to show that $\bar{u}_k^*$ in (5.13) is obtained by learning from the reduced state η_k rather than $\bar{x}_k$, and the performance function of the feedback control system is close to the optimal J^* in (5.11).

Lemma 7. Given MAS dynamics (5.2) and performance function (5.9), η_k in (5.30) satisfies

$$\eta_{k+1} = D\bar{A}D^{\dagger}\eta_k + D\bar{B}\bar{u}_k, \tag{5.32a}$$

$$\bar{J}(\bar{u}_k, \eta_k) = \sum_{k=0}^{\infty} \left[\eta_k^{\top}\bar{Q}_r\eta_k + \bar{u}_k^{\top}\bar{R}\bar{u}_k\right], \tag{5.32b}$$

where $D\bar{A}D^{\dagger} \in \mathbb{R}^{r\times r}$ is Hurwitz, $\bar{x}_k \approx D^{\dagger}\eta_k$ holds for any $\bar{u}_k, \bar{x}_k$, and $\bar{Q}_r = (D^{\dagger})^{\top}\bar{Q}D^{\dagger} \geq 0 \in \mathbb{R}^{r\times r}$.

Remark 32. Provided the Lemma 6 and (5.32), given target MAS dynamics (5.4) and target performance (5.5), η_{e_k} in (5.31) satisfies

$$\eta_{e_k+1} = D_e\bar{A}D_e^{\dagger}\eta_{e_k} + D_e\bar{B}\bar{u}_{e_k}, \tag{5.33a}$$

$$\bar{J}_e(\bar{u}_{e_k}, \eta_{e_k}) = \sum_{k=0}^{\infty} \left[\eta_{e_k}^{\top}\bar{Q}_{e_r}\eta_{e_k} + \bar{u}_{e_k}^{\top}\bar{R}_e\bar{u}_{e_k}\right]. \tag{5.33b}$$

Remark 33. Note that only spatial patterns can be identified by the balanced POD and singular value decomposition (SVD) in $\bar{x}_k$, whereas DMD can identify spatiotemporal patterns [76, 116, 147]. First, DMD is utilized for balanced POD to yield D and D_e. Second, DMD makes it possible to derive dynamic modes from the collected $(\bar{x}_k, \bar{x}_{e_k})$ data. The inverse RL data vectors of the large-scale MAS (5.2) are projected onto the r-dimensional

space using dynamic modes. This process preserves the nN-dimensional dynamic information of (5.2).

5.3.3 Extraction of Dynamic Modes and Building D

We next extract dynamic modes and build D satisfying lossless dimensionality reduction using the collected data trajectories $(\bar{x}_{e_k}, \bar{u}_{e_k}, \bar{x}_k, \bar{u}_k)$ instead of information of $(\bar{A}, \bar{B})$.

To extract dynamic modes using DMD, we develop data-driven models of (5.2) and target system using the collected state measurements $(\bar{x}_{e_k}, \bar{u}_{e_k}, \bar{x}_k, \bar{u}_k)$. Let the matrix pair (ψ_1, ψ_2) be formed such that ψ_2 is the matrix obtained by shifting ψ_1 over time,

$$\psi_1 = \begin{bmatrix} \bar{x}_1(1) & \dots & \bar{x}_1(l-1) \\ \vdots & \ddots & \vdots \\ \bar{x}_{nN}(1) & \dots & \bar{x}_{nN}(l-1) \end{bmatrix} \in \mathbb{R}^{nN\times(l-1)},$$

$$\psi_2 = \begin{bmatrix} \bar{x}_1(2) & \dots & \bar{x}_1(l) \\ \vdots & \ddots & \vdots \\ \bar{x}_{nN}(2) & \dots & \bar{x}_{nN}(l) \end{bmatrix} \in \mathbb{R}^{nN\times(l-1)}, \tag{5.34}$$

where l is the number of data points. The data-driven dynamic model is given by

$$\psi_2 = q\psi_1. \tag{5.35}$$

The approximate solution for $q \in \mathbb{R}^{nN\times nN}$ can be obtained by minimizing $\|\psi_2 - q\psi_1\|_F$. This norm is minimized by

$$q = \psi_2 \psi_1^{\dagger}, \tag{5.36}$$

where $\psi_1^{\dagger}$ is computed by SVD. Here, $\psi_1 = USV^{\iota}$, $U \in \mathbb{R}^{nN\times nN}$, $S \in \mathbb{R}^{nN\times nN}$, and $V \in \mathbb{R}^{(l-1)\times nN}$. The values in S correspond to the singular values σ_j of ψ_1, with j ranging from 1 to nN. The energy content in each σ_j is given by $E_{\sigma_j} = \frac{\sigma_j}{\sum_{j=1}^{nN}\sigma_j}$ [42, 125]. We exclude extreme and less significant singular values from S. The resulting square matrix

is used to compute $\psi_1^{\dagger}$. A high percentage of energy content can be retained by setting a threshold rank r and truncating S accordingly while reducing the dimensionality of ψ_1. After reduction, ψ_1 can be approximated as $\tilde{U}\tilde{S}\tilde{V}^{\iota}$, where $\tilde{U}$, $\tilde{S}$, and $\tilde{V}$ are matrices of dimensions $\mathbb{R}^{nN\times r}$, $\mathbb{R}^{r\times r}$, and $\mathbb{R}^{(l-1)\times r}$, respectively. The projection matrix D in (5.30) is computed from the transpose of $\tilde{U}$ as $D^{\dagger}=\tilde{U}$ [67, 136], gives the projection from $\mathbb{R}^{nN}$ to $\mathbb{R}^{r}$. The approximation of q in (5.36) given by

$$q \approx \tilde{q} = \psi_2 \tilde{V}\tilde{S}^{-1}\tilde{U}^{\iota}, \tag{5.37}$$

and dynamic model (5.35) results in

$$\psi_2 \approx \tilde{q}\psi_1, \tag{5.38}$$

where $\tilde{q} \in \mathbb{R}^{nN\times nN}$. The lower-dimensional model given by

$$\begin{aligned}\eta_{k+1} &= D\tilde{q}D^{\dagger}\eta_k \\ &= D\psi_2\tilde{V}\tilde{S}^{-1}\eta_k \triangleq \bar{q}\eta_k,\end{aligned} \tag{5.39}$$

where $\bar{q} \triangleq D\psi_2\tilde{V}\tilde{S}^{-1}$ and $\bar{q} \in \mathbb{R}^{r\times r}$. Provided Lemma 6, one can find $\bar{u}_k$ that minimizes J in (5.9) as

$$\begin{aligned}\bar{J}^{*}(\bar{x}_k) = \min_{\bar{u}_k}\bar{J}(\bar{u}_k,\bar{x}_k) &= \sum_{k=0}^{\infty}\left[\eta_k^{\top}\bar{Q}_r\eta_k + \bar{u}_k^{\top}R\bar{u}_k\right] \\ &= \eta_k^{\top}P_r\eta_k,\end{aligned} \tag{5.40}$$

where $P_r \in \mathbb{R}^{r\times r} > 0$ is a reduced cost matrix. Given $P > 0$ of original MAS, the cost matrix P_r of the reduced feedback control system spans identical eigenvalues as P, i.e., $S(P_r) = S(DPD^{\dagger})$. Learning the optimal control for the nN-dimensional MAS (5.2) having J in (5.9) is identical to learning for the resulting feedback control system (5.32a).

Equations (5.37) and (5.39) indicate that the eigenvalues of $\tilde{q}$ and $\bar{q}$ are similar, as long as $D=\tilde{U}^{\iota}$, according to [125]. This can be seen by observing that $\tilde{q}v = \psi_2\tilde{V}\tilde{S}^{-1}\tilde{U}^{\iota}v =$

$D\psi_2\tilde{V}\tilde{S}^{-1}v = \bar{q}v = \lambda v$, where v is an eigenvector and λ is the corresponding eigenvalue. Analyzing the eigenvalues of $\bar{q}$ is more practical than analyzing those of $q \approx \tilde{q}$ when r is much smaller than nN. Using the dynamic model presented in equations (5.39), the dynamic modes of $q \approx \tilde{q}$, denoted by Λ, and the eigenvectors of $\bar{q}$, denoted by v, are linearly related

$$\Lambda = \frac{1}{\lambda}\psi_2\tilde{V}\tilde{S}^{-1}v, \quad \Lambda \in \mathbb{R}^{nN\times r}. \tag{5.41}$$

The DMD modes of MAS in (5.2) are the exact eigenvectors of $q \approx \tilde{q} \in \mathbb{R}^{nN\times nN}$, and the columns of Λ represent them. [34, 125, 136] support this statement, and it can be confirmed by

$$\begin{aligned}\tilde{q}\Lambda = \mathbb{P}\tilde{q}\Lambda &= (\tilde{U}\tilde{U}^t)(\psi_2\tilde{V}\tilde{S}^{-1}\tilde{U}^t)(\frac{1}{\lambda}\psi_2\tilde{V}\tilde{S}^{-1}v)\\ &= \tilde{U}\tilde{U}^t\psi_2\tilde{V}\tilde{S}^{-1}v = \tilde{U}\bar{q}v = \lambda\Lambda.\end{aligned} \tag{5.42}$$

Remark 34. Note that we have provided the process of computing D and Λ. Similarly, we obtain D_e and Λ_e for the target system as shown below: Let (ψ_{e_1}, ψ_{e_2}) be formed such that ψ_{e_2} is the matrix obtained by shifting ψ_{e_1} over time,

$$\begin{aligned}\psi_{e_1} &= \begin{bmatrix} \bar{x}_{e_1}(1) & \dots & \bar{x}_{e_1}(l-1) \\ \vdots & \ddots & \vdots \\ \bar{x}_{e_{nN}}(1) & \dots & \bar{x}_{e_{nN}}(l-1) \end{bmatrix} \in \mathbb{R}^{nN\times(l-1)},\\ \psi_{e_2} &= \begin{bmatrix} \bar{x}_{e_1}(2) & \dots & \bar{x}_{e_1}(l) \\ \vdots & \ddots & \vdots \\ \bar{x}_{e_{nN}}(2) & \dots & \bar{x}_{e_{nN}}(l) \end{bmatrix} \in \mathbb{R}^{nN\times(l-1)},\end{aligned} \tag{5.43}$$

$$\psi_{e_2} = q_e\psi_{e_1}, \tag{5.44}$$

where $q_e = \bar{A} - \bar{B}K_e \in \mathbb{R}^{nN\times nN}$ and $\psi_{e_1}^{\dagger}$ is approximated by SVD

$$q_e = \psi_{e_2}\psi_{e_1}^{\dagger}. \tag{5.45}$$

Therein, $\psi_{e_1} = U_e S_e V_e^\iota$, $U_e \in \mathbb{R}^{nN\times nN}$, $S_e \in \mathbb{R}^{nN\times nN}$, and $V_e \in \mathbb{R}^{(l-1)\times nN}$. After excluding less significant singular values σ_{e_j} from S_e, ψ_{e_1} is approximated as $\tilde{U}_e\tilde{S}_e\tilde{V}_e^\iota$, where $\tilde{U}_e$, $\tilde{S}_e$, and $\tilde{V}_e$ are matrices of dimensions $\mathbb{R}^{nN\times r}$, $\mathbb{R}^{r\times r}$, and $\mathbb{R}^{(l-1)\times r}$, respectively. One can compute D_e in (5.31), given $D_e^\dagger = \tilde{U}_e$, and approximate q_e as

$$q_e \approx \tilde{q}_e = \psi_{e_2}\tilde{V}_e\tilde{S}_e^{-1}\tilde{U}_e^\iota, \tag{5.46}$$

$$\psi_{e_2} \approx \tilde{q}_e\psi_{e_1}, \quad \tilde{q}_e \in \mathbb{R}^{nN\times nN}. \tag{5.47}$$

The lower-dimensional target dynamic model is given by

$$\eta_{e_{k+1}} = D_e\tilde{q}_eD_e^\dagger\eta_{e_k} = D_e\psi_{e_2}\tilde{V}_e\tilde{S}_e^{-1}\eta_{e_k} \triangleq \bar{q}_e\eta_{e_k}, \tag{5.48}$$

where $\bar{q}_e \triangleq D_e\psi_{e_2}\tilde{V}_e\tilde{S}_e^{-1} \in \mathbb{R}^{r\times r}$. The target dynamic modes of $q_e \approx \tilde{q}_e$ (denoted by Λ_e) and the eigenvectors of $\bar{q}_e$ (denoted by v_e) are linearly related through

$$\Lambda_e = \frac{1}{\lambda_e}\psi_{e_2}\tilde{V}_e\tilde{S}_e^{-1}v_e, \quad \Lambda_e \in \mathbb{R}^{nN\times r}. \tag{5.49}$$

5.3.4 Data-efficient Model-free Inverse RL

We can now present our proposed inverse RL scheme by drawing from the premise of constructing the (D, D_e) for truncating $(\bar{x}_k, \bar{x}_{e_k})$ to (η_k, η_{e_k}) and extracting dynamic modes (Λ, Λ_e) from the collected data trajectories, as discussed in Section (5.3.2, 5.3.3). The proposed inverse RL scheme consists of four stages: *1. Data collection*, *2. Build* (D,Λ) *&* (D_e,Λ_e)*, 3. Preconditioning of datasets,* and *4. Updates of cost matrix, feedback gain,* and *state reward weight*. In the data collection stage, state measurements $\bar{x}_k$ and $\bar{x}_{e_k}$ are collected as a result of exciting the system (5.2) and (5.4) with exploration noise $\bar{u}_k$ and $\bar{u}_{e_k}$, respectively. Given these measurements, one can build (D,Λ) & (D_e,Λ_e) as shown in Section 5.3.3. In the preconditioning stage, we first define L_r^{h+1} by replacing $\bar{x}_{e_k}$ in (5.21) with η_{e_k} to update P_r and K_r. Second, we define Q_r^{h+1} by replacing $\bar{x}_k$ in (5.26) with η_k

to update $\bar{Q}_r$. The third and fourth stages update the cost matrix P_r, feedback gain K_r, and state reward weight $\bar{Q}_r$ for the lower-dimensional models. The updated datasets are

$$\rho_{rk} = \begin{bmatrix} \rho_{xx}(\Lambda_e \bar{\otimes} \Lambda_e) \in \mathbb{R}^{1\times(r(r+1)/2)} \\ \rho_{xu}(I_{pN} \otimes \Lambda_e) \in \mathbb{R}^{1\times rpN} \\ \rho_{uu} \in \mathbb{R}^{1\times(pN(pN+1)/2)} \end{bmatrix}^\top, \tag{5.50a}$$

$$\theta_{r_k} = \theta_k(\Lambda \bar{\otimes} \Lambda), \tag{5.50b}$$

where $\rho_{rk} \in \mathbb{R}^{1\times\frac{(r+pN)(r+pN+1)}{2}}$ and $\theta_{r_k} \in \mathbb{R}^{1\times\frac{r(r+1)}{2}}$. The following theorem shows that given (5.32) and (5.33), we can define a lower-dimensional Q-function and find lower dimensional feedback gain and reward weight by BLS.

Theorem 10. *Given the lower-dimensional dynamic model of MAS in* (5.32)*, and of target system in* (5.33)*, we can find the lower dimensional feedback gain* K_r^{h+1} *and reward weight* $\bar{Q}_r^h$

$$vem(L_r^{h+1}) = (\rho_r^\top \rho_r)^{-1} \rho_r^\top \Phi_r, \tag{5.51a}$$

$$K_r^{h+1} = (L_{\bar{u}\bar{u}}^{h+1})^{-1} L_{\bar{u}\eta}^{h+1}, \tag{5.51b}$$

$$vem(\bar{Q}_r^{h+1}) = (\Theta_r^\top \Theta_r)^{-1} \Theta_r^\top \Omega_r. \tag{5.51c}$$

Proof. Given the lower-dimensional dynamic model of MAS in (5.32) with η_k, as well as of the target system in (5.33) with η_{e_k}, we can define a lower dimensional Q-function as

$$\mathscr{Q}_r(\eta_{e_k}, \bar{u}_{e_k}, P_r^{h+1}) = \begin{bmatrix} \eta_{e_k} \\ \bar{u}_{e_k} \end{bmatrix}^\top L_r^{h+1} \begin{bmatrix} \eta_{e_k} \\ \bar{u}_{e_k} \end{bmatrix}, \tag{5.52}$$

where the matrix L_r^{h+1} for P_r^{h+1} is

$$L_r^{h+1} = \begin{bmatrix} L_{\eta\eta}^{h+1} & L_{\eta\bar{u}}^{h+1} \\ L_{\bar{u}\eta}^{h+1} & L_{\bar{u}\bar{u}}^{h+1} \end{bmatrix}, \tag{5.53}$$

with $L^{h+1}_{\eta\eta} = (D\bar{A}D^{\dagger})^{\top}P^{h+1}_{r}(D\bar{A}D^{\dagger}) + \bar{Q}^{h}_{r}$, $L^{h+1}_{\eta\bar{u}} = (D\bar{A}D^{\dagger})^{\top}P^{h+1}_{r}D\bar{B}$, $L^{h+1}_{\bar{u}\eta} = (L^{h+1}_{\eta\bar{u}})^{\top}$, and $L^{h+1}_{\bar{u}\bar{u}} = (D\bar{B})^{\top}P^{h+1}_{r}(D\bar{B}) + \bar{R}$. A reduced-dimensional K^{h+1}_{r} is

$$K^{h+1}_{r} = (L^{h+1}_{\bar{u}\bar{u}})^{-1}L^{h+1}_{\bar{u}\eta}. \tag{5.54}$$

One can compute L^{h+1}_{r} using $(\eta_{e_k}, \bar{u}_{e_k})$ in (5.53), such that there is no need for $(\bar{A}, \bar{B})$, given by

$$\begin{aligned}\Big(\begin{bmatrix} \eta^{\top}_{e_{k+1}} & \bar{u}^{\top}_{e_{k+1}} \end{bmatrix} \bar{\otimes} \begin{bmatrix} \eta^{\top}_{e_{k+1}} & \bar{u}^{\top}_{e_{k+1}} \end{bmatrix} - \begin{bmatrix} \eta^{\top}_{e_k} & \bar{u}^{\top}_{e_k} \end{bmatrix} \\ \bar{\otimes} \begin{bmatrix} \eta^{\top}_{e_k} & \bar{u}^{\top}_{e_k} \end{bmatrix} \Big) \mathrm{vem}(L^{h+1}_{r}) = -[\eta^{\top}_{e_k}\bar{Q}^{h}_{r}\eta_{e_k} + \bar{u}^{\top}_{e_k}\bar{R}\bar{u}_{e_k}].\end{aligned} \tag{5.55}$$

Define the operators $(\rho_{rk}, \phi_{rk}, \rho_{r}, \Phi_{r})$ as

$$\begin{aligned}\rho_{rk} = \Big(\begin{bmatrix} \eta^{\top}_{e_{k+1}} & \bar{u}^{\top}_{e_{k+1}} \end{bmatrix} \bar{\otimes} \begin{bmatrix} \eta^{\top}_{e_{k+1}} & \bar{u}^{\top}_{e_{k+1}} \end{bmatrix} - \begin{bmatrix} \eta^{\top}_{e_k} & \bar{u}^{\top}_{e_k} \end{bmatrix} \\ \bar{\otimes} \begin{bmatrix} \eta^{\top}_{e_k} & \bar{u}^{\top}_{e_k} \end{bmatrix} \Big)^{\top} \in \mathbb{R}^{1\times((r+pN)(r+pN+1)/2)},\end{aligned} \tag{5.56a}$$

$$\phi_{rk} = -(\eta^{\top}_{e_k}\bar{Q}^{h}_{r}\eta_{e_k} + \bar{u}^{\top}_{e_k}\bar{R}\bar{u}_{e_k}), \tag{5.56b}$$

$$\rho_{r} = [\rho_{rk}, \rho_{rk-1}, \ldots, \rho_{rk-s+1}]^{\top}, \tag{5.56c}$$

$$\Phi_{r} = [\phi_{rk}, \phi_{rk-1}, \ldots, \phi_{rk-s+1}]^{\top}, \tag{5.56d}$$

where s is the data sample and L^{h+1}_{r} is computed by BLS as

$$\mathrm{vem}(L^{h+1}_{r}) = (\rho^{\top}_{r}\rho_{r})^{-1}\rho^{\top}_{r}\Phi_{r}. \tag{5.57}$$

Note that (5.51a) has $d_{p_r} = ((r+pN)(r+pN+1)/2)$ unknown parameters. Solving (5.51a) requires a full rank of $(\rho_r^\top \rho_r)$ and needs $s \geq d_{p_r}$ data samples at each iteration. Similarly, compute $\bar{Q}_r^{h+1}$ using $(\eta_k, \bar{u}_k)$ by

$$\begin{aligned}(\eta_k^\top \bar{\otimes} \eta_k^\top)\mathrm{vem}(\bar{Q}_r^{h+1}) = (1-\delta)\eta_k^\top \bar{Q}_r^h \eta_k - \delta \bar{u}_k^\top \bar{R} \bar{u}_k \\ + \delta \begin{bmatrix} \eta_{k-1} \\ \bar{u}_{k-1} \end{bmatrix}^\top M_r^{h+1} \begin{bmatrix} \eta_{k-1} \\ \bar{u}_{k-1} \end{bmatrix} \\ - \delta \begin{bmatrix} \eta_k \\ \bar{u}_k \end{bmatrix}^\top M_r^{h+1} \begin{bmatrix} \eta_k \\ \bar{u}_k \end{bmatrix}. \end{aligned} \tag{5.58}$$

Define the operators $(\theta_{rk}, \omega_{rk}, \Theta_r, \Omega_r)$ as

$$\theta_{rk} = [\eta_k^\top \bar{\otimes} \eta_k^\top]^\top \in \mathbb{R}^{1\times(r(r+1)/2)}, \tag{5.59a}$$

$$\begin{aligned}\omega_{rk} = \delta \begin{bmatrix} \eta_{k-1} \\ \bar{u}_{k-1} \end{bmatrix}^\top M_r^{h+1} \begin{bmatrix} \eta_{k-1} \\ \bar{u}_{k-1} \end{bmatrix} - \delta \begin{bmatrix} \eta_k \\ \bar{u}_k \end{bmatrix}^\top M_r^{h+1} \begin{bmatrix} \eta_k \\ \bar{u}_k \end{bmatrix} \\ + (1-\delta)\eta_k^\top \bar{Q}_r^h \bar{x}_k - \delta \bar{u}_k^\top \bar{R} \bar{u}_k, \end{aligned} \tag{5.59b}$$

$$\Theta_r = [\theta_{rk}, \theta_{rk-1}, \ldots, \theta_{rk-s+1}]^\top, \tag{5.59c}$$

$$\Omega_r = [\omega_{rk}, \omega_{rk-1}, \ldots, \omega_{rk-s+1}]^\top, \tag{5.59d}$$

where $M_r^{h+1} = L_r^{h+1} - \begin{bmatrix} \bar{Q}_r^h & 0 \\ 0 & \bar{R} \end{bmatrix}$ and $\bar{Q}_r^{h+1}$ are

$$\mathrm{vem}(\bar{Q}_r^{h+1}) = (\Theta_r^\top \Theta_r)^{-1} \Theta_r^\top \Omega_r. \tag{5.60}$$

Note that (5.51c) has $d_{q_r} = (r(r+1)/2)$ unknown parameters. Solving (5.51c) requires a full rank of $(\Theta_r^\top \Theta_r)$ and needs $s \geq d_{q_r}$ data samples at each iteration. □

Preconditioning in (5.50a) implies needing $d_{p_r} = ((r+pN)(r+pN+1)/2)$ unknown parameters in (5.51a) and $d_{q_r} = (r(r+1)/2)$ unknown parameters in (5.51c). The BLS

solution to (5.51a) requires, at least, $s \geq d_{p_r}$ data samples for a full rank of $(\rho_r^\top \rho_r)$ at each iteration, and the BLS solution to (5.51c) requires, at least, $s \geq d_{q_r}$ data samples for a full rank of $(\Theta_r^\top \Theta_r)$ at each iteration.

We now analyze Algorithm 8 to investigate properties such as convergence, stability, and the non-uniqueness of the learned reward weights compared to the target reward weights.

Theorem 11. ***Convergence analysis of Algorithm 8***. *Following Assumption* 1*, Lemma 6, and Theorem 10, Algorithm 8 solves the large-scale reward-shaping game problem and converges to* $(P_r^\infty, \bar{Q}_r^\infty, K_r^\infty)$ *which, in turn, converges to the solution* $(P^\infty, \bar{Q}^\infty, K^\infty)$.

Proof. Given (5.25) and (5.15), we have

$$\begin{aligned} &\bar{Q}^{h+1} - (1-\delta)\bar{Q}^h - \delta(P^{h+1} - \bar{A}^\top P^{h+1}\bar{A} \\ &+ \bar{A}^\top P^{h+1}\bar{B}(\bar{R} + \bar{B}^\top P^{h+1}\bar{B})^{-1}\bar{B}^\top P^{h+1}\bar{A}) = 0. \end{aligned} \tag{5.63}$$

Then, we can write

$$\bar{Q}^{h+1} = \bar{Q}^h + \delta(K_e - K^{h+1})^\top \bar{R}(K_e - K^{h+1}). \tag{5.64}$$

Let

$$\nabla^{h+1} = \delta(K_e - K^{h+1})^\top \bar{R}(K_e - K^{h+1}). \tag{5.65}$$

As $\bar{R} > 0$ and selecting a suitable scalar $\delta \in (0, 1]$, it follows that $\nabla^{h+1} \geq 0$. Then, (5.64) shows that $\bar{Q}^{h+1} \geq \bar{Q}^h$. It should be noted that the equality holds if and only if $K_e = K^{h+1}$. Likewise, if $\bar{Q}^0 \geq 0$, we have $\bar{Q}^1 \geq \bar{Q}^0$ and $\bar{Q}^{h+1} \geq \bar{Q}^h \geq \cdots \geq \bar{Q}^1 \geq 0$, implying $\bar{Q}^h$ is monotonically increasing over the iteration index h. Equation (5.64) becomes

$$\bar{Q}^{h+1} = \nabla^{h+1} + \nabla^h + \cdots + \nabla^1 + \bar{Q}^0. \tag{5.66}$$

Given (5.8), it is known that the target MAS feedback gain K_e^* is optimal with respect to the $\bar{Q}_e$ and $\bar{R}_e$. In practice, it is also known that K_e^* is optimal with infinitely many $\bar{Q} \geq 0$

Algorithm 8 Data-efficient Model-free Scalable Inverse Reinforcement Learning Algorithm

1 **Start**: Given l, set $h = 0$, and select a small threshold e. For MAS in (5.2), select initial $\bar{Q}_r^0 \geq 0$ and $\bar{R}_r > 0$.

2 **Data collection**: For $h = 0, 1, 2, \ldots, l-1$, collect target MAS data trajectories $(\bar{x}_{e_k}, \bar{u}_{e_k})$ and $(\bar{x}_k, \bar{u}_k)$.

3 **Build** (D, Λ) **&** (D_e, Λ_e):

- Take data matrices (ψ_1, ψ_2) and (ψ_{e_1}, ψ_{e_2}) as inputs.
- Augment: $\psi_2 = q\psi_1$ and $\psi_{e_2} = q\psi_{e_1}$.
- Compute SVD: $(U, S, V) \leftarrow \psi_1, (U_e, S_e, V_e) \leftarrow \psi_{e_1}$.
- Truncate: $(\tilde{U}, \tilde{S}, \tilde{V}) \leftarrow \psi_1, (\tilde{U}_e, \tilde{S}_e, \tilde{V}_e) \leftarrow \psi_{e_1}$.
- Output: Identify (D, Λ) and (D_e, Λ_e).

4 **Preconditioning of datasets**: Compute $(\rho_{rk}, \phi_{rk}, \rho_r, \Phi_r)$ and $(\theta_{rk}, \omega_{rk}, \Theta_r, \Omega_r)$ given in (5.50a), (5.56), and (5.59).

5 **Update cost matrix and feedback gain**: Compute L_r^{h+1} using (5.51a) and update K_r^{h+1} as in (5.51b).

6 **Update state reward weight** : Compute Q_r^{h+1} using (5.51c).

7 **Stop if** $\|K_r^{h+1} - K_r^h\| \leq e$. Otherwise, set $h \leftarrow h+1$ and go to Step 4.

8 **Retrieve**:

$$K^{h+1} = DK_r^{h+1}, \tag{5.61}$$

$$\bar{Q}^{h+1} = D^{\dagger}\bar{Q}_r^{h+1}D. \tag{5.62}$$

and $\bar{R} > 0$, which are not unique to respective $\bar{Q}_e$ and $\bar{R}_e$. As the h increases, it is possible for $\bar{Q}^h$ to increase and approach a neighboring value of at least one $\bar{Q}_e$. By varying the value of $\delta \in (0,1]$ to adjust the increase of $\bar{Q}^{h+1}$, it is possible to make the increase arbitrarily small, such that $\bar{Q}^{h+1}$ approximates $\bar{Q}_e$ more closely in terms of small threshold β over h. As a result, K^{h+1} approaches the K_e with the small threshold b selected, such that $\|K^{h+1} - K_e\| \leq b$. Then,

$$\|\bar{Q}^{h+1}\| = \|\nabla^h + \cdots + \nabla^1\| \geq \|\delta h b^2 \bar{R}\|. \tag{5.67}$$

Given (5.67), we have $\|\bar{Q}^{h+1}\| \geq \delta h b^2 \lambda_{min}(\bar{R})$, signifying the limited steps

$$h \leq ceil\left(\frac{\|\bar{Q}_e\|}{\delta b^2 \lambda_{min}(\bar{R})}\right) \equiv h_t. \tag{5.68}$$

This shows that the convergence. Then, updating (5.64), which is equivalent to (5.63), will provide an estimate of $\bar{Q}^{h+1}$ that is approximately equal to $\bar{Q}_e$. As $h \to \infty$, P^{h+1} and K^{h+1} are approximately equal to P^h and K^h, respectively. For the converged solutions, we have

$$\begin{aligned} &\bar{Q}^{\infty} - P^{\infty} + \bar{A}^{\top} P^{\infty} \bar{A} - \bar{A}^{\top} P^{\infty} \bar{B} \\ &\quad \times (\bar{R} + \bar{B}^{\top} P^{\infty} \bar{B})^{-1} \bar{B}^{\top} P^{\infty} \bar{A} = 0, \end{aligned} \tag{5.69a}$$

$$K^{\infty} = (\bar{R} + \bar{B}^{\top} P^{\infty} \bar{B})^{-1} \bar{B}^{\top} P^{\infty} \bar{A} = K_e^*. \tag{5.69b}$$

Given (5.20), we can write

$$\begin{aligned} &\begin{bmatrix} \bar{x}_{e_k} \\ \bar{u}_{e_k} \end{bmatrix}^{\top} \begin{bmatrix} \bar{A} & \bar{B} \\ -K_e\bar{A} & -K_e\bar{B} \end{bmatrix}^{\top} L^{h+1} \begin{bmatrix} \bar{A} & \bar{B} \\ -K_e\bar{A} & -K_e\bar{B} \end{bmatrix} \begin{bmatrix} \bar{x}_{e_k} \\ \bar{u}_{e_k} \end{bmatrix} \\ &\quad - \begin{bmatrix} \bar{x}_{e_k} \\ \bar{u}_{e_k} \end{bmatrix}^{\top} L^{h+1} \begin{bmatrix} \bar{x}_{e_k} \\ \bar{u}_{e_k} \end{bmatrix} = -[\bar{x}_{e_k}^{\top} \bar{Q}^h \bar{x}_{e_k} + \bar{u}_{e_k}^{\top} \bar{R} \bar{u}_{e_k}]. \end{aligned} \tag{5.70}$$

One can conclude that iterating on (L^{h+1}, M^{h+1}) is similar to iterating on $(P^{h+1}, \bar{Q}^{h+1})$

$$\begin{aligned} P^{h+1} &= \begin{bmatrix} I \\ -K_e^{h+1} \end{bmatrix}^\top L^{h+1} \begin{bmatrix} I \\ -K_e^{h+1} \end{bmatrix} \\ &= \bar{Q}^h + {K_e^{h+1}}^\top \bar{R} K_e^{h+1} + (\bar{A} - \bar{B}K_e^{h+1})^\top \\ &\quad \times P^{h+1}(\bar{A} - \bar{B}K_e^{h+1}), \end{aligned} \tag{5.71a}$$

$$\begin{aligned} \bar{x}_k^\top \bar{Q}^{h+1} \bar{x}_k &= (1-\delta)\bar{x}_k^\top \bar{Q}^h \bar{x}_k - \delta \bar{u}_k^\top \bar{R} \bar{u}_k \\ &\quad + \delta \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_{k-1} \\ \bar{u}_{k-1} \end{bmatrix} \\ &\quad - \delta \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}^\top M^{h+1} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}. \end{aligned} \tag{5.71b}$$

Since (5.71a) implies (5.14), K^{h+1} in (5.29) is equivalent to (5.12a) over iteration h. Similarly, equation (5.71b) with M^{h+1} is equivalent to (5.63). By employing BLS in (5.24) and (5.27), the unique solution for L^{h+1}, K^{h+1} and $\bar{Q}^{h+1}$ are obtained while the full rank condition is met.

Based on Lemma 6 and Remark 32, it can be concluded that (η_k, η_{e_k}) satisfies (5.32) and (5.33). Specifically, $\bar{x}_k \approx D^\dagger \eta_k$ holds $\forall \bar{u}_k, \bar{x}_k$ and $\bar{x}_{e_k} \approx D_e^\dagger \eta_{e_k}$ holds $\forall \bar{u}_{e_k}, \bar{x}_{e_k}$, with the Hurwitz $D\bar{A}D^\dagger$ and $D_e\bar{A}D_e^\dagger$, respectively. Then, the control policy $\bar{u}_k = -K_r^{h+1}\eta_k$ stabilizes system (5.32a) at every iteration h, where $K^{h+1} = D{K_r}^{h+1}$. Therefore, $(D\bar{A}D^\dagger - D\bar{B}K_r)$ is Hurwitz. With Remark 34, if (D, D_e) and (Λ, Λ_e) are found, then the scalable model-free RL can be achieved, where $K_r = D^\dagger K \in \mathbb{R}^{pN \times r}$ and $P_r = DPD^\dagger > 0 \in \mathbb{R}^{r \times r}$ satisfy (5.54) and (5.53), respectively. Considering K_r be the initial stabilizing gain, (P_r^{h+1}, K_r^{h+1}) is uniquely obtained by BLS in (5.51a). Q_r^{h+1} is uniquely obtained by BLS in (5.51c) while satisfying the full-rank condition. As shown in Lemma 6 that $K^{h+1} = DK_r^{h+1}$ and $\bar{Q}^{h+1} = D^\dagger \bar{Q}_r^{h+1} D$, Algorithm 8 converges to $(P^\infty, \bar{Q}^\infty, K^\infty)$ as $h \to \infty$. □

Theorem 12. ***Non-uniqueness analysis***. *Follow Theorem 11. If there exist* $\bar{Q}_\mu = D^\dagger \bar{Q}_{\mu_r}^{h+1} D \in \mathbb{R}^{nN\times nN}$, $P_\mu = D^\dagger P_{\mu_r}^{h+1} D \in \mathbb{R}^{nN\times nN}$, *and* $\bar{R}_\mu = \bar{R} - \bar{R}_e$ *that satisfy*

$$0 = \bar{B}^\top P_\mu \bar{A} - (\bar{R}_\mu + \bar{B}^\top P_\mu \bar{B}) K_e, \tag{5.72}$$

$$0 = \bar{Q}_\mu - P_\mu + \bar{A}^\top P_\mu \bar{A} - \bar{A}^\top P_\mu \bar{B} K_e, \tag{5.73}$$

then, we can show that any $\bar{Q}^\infty = \bar{Q}_e + \bar{Q}_\mu$ *and any* $P^\infty = P_e + P_\mu$ *assure* (5.69)*, where* $(P_e, \bar{Q}_e, K_e)$ *satisfy* (5.8)*. Any converged solution from Algorithm 1 lies within the set of all these solutions, implying* $\bar{Q} = D^\dagger \bar{Q}_r D$ *is non-unique.*

Proof. Given P_e in (5.8c), K_e in (5.8b), and by substitution $\bar{Q}_e = D^\dagger \bar{Q}_r^\infty D - \bar{Q}_\mu$ and $P_e = D^\dagger P_r^\infty D - P_\mu$, we can write

$$\begin{aligned}
0 = {} & \bar{A}^\top (P^\infty - P_\mu)\bar{A} - \bar{A}^\top (P^\infty - P_\mu)\bar{B} \\
& \times (\bar{R}_e + \bar{B}^\top (P^\infty - P_\mu)\bar{B})^{-1} \bar{B}^\top (P^\infty - P_\mu)\bar{A} \\
& + (\bar{Q}^\infty - \bar{Q}_\mu) - (P^\infty - P_\mu) \\
= {} & \bar{A}^\top P^\infty \bar{A} - \bar{A}^\top P^\infty \bar{B} (\bar{R}_e + \bar{B}^\top P_e \bar{B})^{-1} \bar{B}^\top P_e \bar{A} - \bar{Q}_\mu \\
& - P_\mu - \bar{A}^\top P_\mu \bar{A} + \bar{A}^\top P_\mu \bar{B} K_e + \bar{Q}^\infty - P^\infty.
\end{aligned} \tag{5.74}$$

From (5.73), we can write (5.74) as

$$0 = \bar{A}^\top P^\infty \bar{A} + \bar{A}^\top P^\infty \bar{B} K_e + \bar{Q}^\infty - P^\infty. \tag{5.75}$$

One can write (5.72) as

$$\bar{B}^\top P_\mu \bar{A} + \bar{B}^\top P_e \bar{A} = (\bar{R}_\mu + \bar{R}_e + \bar{B}^\top (P_\mu + P_e)\bar{B}) K_e, \tag{5.76}$$

then substitution of $\bar{R}_\mu = \bar{R} - \bar{R}_e$ and $P_e = P^\infty - P_\mu$ gives

$$\begin{aligned}
0 = {} & \bar{B}^\top P^\infty \bar{A} - (\bar{R} + \bar{B}^\top P^\infty \bar{B}) K_e \\
= {} & K_e - (\bar{R} + \bar{B}^\top P^\infty \bar{B})^{-1} \bar{B}^\top P^\infty \bar{A}.
\end{aligned} \tag{5.77}$$

Given $K^{h+1} = DK_r^{h+1}$ and $\bar{Q}^{h+1} = D^\dagger \bar{Q}_r^{h+1} D$, the solutions obtained by Algorithm 8 in (5.69b) can be inferred from (5.77).

The converged solutions obtained through Algorithm 8 will be equal to $\bar{Q}_e$ and P_e if and only if (5.72) and (5.73) achieve a unique solution, which is characterized by $\bar{Q}_\mu = 0$, $\bar{R}_\mu = 0$, and $P_\mu = 0$. However, if $\bar{Q}_\mu \neq 0$, $\bar{R}_\mu \neq 0$, and $P_\mu \neq 0$, then obtained $\bar{Q}^\infty$, P^∞, and $\bar{R}^\infty$ will differ from $\bar{Q}_e$, P_e, and $\bar{R}_e$, while still producing the same optimal gain $K^\infty = K_e$. This implies that the reward weights leading to the same optimal control gain might not be unique. □

Theorem 13. ***Stability analysis.*** *Given Lemma 6 and Theorems 10, 11, for each iteration index $h = 0, 1, \ldots,$ initiating with $\bar{Q}_r^0 \geq 0$ and $\bar{R}_r > 0$, one can show $P^{h+1} = (P^{h+1})^\top > 0$ such that*

$$P^{h+1} - \bar{A}^\top P^{h+1} \bar{A} + \bar{A}^\top P^{h+1} \bar{B} \times (\bar{R} + \bar{B}^\top P^{h+1} \bar{B})^{-1} \bar{B}^\top P^{h+1} \bar{A} \geq 0. \tag{5.78}$$

Furthermore, with K_r^{h+1} in (5.51b)*, the control input*

$$\bar{u}_k = -K^{h+1} \bar{x}_k \tag{5.79}$$

stabilizes MAS in (5.2) *and minimizes $J(\bar{u}_k, \bar{x}_k)$ in* (5.9)*.*

Proof. Given (5.64), for each iteration index $h = 0, 1, \ldots,$ $\bar{Q}^{h+1} \geq 0$. After selecting a suitable scalar $\delta \in (0, 1]$, we have $\bar{Q}^{h+1} - (1-\delta)\bar{Q}^h \geq 0$. If P^{h+1} satisfies (5.14) with $\bar{Q}^h \geq 0$, then from (5.63), $P^{h+1} = (P^{h+1})^\top > 0$. As shown in Lemma 7, $\bar{Q}_r = (D^\dagger)^\top \bar{Q} D^\dagger \geq 0$. Then, we have condition $D^\dagger \bar{Q}_r^{h+1} D - (1-\delta) D^\dagger \bar{Q}_r^{h+1} D \geq 0$. This ensures that P_r^{h+1} in (5.53) satisfies with $\bar{Q}_r^h \geq 0$. Theorem 10 implies that $K_r = D^\dagger K$ and $P_r = DPD^\dagger > 0$ satisfy (5.53) and (5.54). Also, based on (5.58), $\bar{Q}_r^h \geq 0$ and $P_r^{h+1} = (P_r^{h+1})^\top > 0$, one can see that (5.78) holds.

Consider J in (5.9), where K^{h+1} and $\bar{Q}^{h+1}$ are updated by (5.12a) and (5.63), respectively. Then, we have

$$\begin{aligned}
&J(\bar{x}_{k+1})^{h+1} - J(\bar{x}_k)^{h+1} \\
&= \bar{x}_{k+1}^\top P^{h+1} \bar{x}_{k+1} - \bar{x}_k^\top P^{h+1} \bar{x}_k \\
&= \bar{x}_k^\top ((\bar{A} - \bar{B}K^{h+1})^\top P^{h+1} (\bar{A} - \bar{B}K^{h+1}) - P^{h+1}) \bar{x}_k \\
&= \bar{x}_k^\top \left(\frac{(1-\delta)}{\delta} \bar{Q}^h - \frac{\bar{Q}^{h+1}}{\delta} - {K^{h+1}}^\top \bar{R} K^{h+1} \right) \bar{x}_k. \qquad (5.80)
\end{aligned}$$

Provided $\bar{Q}^{h+1} - (1-\delta)\bar{Q}^h \geq 0$ and $\bar{R} > 0$, we have

$$J(\bar{x}_{k+1})^{h+1} - J(\bar{x}_k)^{h+1} < 0. \qquad (5.81)$$

As shown in Lemma 7, $\bar{u}_k^*$ in (5.13) is learned using the η_k instead of $\bar{x}_k$, and $\bar{J}$ in (5.32b) is close to the optimal J in (5.9). Thus, control input $\bar{u}_k$ stabilizes MAS in (5.2). To show optimality, consider

$$\begin{aligned}
&\sum_{w=k}^{\infty} \left[J(\bar{x}_{w+1})^{h+1} - J(\bar{x}_w)^{h+1} \right] \\
&= \sum_{w=k}^{\infty} \bar{x}_w^\top ((\bar{A} - \bar{B}K^{h+1})^\top P^{h+1} (\bar{A} - \bar{B}K^{h+1}) - P^{h+1}) \bar{x}_w \\
&= \sum_{w=k}^{\infty} \bar{x}_w^\top (\bar{A}^\top P^{h+1} \bar{A} - P^{h+1}) \bar{x}_w + \bar{u}_w^\top \bar{B}^\top P^{h+1} \bar{A} \bar{x}_w \\
&\quad + \bar{x}_w^\top \bar{A}^\top P^{h+1} \bar{B} \bar{u}_w + \bar{u}_w^\top \bar{B}^\top P^{h+1} \bar{B} \bar{u}_w \\
&= \bar{x}_\infty^\top P^{h+1} \bar{x}_\infty - \bar{x}_k^\top P^{h+1} \bar{x}_k. \qquad (5.82)
\end{aligned}$$

Since $\bar{x}_\infty = 0$, and using (5.63), we have

$$\begin{aligned}
&\sum_{w=k}^{\infty} [\bar{x}_w^\top (\bar{A}^\top P^{h+1} \bar{B} (\bar{R} + \bar{B}^\top P^{h+1} \bar{B})^{-1} \bar{B}^\top P^{h+1} \bar{A} \\
&\quad + \left(\frac{(1-\delta)}{\delta} \bar{Q}^h - \frac{\bar{Q}^{h+1}}{\delta} \right)) \bar{x}_w + \bar{u}_w^\top \bar{B}^\top P^{h+1} \bar{A} \bar{x}_w + \bar{x}_w^\top \\
&\quad \times \bar{A}^\top P^{h+1} \bar{B} \bar{u}_w + \bar{u}_w^\top \bar{B}^\top P^{h+1} \bar{B} \bar{u}_w] + \bar{x}_k^\top P^{h+1} \bar{x}_k = 0. \qquad (5.83)
\end{aligned}$$

The performance function $J(\bar{x}_k)^{h+1} \approx \bar{J}(\bar{\eta}_k)^{h+1}$ obtains its minimum provided $\bar{Q}_r^{h+1} - (1-\delta)\bar{Q}_r^h \geq 0$, $\delta \in (0,1]$, $\bar{R} > 0$, and $\bar{u}_k = -K^{h+1}\bar{x}_k$, where K_r^{h+1} is computed by (5.51b). □

5.3.5 Scalable Performance of Algorithm 8

We demonstrate that Algorithm 8 is computationally manageable. Given Remark 31, optimal control learning requires, at least, $s \geq ((nN+pN)(nN+pN+1)/2)$ data samples at each iteration to find a unique solution to (5.24) using BLS. In contrast, Algorithm 8 requires, at least, $s \geq ((r+pN)(r+pN+1)/2)$ data samples at each iteration. Considering s as data samples and d_p as unknown parameters, the computational cost of BLS for $s \geq d_p$ is of the order $O(d_p^2 s)$ [110]. When finding an optimal control for large N, (5.24) has higher complexity of $O\big(((nN+pN)(nN+pN+1)/2)^2 s\big)$. This is in contrast to Algorithm 8 that has $O\big(((r+pN)(r+pN+1)/2)^2 s\big)$, where $r << (nN)$.

Similarly, IOC learning requires, at least, $s \geq (nN(nN+1)/2)$ data samples to find a unique solution to (5.27) using BLS. Conversely, Algorithm 8 requires, at least, $s \geq (r(r+1)/2)$ data samples at each iteration. When finding a reward function using IOC for large N, IOC learning by (5.27) has a higher computational cost, of the order $O\big(((nN(nN+1)/2))^2 s\big)$, unlike Algorithm 8 with $O\big((r(r+1)/2)^2 s\big)$.

Thus, Algorithm 8 is data-efficient as r is much smaller than nN. Comparative simulation studies, at the end of Section IV, will support this assertion.

5.4 Simulation Results

This section validates the scalability of Algorithm 8 with a large-scale MAS.

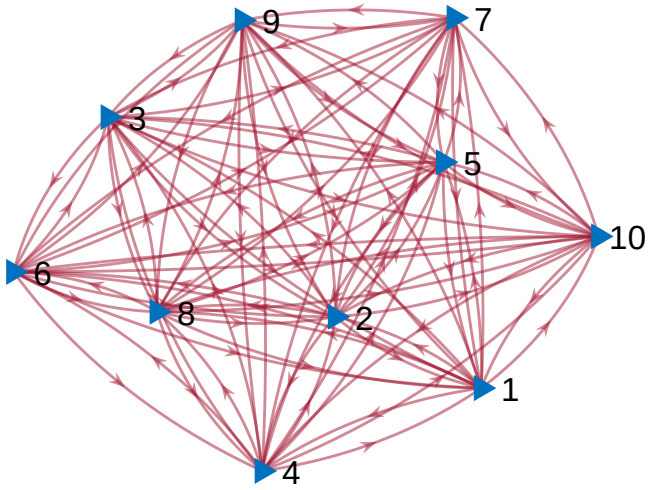


Figure 5.2. Large-scale consensus network..

We consider the interconnected MASs with their connection shown in Fig. 5.2. Each agent operates independently in terms of dynamics but shares a common performance function. The system matrices of each agent are adopted from [156] as

$$A = \begin{bmatrix} 0.9064 & 0.0816 & -0.0005 \\ 0.0743 & 0.9012 & -0.0007 \\ 0 & 0 & 0.1326 \end{bmatrix}, B = \begin{bmatrix} -0.0015 \\ -0.0096 \\ 0.8673 \end{bmatrix}. \tag{5.84}$$

We consider $N = 10$ in (5.2) and (5.9), where $\bar{x} = (x_1^{\mathrm{T}}, \ldots, x_{10}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{30}$, $x_i \in \mathbb{R}^3$ is the state of each agent i, $\bar{A} = I_{10} \otimes A \in \mathbb{R}^{30\times 30}$, and $\bar{B} = I_{10} \otimes B \in \mathbb{R}^{30\times 10}$. The performance function (5.9) captures the mutual interaction between the agents and quantifies the network's performance. To construct (5.9), we initialize the matrices $\bar{Q} = I_{10} \otimes Q_1 + \mathscr{L} \otimes Q_2$ and $\bar{R} = I_{10} \otimes R$, where $\mathscr{L} \in \mathbb{R}^{10\times 10}$, $Q_1 = 3 \times I_3$, $Q_2 = 0.5 \times I_3$, and $R = 1$.

To provide a general analysis, we consider $R \neq R_e$ and $\bar{R} \neq \bar{R}_e$. We select $\bar{Q}_e = I_{10} \otimes Q_{e_1} + \mathscr{L} \otimes Q_{e_2}$ and $\bar{R}_e = I_{10} \otimes R$ in (5.5), where $\mathscr{L} \in \mathbb{R}^{10\times 10}$, $Q_{e_1} = 3 \times I_3$, $Q_{e_2} = 0.5 \times I_3$, and $R_e = 2$. We gather the state observations by applying control input $\bar{u}_k$ in (5.13) and collect the state observations $\bar{x}_k$ for k ranging from 0 to 1800. Moreover, we select the

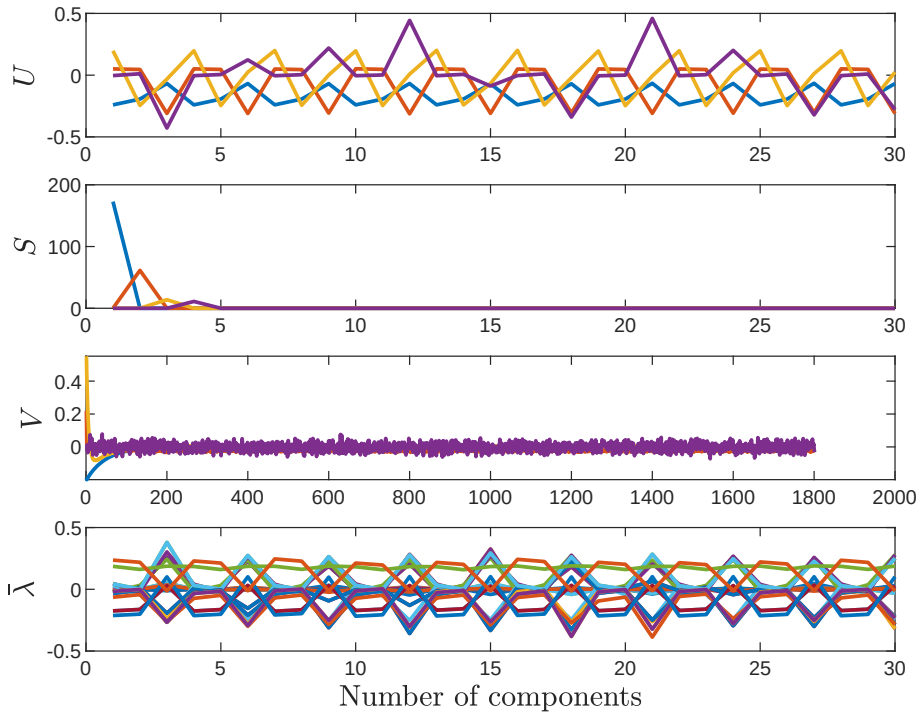


Figure 5.3. Dynamic modes and SVD matrices of the consensus network with ten agents..

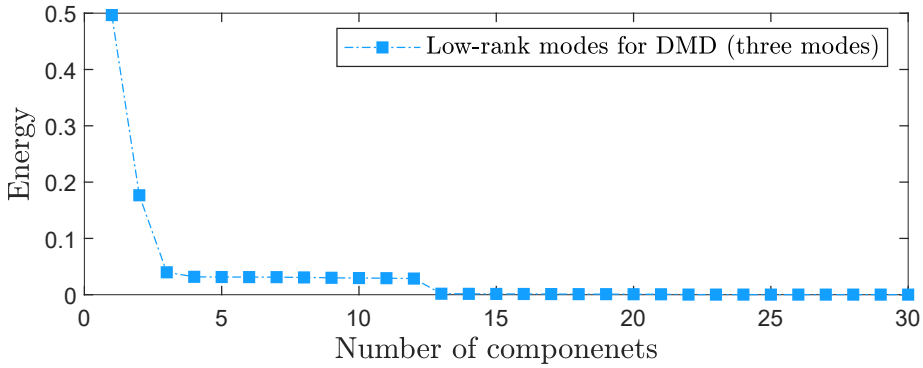


Figure 5.4. Energy content of each σ_j, i.e., $E_{\sigma_j} = \frac{\sigma_j}{\sum_{j=1}^{nN} \sigma_j}$. .

random probing noise $rand(10,1)$, learning rate $\delta = 1$, and the small threshold value of $e = 0.001$.

Figure 5.3 depicts all dynamic modes $\bar{\lambda} \in \mathbb{R}^{30\times 30}$ and SVD matrices $U \in \mathbb{R}^{30\times 30}$, $S \in \mathbb{R}^{30\times 30}$, and $V \in \mathbb{R}^{1800\times 30}$ for the original MAS (5.2). Figure 5.4 shows that it is

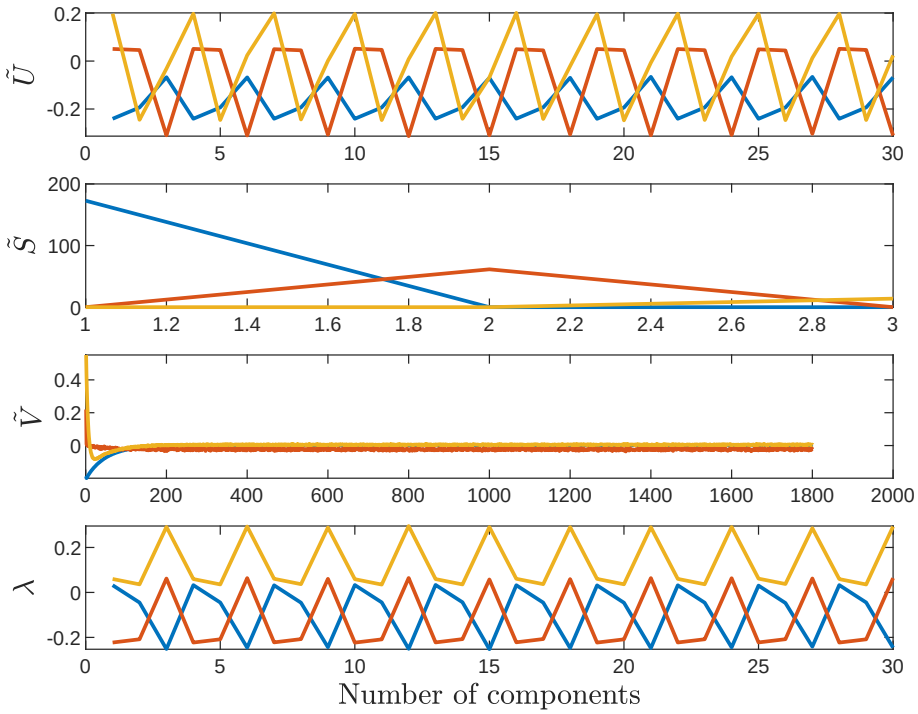


Figure 5.5. Dominant modes of the consensus network with ten agents..

lossless to capture the information by using the first 3 dominant singular. Figure 5.5 shows the dominant dynamic modes $\lambda \in \mathbb{R}^{30\times 3}$ and SVD matrices $\tilde{U} \in \mathbb{R}^{30\times 3}$, $\tilde{S} \in \mathbb{R}^{3\times 3}$, and $\tilde{V} \in \mathbb{R}^{1800\times 3}$ of the low-dimensional system.

The lower dimensional feedback gain K_r^{h+1} and reward weight $\bar{Q}_r^h$ are computed by (5.51b) and (5.51c) in Algorithm 8. Figure 5.6 demonstrates the convergence of K_r, $\bar{Q}_r$, and L_r. Figure 5.7 depicts the differences between the learned weights K_r, $\bar{Q}_r$, L_r and the respective target's ones K_{e_r}, $\bar{Q}_{e_r}$, L_{e_r}. The converged solutions K^∞ and $\bar{Q}^\infty$ are obtained by using (5.61) and (5.62). Figure 5.7 illustrates that the converged values of $\bar{Q}_r$ and K_r closes to the target value $\bar{Q}_{e_r}$ and K_{e_r} exhibit a satisfactory reward-shaping. Applying the K^∞, the MAS in Figure 5.2 finds a performance function comparable to the target MASs.

Table 5.1 compares the computational time by inverse RL with and without DMD preconditioning. By comparing, Algorithm 8 with DMD requires fewer data samples for

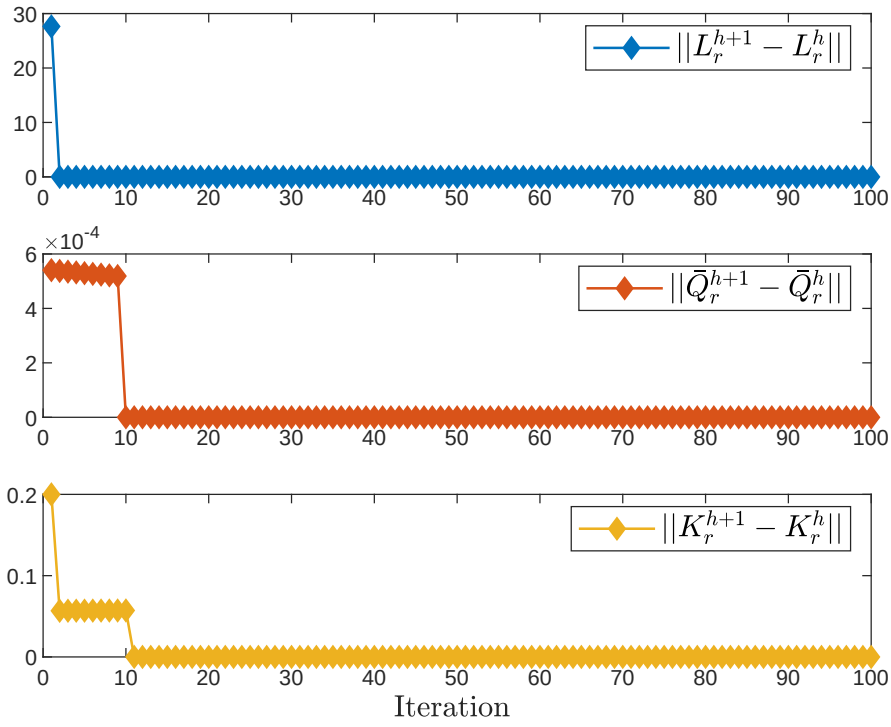


Figure 5.6. Convergence of L_r^h, $\bar{Q}_r^h$, and K_r^h..

each iteration and shorter computational time for reward-shaping. The computations are performed in MATLAB 2021a with an Intel(R) Xeon(R) W-10855M CPU operating at 2.80 GHz with 32 GB RAM.

5.5 Conclusion

This paper proposes a data-efficient model-free inverse RL approach for reward shaping in large-scale MASs. Our approach employs DMD for lossless dimensionality reduction and enhances the scalability of the inverse RL. This reduction is achieved entirely in a model-free manner using state measurements. The convergence and stability of the large-scale MAS in inverse RL are guaranteed. Simulation studies demonstrate the effectiveness of the method.

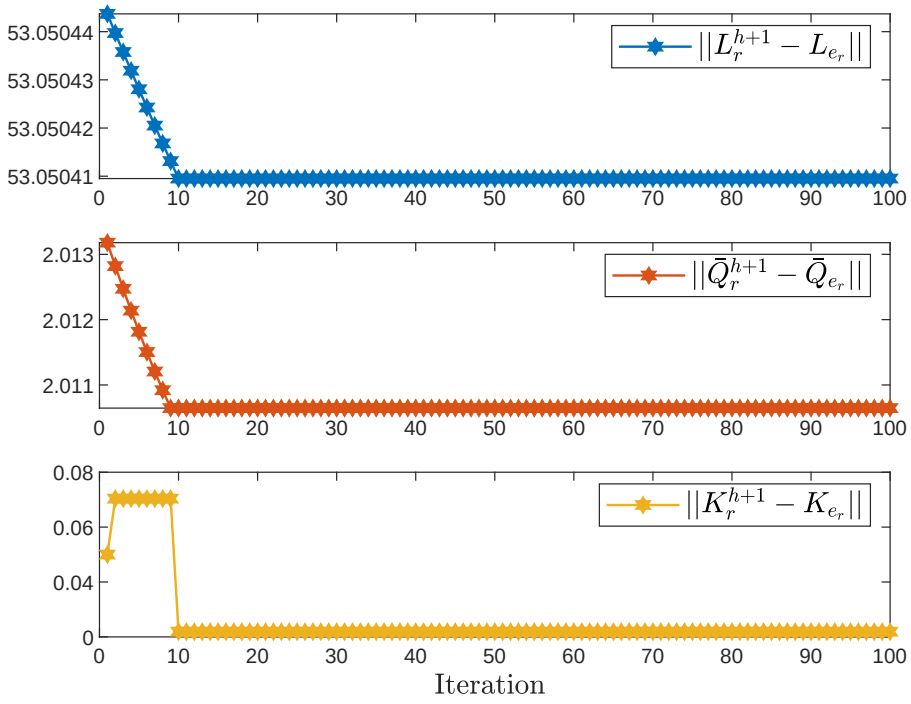


Figure 5.7. Difference between L_r^{h+1}, $\bar{Q}_r^{h+1}$, K_r^{h+1} and L_{e_r}, $\bar{Q}_{e_r}$, and K_{e_r}..

Table 5.1. Computational time for reward-shaping using our proposed Algorithm 8

	Spatio-temporal dimensions of state	Dimension of operators	Computational time (*Seconds*)
Without Pre-conditioning	$\mathbb{R}^{30\times1800}$	$\rho_k \in \mathbb{R}^{1\times820}$ $\theta_k \in \mathbb{R}^{1\times465}$	545.347
With Precon-ditioning	$\mathbb{R}^{3\times1800}$	$\rho_{rk} \in \mathbb{R}^{1\times91}$ $\theta_{rk} \in \mathbb{R}^{1\times6}$	1.07

CHAPTER 6

Future Work

We foresee several potential research avenues that could enhance and build upon our current work.

1. Developing an output feedback (OPFB) controller when the system matrices are unknown, and only have access to output and input data can be challenging. This problem falls under the domain of system identification and control, and it typically involves designing a controller that can adapt to unknown system dynamics based on observed output data. The main objective of this work will be to show the integration of DMD within RL to offer a computationally manageable OPFB control solution suitable for extensive networks. The key contributions of this paper will encompass:
    - Introducing an off-policy RL controller tailored for large-scale systems, utilizing only output measurements to derive optimal control policies.
    - Illustrating the limitations in scalability and efficiency associated with the model-free RL approach for OPFB control within network systems.
    - Developing a data-efficient RL-based OPFB controller, which will leverage off-policy RL and DMD, ensuring scalability, bias-free solutions, and eliminating the necessity for system state measurements during the learning process.
    - Providing rigorous theoretical proofs and demonstrating the algorithm's effectiveness through extensive numerical analyses.
2. In future expansions, there's potential to adapt the tools proposed in Chapter 2 for cooperative control among multiple power microgrids. The synchronization of islanded multi-microgrids' states before reconnection is a critical aspect. The "learner" micro-

grids, utilizing demonstrated trajectories from the ”expert” microgrids, can employ inverse RL to mirror the unknown cost functions. This approach aims to synchronize learner states with the common values established by the expert microgrids. Possible expansions could involve heterogeneous nonlinear multi-agent systems or variations in graph topologies between expert and learner MAS.

3. A potential direction ahead could explore DMD-based RL for high-dimensional systems featuring heterogeneous subsystems, and extended LQR formulations.
4. For future work, investigating the inclusion of external uncontrolled noise within the data collection process could be a prospective avenue to bolster the robustness of the proposed approach in Chapter 4.

REFERENCES

[1] IEEE recommended practice for excitation system models for power system stability studies. *IEEE Std 421.5-2016 (Revision of IEEE Std 421.5-2005)*, pages 1–207, 2016.

[2] Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. *Proc. of the 24th Annu. Conf. Learn. Theory*, pages 1–26, 2011.

[3] P. Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21th Annual International Conference on Machine Learning*, New York, USA, July 2004.

[4] Pieter Abbeel, Adam Coates, and Andrew Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research*, 29(13):1608–1639, November 2010.

[5] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. *Proc. Int. Conf. Mach. Learn.*, pages 1–8, 2004.

[6] Asma Al-Tamimi, Frank L Lewis, and Murad Abu-Khalaf. Model-free Q-learning designs for linear discrete-time zero-sum games with application to h-infinity control. *Automatica*, 43(3):473–481, 2007.

[7] Paul M Anderson and Aziz A Fouad. *Power System Control And Stability*. John Wiley & Sons, 2008.

[8] Javad Ansari, Ali Reza Abbasi, and Bahman Bahmani Firouzi. Decentralized LMI-based event-triggered integral sliding mode lfc of power systems with disturbance observer. *Int. J. Electr. Power Energy Syst.*, 138:107971, 2022.

[9] Athanasios C Antoulas. *Approximation of large-scale dynamical systems*. Philadelphia, PA, USA: SIAM, 2005.

[10] Hassan Arbabi, Milan Korda, and Igor Mezić. A data-driven koopman model predictive control framework for nonlinear partial differential equations. In *IEEE Conf. Decis. Control*, pages 6409–6414, 2018.

[11] Karl J Åström and Björn Wittenmark. *Adaptive Control*. Courier Corporation, 2013.

[12] Christopher G. Atkeson and Stefan Schaal. Robot learning from demonstration. In *International Conference on Machine Learning*, volume 97, pages 12–20, San Francisco, CA, USA, July 1997.

[13] V. Balakrishnan and L. Vandenberghe. Semidefinite programming duality and linear time-invariant systems. *IEEE Trans. Autom. Control*, 48(1):30–41, 2003.

[14] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Syst., Man, Cybern.*, SMC-13(5):834–846, 1983.

[15] D. P. Bertsekas. *Dynamic Programming and Optimal Control: Vol. 2*. Athena Scientific, Belmont, 2000.

[16] D. P. Bertsekas. *Rollout, Policy Iteration, and Distributed Reinforcement Learning*. Athena scientific, Belmont, 2020.

[17] Dimitri Bertsekas. *Dynamic Programming And Optimal Control: Volume I*, volume 1. Athena scientific, 2012.

[18] Tao Bian and Zhong-Ping Jiang. Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design. *Automatica*, 71:348–360, 2016.

[19] Aude Billard, Sylvain Calinon, Rudiger Dillmann, and Stefan Schaal. Robot programming by demonstration. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*. Springer, Berlin, Heidelberg, 2008.

[20] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag, Berlin, Germany, 2006.

[21] Francesco Borrelli and TamÁs Keviczky. Distributed LQR design for identical dynamically decoupled systems. *IEEE Trans. Autom. Control*, 53(8):1901–1912, 2008.

[22] S.J. Bradtke, B.E. Ydstie, and A.G. Barto. Adaptive linear quadratic control using policy iteration. *Proc. of Amer. Control Conf.*, 3:3475–3479, 1994.

[23] Steven J Bradtke, B Erik Ydstie, and Andrew G Barto. Adaptive linear quadratic control using policy iteration. *Amer. Control Conf.*, 3:3475–3479, 1994.

[24] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[25] Marko Budišić, Ryan Mohr, and Igor Mezić. Applied koopmanism. *Chaos: Interdiscip. J. Nonlinear Sci.*, 22(4):047510, 2012.

[26] Ci Chen, Frank L Lewis, and Bo Li. Homotopic policy iteration-based learning design for unknown linear continuous-time systems. *Automatica*, 138:110153, 2022.

[27] Teddy M. Cheng and Andrey V. Savkin. Decentralized control of multi-agent systems for swarming with a given geometric pattern. *International Journal of Computers Mathematics with Applications*, 61(4):731–744, Feb 2011.

[28] Sonia Chernova and Andrea L. Thomaz. *Robot Learning from Human Teachers*, volume 8 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. April 2014.

[29] Ronghu Chi, Huimin Zhang, Biao Huang, and Zhongsheng Hou. Quantitative data-driven adaptive iterative learning control: From trajectory tracking to point-to-point tracking. *IEEE Trans. Cybern.*, 52(6):4859–4873, 2022.

[30] J. Chow and P. Kokotovic. A decomposition of near-optimum regulators for systems with slow and fast modes. *IEEE Trans. Autom. Control*, 21(5):701–705, 1976.

[31] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Adv. in neural Inf. process. Syst.*, 31, 2018.

[32] Robert Crites and Andrew Barto. An actor/critic algorithm that is equivalent to Q-learning. *Advances Neural Inf. Process. Syst.*, 7:401–408, 1994.

[33] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Found. Comput. Math.*, 20(4):633–679, 2020.

[34] Vrushabh S. Donge, Bosen Lian, Frank L. Lewis, and Ali Davoudi. Accelerated reinforcement learning via dynamic mode decomposition. *IEEE Trans. Control Netw. Syst. doi:10.1109/TCNS.2023.3259060*, pages 1–12, 2023.

[35] Vrushabh S. Donge, Bosen Lian, Frank L. Lewis, and Ali Davoudi. Multiagent graphical games with inverse reinforcement learning. *IEEE Trans. Control Netw. Syst.*, 10(2):841–852, 2023.

[36] D.L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, 1995.

[37] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse optimal control with linearly-solvable MDPs. *Proc. 27th Int. Conf. Mach. Learn.*, pages 335–342, 2010.

[38] Carl Eckart and G. Marion Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[39] Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. *Proc. Int. Conf. Mach. Learn.*, 80:1467–1476, 2018.

[40] Gene F Franklin, J David Powell, and Michael L Workman. *Digital control of dynamic systems*. Boston, MA, USA: Addison-wesley, 1998.

[41] Matan Gavish and David L. Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014.

[42] Keith Glover. All optimal hankel-norm approximations of linear multivariable systems and their L^{∞}-error bounds. *Int. J. Control*, 39(6):1115–1193, 1984.

[43] Graham C Goodwin and Kwai Sang Sin. *Adaptive Filtering Prediction And Control*. Courier Corporation, 2014.

[44] Marek Grzes. Reward shaping in episodic reinforcement learning. *Conf. Auton. Agents and Multi-Agent Syst.*, pages 565–573, 2017.

[45] W. M. Haddad and V. Chellaboina. *Nonlinear Dynamical Systems and Control: A Lyapunov-based Approach*. Princeton University Press, Oxford, 2008.

[46] Wassim M Haddad and VijaySekhar Chellaboina. *Nonlinear dynamical systems and control: a Lyapunov-based approach*. Princeton university press, 2011.

[47] D. Hadfield-Menell, A. Dragan, P. Abbeel, and S. Russell. Cooperative inverse reinforcement learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3916–3924, Red Hook, NY, USA, December 2016.

[48] Wenjie Han and Aleksandar M Stanković. Model-predictive control design for power system oscillation damping via excitation- A data-driven approach. *IEEE Trans. Power Syst.*, 38(2):1176–1188, 2023.

[49] Wenjie Han and Aleksandar M. Stanković. Model-predictive control design for power system oscillation damping via excitation - a data-driven approach. *IEEE Trans. Power Syst.*, pages 1–1, 2022.

[50] Yiqiang Han, Wenjian Hao, and Umesh Vaidya. Deep learning of koopman representation for control. *IEEE Conf. Decis.Control*, pages 1890–1895, 2020.

[51] Masih Haseli and Jorge Cortés. Learning koopman eigenfunctions and invariant subspaces from data: Symmetric subspace decomposition. *IEEE Trans. Autom, Control*, 67(7):3442–3457, 2022.

[52] G. Hewer. An iterative technique for the computation of the steady state gains for the discrete optimal regulator. *IEEE Trans. Autom. Control*, 16(4):382–384, 1971.

[53] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4572–4580, Red Hook, NY, USA, 2016.

[54] Philip Holmes, John L Lumley, Gahl Berkooz, and Clarence W Rowley. *Turbulence, Coherent Structures, Dynamical Systems, And Symmetry*. Cambridge university press, 2012.

[55] Morteza Ibrahimi, Adel Javanmard, and Benjamin Roy. Efficient reinforcement learning for high dimensional linear quadratic systems. *Adv. in Neural Inf. Process. Syst.*, 25, 2012.

[56] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, and Jong Wook Kim. Q-learning algorithms: A comprehensive classification and applications. *IEEE Access*, 7:133653–133667, 2019.

[57] Yi Jiang, Bahare Kiumarsi, Jialu Fan, Tianyou Chai, Jinna Li, and Frank L. Lewis. Optimal output regulation of linear discrete-time systems with unknown dynamics using reinforcement learning. *IEEE Trans. Cybern.*, 50(7):3147–3156, 2020.

[58] Yu Jiang and Zhong-Ping Jiang. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*, 48(10):2699–2704, 2012.

[59] Gangshan Jing, He Bai, Jemin George, and Aranya Chakrabortty. Model-free optimal control of linear multiagent systems via decomposition and hierarchical approximation. *IEEE Trans. Control Netw. Syst.*, 8(3):1069–1081, 2021.

[60] M. Johnson, N. Aghasadeghi, and T. Bretl. Inverse optimal control for deterministic continuous-time nonlinear systems. In *52nd IEEE Conference on Decision and Control*, pages 2906–2913, Dec 2013.

[61] Marcus Johnson, Shubhendu Bhasin, and Warren E Dixon. Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm. *IEEE Conf. Decis. Control*, pages 142–147, 2011.

[62] J. Juang and Richard Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *J. Guid., Control, Dyn.*, 8(5):620–627, 1985.

[63] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of koopman eigenfunctions for control. *Mach. Learn.: Sci. and Technol.*, 2(3):035023, 2021.

[64] Rudolf E Kalman. Canonical structure of linear dynamical systems. *Proc. of the Nat. Acad. of Sci.*, 48(4):596–600, 1962.

[65] Rudolf E. Kálmán. When is a linear control system optimal. *J. Basic Eng.*, 86:51–60, 1964.

[66] Rudolf Emil Kalman. When is a linear control system optimal? *Trans. of the ASME. Series D. Journal of Basic Engineering*, 86:51–60, 1964.

[67] Kenji Kashima. Noise response data reveal novel controllability gramian for nonlinear network dynamics. *Sci. Rep.*, 6:1–8, 2016.

[68] Hassan K. Khalil and J. W. Grizzle. *Nonlinear Systems*, volume 3. Prentice-Hall, NJ, 2002.

[69] Bahare Kiumarsi, Hamidreza Modares, Frank L. Lewis, and Zhong-Ping Jiang. H_∞ optimal control of unknown linear discrete-time systems: An off-policy reinforcement learning approach. In *IEEE Conf. Robot., Autom., Mechatron.*, pages 41–46, 2015.

[70] Bahare Kiumarsi, Kyriakos G. Vamvoudakis, Hamidreza Modares, and Frank L. Lewis. Optimal and autonomous control using reinforcement learning: A survey. *IEEE Trans. Neural Netw. and Learning Syst.*, 29(6):2042–2062, 2018.

[71] D. Kleinman. On an iterative technique for riccati equation computations. *IEEE Trans. Autom. Control*, 13(1):114–115, 1968.

[72] D. Kleinman. Stabilizing a discrete, constant, linear system with application to iterative methods for solving the riccati equation. *IEEE Trans. Autom. Control*, 19(3):252–254, 1974.

[73] Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.

[74] Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proc. of the Nat. Acad. of Sci.*, 17(5):315–318, 1931.

[75] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018.

[76] J. Kutz, Steven Brunton, Bingni Brunton, and Joshua Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Philadelphia, PA, USA: SIAM, 2016.

[77] Huibert Kwakernaak and Raphael Sivan. *Linear optimal control systems*. Wiley-interscience New York, 1972.

[78] Frank L. Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[79] Frank L Lewis, Draguna Vrabie, and Vassilis L Syrmos. *Optimal control*. Hoboken, NJ, USA: John Wiley & Sons, 2012.

[80] Jinna Li, Zhenfei Xiao, Jialu Fan, Tianyou Chai, and Frank L Lewis. Off-policy Q-learning: Solving nash equilibrium of multi-player games with network-induced delay and unmeasured state. *Automatica*, 136:110076, 2022.

[81] Ting Li, Dongsheng Yang, Xiangpeng Xie, and Huaguang Zhang. Event-triggered control of nonlinear discrete-time system with unknown dynamics based on HDP(λ). *IEEE Trans. Cybern.*, 52(7):6046–6058, 2022.

[82] Yongming Li, Yanjun Liu, and Shaocheng Tong. Observer-based neuro-adaptive optimized control of strict-feedback nonlinear systems with state constraints. *IEEE Trans. Neural Netw. Learn. Syst.*, 33(7):3131–3145, 2022.

[83] B. Lian, W. Xue, F. L. Lewis, and T. Chai. Online inverse reinforcement learning for nonlinear systems with adversarial attacks. *International Journal of Robust and Nonlinear Control*, 31(14):6646–6667, 2021.

[84] Bosen Lian, Vrushabh S Donge, Frank L Lewis, Tianyou Chai, and Ali Davoudi. Data-driven inverse reinforcement learning control for linear multiplayer games. *IEEE Trans. Neural Netw. and Learn. Syst.*, 2022, doi: 10.1109/TNNLS.2022.3186229.

[85] Bosen Lian, Vrushabh S Donge, Wenqian Xue, Frank L Lewis, and Ali Davoudi. Distributed minmax strategy for multiplayer games: Stability, robustness, and algorithms. *IEEE Trans. Neural Netw. Learn. Syst., doi: 10.1109/TNNLS.2022.3215629*, 2022.

[86] Bosen Lian, Wenqian Xue, Frank L. Lewis, and Tianyou Chai. Inverse reinforcement learning for adversarial apprentice games. *IEEE Trans. Neural Netw. and Learn. Syst. doi:10.1109/TNNLS.2021.3114612*, pages 1–14, 2021.

[87] Bosen Lian, Wenqian Xue, Yijing Xie, Frank L. Lewis, and Ali Davoudi. Off-policy inverse q-learning for discrete-time antagonistic unknown systems. *Automatica://doi.org/10.1016/j.automatica.2023.111171*, 155:111171, 2023.

[88] Victor G. Lopez, Mohammad Alsalti, and Matthias A. Müller. Efficient off-policy Q-learning for data-based discrete-time LQR problems. *IEEE Trans. Autom. Control*, 68(5):2922–2933, 2023.

[89] Jingwei Lu, Qinglai Wei, Tianmin Zhou, Ziyang Wang, and Fei-Yue Wang. Event-triggered near-optimal control for unknown discrete-time nonlinear systems using parallel control. *IEEE Trans. Cybern.*, 53(3):1890–1904, 2023.

[90] Biao Luo, Huai-Ning Wu, and Tingwen Huang. Off-policy reinforcement learning for H_∞ control design. *IEEE Trans. Cybern.*, 45(1):65–76, 2015.

[91] 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 Trans. Cybern*, 51(7):3630–3640, 2021.

[92] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Commun.*, 9(1):1–10, 2018.

[93] Dominique Luzeaux. *Engineering Large-Scale Complex Systems*. John Wiley & Sons, Ltd, 2013.

[94] Magdi S. Mahmoud and Yuanqing Xia. *Networked Control Systems*. Cambridge, MA, USA: Butterworth-Heinemann, 2019.

[95] H. Modares, F. L. Lewis, and Z. 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, Oct 2015.

[96] 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.

[97] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Autom. Control*, 26(1):17–32, 1981.

[98] K. H. Movric and F. L. Lewis. Cooperative optimal control for multi-agent systems on directed graph topologies. *IEEE Transactions on Automatic Control*, 59(3):769–774, March 2014.

[99] J. Moyalan, Hyungjin Choi, Yongxin Chen, and Umesh Vaidya. Data-driven optimal control via linear transfer operators: A convex approach. *Automatica*, 150:110841, 2023.

[100] Chaoxu Mu, Ding Wang, and Haibo He. Data-driven finite-horizon approximate optimal control for discrete-time nonlinear systems using iterative HDP approach. *IEEE Trans. Cybern.*, 48(10):2948–2961, 2017.

[101] Sayak Mukherjee, He Bai, and Aranya Chakrabortty. On model-free reinforcement learning of reduced-order optimal control for singularly perturbed systems. *IEEE Conf. Decis. Control*, pages 5288–5293, 2018.

[102] S. Natarajan, G. Kunapuli, K. Judah, P. Tadepalli, K. Kersting, and J. Shavlik. Multi-agent inverse reinforcement learning. In *International Conference on Machine Learning and Applications*, pages 395–400, Washington D.C., USA, 2010.

[103] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. *Int. Conf. Mach. Learn.*, 99:278–287, 1999.

[104] Andrew Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670, 2000.

[105] Muhammad Umar B. Niazi, Carlos Canudas-de Wit, and Alain Y. Kibangou. Average state estimation in large-scale clustered network systems. *IEEE Trans. Control Netw. Syst.*, 7(4):1736–1745, 2020.

[106] F. Ornelas-Tellez, E. N. Sanchez, A. G. Loukianov, and J. Jesus Rico. Robust inverse optimal control for discrete-time nonlinear system stabilization. *European Journal of Control*, 20(1):38–44, Jan 2014.

[107] Y. Peng, Q. Chen, and W. Sun. Reinforcement q-learning algorithm for H_∞ tracking control of unknown discrete-time linear systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(11):4109–4122, Nov 2020.

[108] B. Piot, M. Geist, and O. Pietquin. Bridging the gap between imitation learning and inverse reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 28(8):1814–1826, Aug 2017.

[109] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, volume 703. John Wiley & Sons, Hoboken, NJ, Mar. 2007.

[110] LA Prashanth, Nathaniel Korda, and R Munos. Fast LSTD using stochastic approximation: Finite time analysis and application to traffic control. *Joint Eur. Conf. Mach. Learn. and Knowl. Discovery in Databases*, pages 66–81, 2014.

[111] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.*, 15(1):142–161, 2016.

[112] J. Qin, M. Li, Y. Shi, Q. Ma, and W. X. Zheng. Optimal synchronization control of multiagent systems with input saturation via off-policy reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(1):85–96, Jan 2019.

[113] Dileep Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2586–2591, San Francisco, CA, USA, January 2007.

[114] Jette Randløv and Preben Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. *Int. Conf. Mach. Learn.*, 98:463–471, 1998.

[115] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Adv. in Neural Inf. Process. Syst.*, 32:350–360, 2019.

[116] Clarence W Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *Int. J. Bifurcation and Chaos*, 15(03):997–1013, 2005.

[117] Clarence W Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S Henningson. Spectral analysis of nonlinear flows. *J. Fluid Mech.*, 641:115–127, 2009.

[118] Tomonori Sadamoto and Aranya Chakrabortty. Fast real-time reinforcement learning for partially-observable large-scale systems. *IEEE Trans. Artif. Intell.*, 1(3):206–218, 2020.

[119] Tomonori Sadamoto, Aranya Chakrabortty, and Jun Imura. Fast online reinforcement learning control using state-space dimensionality reduction. *IEEE Trans. Control Netw. Syst.*, 8(1):342–353, 2021.

[120] Tomonori Sadamoto, Takayuki Ishizaki, and Jun-ichi Imura. Average state observers for large-scale network systems. *IEEE Trans. Control Netw. Syst.*, 4(4):761–769, 2017.

[121] E. N. Sanchez and F. Ornelas-Tellez. *Discrete-Time Inverse Optimal Control for Nonlinear Systems*. CRC Press, 1st edition, 2013.

[122] NR Sandell, Pravin Varaiya, Michael Athans, and M Safonov. A survey of decentralized control methods for large scale systems. *Systems Eng. for Power, US Dept. of Commerce*, pages 334–335, 1975.

[123] Aleksandar A Sarić, Mark K Transtrum, Andrija T Sarić, and Aleksandar M Stanković. Integration of physics-and data-driven power system models in transient analysis after major disturbances. *IEEE Syst. J.*, 17(1):479–490, 2023.

[124] Carsten Scherer and Siep Weiland. Linear matrix inequalities in control. *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 3(2), 2000.

[125] PETER J. SCHMID. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.*, 656:5–28, 2010.

[126] J. Song, H. Ren, D. Sadigh, and S. Ermon. Multi-agent generative adversarial imitation learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7472–7483, Red Hook, NY, USA, 2018.

[127] Lixing Song, Junheng Wang, and Junhong Xu. A data-efficient reinforcement learning method based on local koopman operators. *20th IEEE Int. Conf. Mach. Learn. Appl.*, pages 515–520, 2021.

[128] Ruizhuo Song, Qinglai Wei, Huaguang Zhang, and Frank L. Lewis. Discrete-time non-zero-sum games with completely unknown dynamics. *IEEE Trans. Cybern.*, 51(6):2929–2943, 2021.

[129] Yue Sun and Maryam Fazel. Learning optimal controllers by policy gradient: Global optimality via convex parameterization. *IEEE Conf. Decis. Control*, pages 4576–4581, 2021.

[130] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 2018.

[131] Umar Syed and Robert E. Schapire. A game-theoretic approach to apprenticeship learning. In *International Conference on Neural Information Processing Systems*, pages 1449–1456, Red Hook, NY, USA, December 2008.

[132] Marko Tanaskovic, Lorenzo Fagiano, Carlo Novara, and Manfred Morari. Data-driven control of nonlinear systems: An on-line direct approach. *Automatica*, 75:1–10, 2017.

[133] Wenchong Tian, Zhenliang Liao, Zhiyu Zhang, Hao Wu, and Kunlun Xin. Flooding and overflow mitigation using deep reinforcement learning based on koopman

operator of urban drainage systems. *Water Resour. Res.*, 58(7):e2021WR030939, 2022.

[134] Jonathan H Tu. *Dynamic mode decomposition: Theory and applications*. PhD thesis, Princeton University, 2013.

[135] Jonathan H Tu and Clarence W Rowley. An improved algorithm for balanced POD through an analytic treatment of impulse response tails. *J. Comput. Phys.*, 231(16):5317–5333, 2012.

[136] Jonathan H. Tu, Clarence Worth Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *J. Comput. Dyn.*, 1(2):391–421, 2014.

[137] M. Tucci, S. Riverso, J. C. Vasquez, J. M. Guerrero, and G. Ferrari-Trecate. A decentralized scalable approach to voltage control of dc islanded microgrids. *IEEE Transactions on Control Systems Technology*, 24(6):1965–1979, Nov 2016.

[138] M M Vainberg. *Variational method and method of monotone operators in the theory of nonlinear equations*. John Wiley & Sons, 1974.

[139] C. J. Vega, O. J. Suarez, E. N. Sanchez, G. Chen, S. Elvira-Ceja, and D. I. Rodriguez. Trajectory tracking on uncertain complex networks via nn-based inverse optimal pinning control. *IEEE Transactions on Neural Networks and Learning Systems*, 31(3):854–864, Mar 2020.

[140] Eleftherios E. Vlahakis, Leonidas D. Dritsas, and George D. Halikias. Distributed LQR-based suboptimal control for coupled linear systems. *Int. Fed. Autom. Control*, 52(20):109–114, 2019.

[141] Draguna Vrabie, O Pastravanu, Murad Abu-Khalaf, and Frank L Lewis. Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica*, 45(2):477–484, 2009.

[142] Krešimir Vrdoljak, Nedjeljko Perić, and Ivan Petrović. Sliding mode based load-frequency control in power systems. *Electr. Power Syst. Res.*, 80(5):514–527, 2010.

[143] D. Vrushabh, P. Akshay, K. Sonam, S. Wagh, and N. M. Singh. Robust path integral control on stochastic differential games. *IEEE Med. Conf. Control Automa.*, pages 665–670, 2020.

[144] Meixi Wang, Xuyang Lou, Wei Wu, and Baotong Cui. Koopman-based MPC with learned dynamics: Hierarchical neural network approach. *IEEE Trans. Neural Netw. and Learn. Syst.*, pages 1–10, 2022.

[145] Christopher JCH Watkins and Peter Dayan. Q-learning. *Mach. learn.*, 8:279–292, 1992.

[146] Q. Wei, R. Song, D. Liu, and B. Luo. Chapter nine- continuous-time distributed adaptive dynamic programming for heterogeneous multiagent optimal synchronization control. In K. G. Vamvoudakis and S. Jagannathan, editors, *Control of Complex Systems*, pages 275–304. Butterworth-Heinemann, 2016.

[147] K. Willcox and Jaime Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA J.*, 40(11):2323–2330, 2002.

[148] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data-driven approximation of the koopman operator: Extending dynamic mode decomposition. *J. of Nonlinear Sci.*, 25(6):1307–1346, 2015.

[149] Xiangpeng Xie, Cong Wei, Zhou Gu, and Kaibo Shi. Relaxed resilient fuzzy stabilization of discrete-time takagi–sugeno systems via a higher order time-variant balanced matrix method. *IEEE Trans. Fuzzy Syst.*, 30(11):5044–5050, 2022.

[150] W. Xue, P. Kolaric, J. Fan, B. Lian, T. Chai, and F. L. Lewis. Inverse reinforcement learning in tracking control based on inverse optimal control. *IEEE Transactions on Cybernetics*, 52(10):10570–10581, Oct 2022.

[151] 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 Trans. Cybern.*, 52(10):10570–10581, 2022.

[152] 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 Trans. Neural Netw. Learn. Syst.*, 34(5):2386–2399, 2023.

[153] Ziming Yan and Yan Xu. A multi-agent deep reinforcement learning method for cooperative load frequency control of a multi-area power system. *IEEE Trans. Power Syst.*, 35(6):4599–4608, 2020.

[154] Yongliang Yang, Zhishan Guo, Haoyi Xiong, Da-Wei Ding, Yixin Yin, and Donald C Wunsch. Data-driven robust control of discrete-time uncertain linear systems via off-policy reinforcement learning. *IEEE Trans. Neural Netw. Learn. Syst.*, 30(12):3735–3747, 2019.

[155] Yongliang Yang, Hamidreza Modares, Kyriakos G. Vamvoudakis, Wei He, Cheng-Zhong Xu, and Donald C. Wunsch. Hamiltonian-driven adaptive dynamic programming with approximation errors. *IEEE Trans. Cybern.*, 52(12):13762–13773, 2022.

[156] Lijing Zhai and Kyriakos G Vamvoudakis. A data-based private learning framework for enhanced security against replay attacks in cyber-physical systems. *Int. J. Robust Nonlinear Control*, 31(6):1817–1833, 2021.

[157] Huaipin Zhang, Ju H. Park, Dong Yue, and Xiangpeng Xie. Finite-horizon optimal consensus control for unknown multiagent state-delay systems. *IEEE Trans. Cybern.*, 50(2):402–413, 2020.

[158] Li Zhang, Jialu Fan, Wenqian Xue, Victor G. Lopez, Jinna Li, Tianyou Chai, and Frank L. Lewis. Data-driven H_∞ optimal output feedback control for linear discrete-time systems based on off-policy Q-learning. *IEEE Trans. Neural Netw. Learn. Syst.*, pages 1–15, 2021.

[159] Yang Zhang, Tao Huang, and Ettore Francesco Bompard. Big data analytics in smart grids: a review. *Energy Inform.*, 1(1):1–24, 2018.

[160] Wanbing Zhao, Hao Liu, Frank L. Lewis, and Xinlong Wang. Data-driven optimal formation control for quadrotor team with unknown dynamics. *IEEE Trans. Cybern.*, 52(8):7889–7898, 2022.

[161] Tong Zhou, Keyou You, and Tao Li. *Estimation and Control of Large Scale Networked Systems*. Butterworth-Heinemann, 2018.

[162] X. Zhou, W. Wang, T. Wang, Y. Lei, and F. Zhong. Bayesian reinforcement learning for multi-robot decentralized patrolling in uncertain environments. *IEEE Transactions on Vehicular Technology*, 68(12):11691–11703, Dec 2019.

BIOGRAPHICAL STATEMENT

Vrushabh S. Donge received the B.E. degree in electrical engineering from the Government College of Engineering, Aurangabad, Maharashtra, India, in 2016, and the M.Tech. degree in control systems from the Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India, in 2020. He is currently working toward the Ph.D. degree in electrical engineering with the University of Texas at Arlington, Arlington, TX, USA. His research interests include optimal control, reinforcement learning, multi-agent systems and distributed control.