Distributed Reinforcement Learning for Networked Dynamical Systems

Tomonori Sadamoto, *Member, IEEE*, Ayafumi Kikuya, and Aranya Chakrabortty, *Senior Member, IEEE*

***Abstract*— We propose a scalable algorithm for learning distributed optimal controllers for networked dynamical systems. Assuming that the network is comprised of nearly homogeneous subsystems, each sub-controller is trained by the local state and input information from its corresponding subsystem and filtered information from its neighbors. Thus, the costs of both learning and control become independent of the number of subsystems. We show the optimality and convergence of the algorithm for the case when the individual subsystems are identical, based on an algebraic property of such networks. Thereafter, we show the robustness of the algorithm when applied to general heterogeneous networks. The effectiveness of the design is investigated through numerical simulations.**

***Index Terms*— Distributed Reinforcement Learning, Scalability, Distributed Control, Networked Dynamical Systems**

I. Introduction

DISTRIBUTED control is a widely used control methodology for networked dynamic systems (NDS), with examples such as multi-agent networks [1]–[3], robotic networks [4], social networks [5], transportation networks [6] and smart grids [7]. With the recent emergence of advanced machine learning techniques for control [8], especially of reinforcement learning (RL) [9], several papers have been written on developing RL-based distributed control of networked systems [10]–[12]. The majority of these studies, however, only consider distributing the actuation of the learned controllers while the learning process itself still remains centralized. Due to this centralized structure, the computational complexity of learning increases notably as the number of subsystems in the network increases.

The limitation in scaling up RL algorithms has spurred recent developments in model-free distributed design [13] of controllers, whereby multiple subsystems can independently learn their local controllers through intercommunication. For the sake of convenience, we will hereafter refer to these methodologies as *distributed RL* [14]–[16]. A significant advantage of distributed RL is that computational complexity remains invariant to the number of subsystems as learning can be executed in parallel. However, guaranteeing the optimality and stability of the closed-loop system for any generic NDS with any arbitrary topology still remains an open question. One approach that has been proposed in the recent literature to make the problem tractable is to exploit the structure of the network or of the control objective. For example, [17], [18] have developed distributed RL for linear multi-agent systems when the agent dynamics are decoupled, assuming that the reward function depends on a global state that is accessible by every agent. The authors of [14], [19], [20] have extended this to cases when the agent dynamics are interconnected. The approach relies on an exponential decay property of networks such that the effects of local decisions act only locally. However, this approach would not be practical when the network of interest does not have the exponential decay property, e.g., for networks with short average distances.

In this paper we present an alternative approach for exploiting network structure for distributed RL, based on an algebraic property of networks that are composed of identical subsystems connected over a complete graph [21], [22]. We refer to this type of NDS as a *homogeneous* NDS. Our main contributions are as follows:

- Starting with the ideal case of a homogeneous NDS, we first develop a centralized RL algorithm that finds a distributed optimal controller. The optimality of the distributed controller relies on the fact that the overall Riccati equation can be separated into two independent Riccati equations - one for the intra-subsystem dynamics and another for the inter-subsystem dynamics. These two dynamics respectively obey the difference and the average of subsystem states. Therefore, by applying an existing RL algorithm such as Off-Policy Iteration (Off-PI) [23], which is a one-shot and quadratic-convergent algorithm, to the state and input data, we can find a distributed optimal controller. However, this algorithm by itself is not completely scalable because it requires average data across all subsystems.
- To solve the scalability issue, we augment our method with a consensus-based model-free distributed observer [24] to estimate the average state instead of the entire state. The distributive nature of the observer allows the subsystems to have their own individual learning algorithm that can learn the RL controller distributively as well, requiring state information from only their neighboring subsystems over any connected and undirected network graph. The convergence of the proposed scheme, along with the optimality and stability of the learned distributed controller, is theoretically guaranteed

The work by the first author was supported in part by JSPS KAKENHI JP22K14276 and JSPS KAKENHI(B) 20H02169.

T. Sadamoto and A. Kikuya are with Department of Mechanical and Intelligent Systems Engineering, Graduate School of Informatics and Engineering, The University of Electro-Communications; 1-5-1, Chofugaoka, Chofu, Tokyo 182-8585, Japan. Email: {sadamoto, a.kikuya}@uec.ac.jp.

A. Chakrabortty is with Electrical & Computer Engineering, North Carolina State University; Raleigh, North Carolina, USA, 27695. Email: achakra2@ncsu.edu.

due to the finite-time convergence of the distributed observer. The communication topology of the distributed observer is independent of that of the plant NDS. Sparser communication topologies, however, as will be shown later in simulations, require longer time of convergence for the learning phase.

• Finally, we consider the heterogeneous case, i.e., when the network is comprised of subsystems with different dynamic models, connected over any arbitrary network topology. Direct extension of our distributed RL design to such networks while guaranteeing stability and convergence, however, unfortunately is not possible. We, therefore, establish the applicability of the design to heterogeneous NDS using a robustness approach. We specificaly derive a numerical bound on how different the individual subsystems are allowed to be while the proposed algorithm can still guarantee stability and convergence. We validate our claims using numerical simulations.

To the best of our knowledge, this is the first distributed learning method that allows for designing the network structure among *learners*, and at the same time guarantees convergence to the optimal controller. Although this theoretical guarantee is limited to nearly-homogeneous NDS, the proposed method has a scale-free property. For linear NDSs consisting of κ subsystems each of whose dimension is n, the computational complexity of one of the computationally efficient RL algorithms [23] is of the order $(\kappa n)^6$ [25], whereas that of the proposed method is of the order n^6. Therefore, the proposed method is effective for NDSs consisting of a large number of (relatively small dimensional) subsystems. The most relevant existing works, as indicated above, are [14], [19], [20], which address weakly-coupled networks such that the effects of local decisions act only locally. In contrast, the proposed method targets strongly-coupled networks where local actions have global impacts, two leading examples being consensus networks [26] and power networks [7]. Integration of these two different methods is one of our future goals.

The rest of the paper is organized as follows. Section II describes the problem setting. Section III-A presents the centralized RL algorithm. A way of distributing the algorithm is described in Section III-B. Section IV shows a robustness of the proposed algorithm to general network systems where heterogeneous subsystems are interconnected via any graph structure. Section V shows the effectiveness of the proposed algorithm through simulations, and Section VI concludes this paper.

Notation: We denote the set of n-dimensional real vectors as $\mathbb{R}^n$, the set of integers as $\mathbb{Z}$, the positive (semi)definiteness of a symmetric matrix A by $A > 0$ ($A \geq 0$), the negative (semi)definiteness of A by $A < 0$ ($A \leq 0$), the n-dimensional column vector whose every entry is 1 by $\mathbf{1}_n$, the n-dimensional identity matrix as I_n, the set of $a_1, \dots a_n$ by $\{a_i\}_{i\in\{1,\dots n\}}$, the block-diagonal matrix having matrices $M_1, \cdots, M_n$ on its diagonal blocks by $\mathrm{diag}(M_1, \dots, M_n)$, and the finite-length sequence of $x(t)$ for $t \in [t_1, t_2]$ by $\{x(t)\}_{t_1 \leq t \leq t_2}$. The subscript n is omitted if obvious. Given A, let $\mathrm{sym}(A) := A + A^\mathsf{T}$. The operator $\otimes$ denotes the Kronecker product. For a matrix $P := [p_1, \dots, p_n]$, $\mathrm{vec}(P) := [p_1^\mathsf{T}, \dots, p_n^\mathsf{T}]^\mathsf{T}$. For $x \in \mathbb{R}^n$, the 2- and infinity-norm are denoted by $\|x\|_2$ and $\|x\|_\infty$. For functions $f(x), g(x)$, if there exist M and

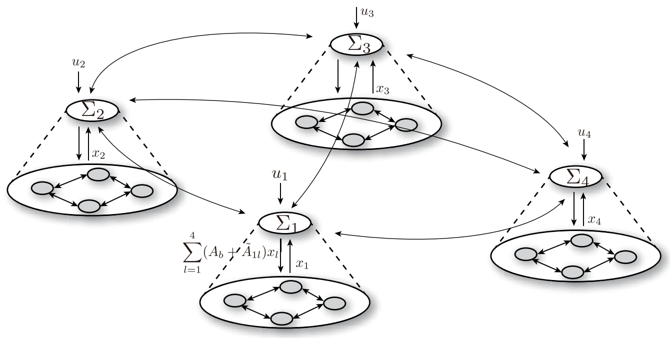

Fig. 1. An example of the interconnection of $\Sigma_1, \cdots, \Sigma_\kappa$ when $n = 4$ and $\kappa = 4$. For simplicity, we omit $\sum_{l=1}^{\kappa}(A_b + \tilde{A}_{kl})x_l$ for $l \in \{2, 3, 4\}$.

δ such that $\|x\| < \delta \Rightarrow \|f(x)\| \leq M\|g(x)\|$, then we say $f(x) = O(g(x))$. Given systems $\Sigma_1 : u \to y$ and $\Sigma_2 : y \to u$, we denote the closed-loop system of those as (Σ_1, Σ_2).

II. Problem Setup

A. Networked Dynamical Systems

Consider a network system where κ subsystems are coupled through a graph. For $k \in \{1, \dots, \kappa\}$, the k-th subsystem dynamics are described as

$$\Sigma_k : \dot{x}_k = (A_a + \tilde{A}_{kk})x_k + \sum_{l=1}^{\kappa}(A_b + \tilde{A}_{kl})x_l + (B_a + \tilde{B}_k)u_k \tag{1}$$

where $x_k \in \mathbb{R}^n$ is a state, $u_k \in \mathbb{R}^m$ is a control input. In (1), $\{A_a, A_b, B_a\}$ represents a nominal dynamical state-space model for the subsystems, while $\{\tilde{A}_{kk}, \tilde{A}_{kl}, \tilde{B}_k\}$ represents a perturbation from that model for each subsystem. Throughout the paper, we suppose that the perturbation is *small*, i.e., individual subsystems are nearly homogeneous, and the weights of interconnection are close to each other. Later in Section IV, we will discuss how the perturbation affects our learning algorithm.

Let $\tilde{A} \in \mathbb{R}^{\kappa n \times \kappa n}$ be such that the (k,l)-th n-by-n block matrix is $\tilde{A}_{kl}$ for $k \neq l$ while the (k,k)-th block matrix is $\sum_{l=1}^{\kappa} \tilde{A}_{kl}$, and $\tilde{B} := \mathrm{diag}(\tilde{B}_1, \dots, \tilde{B}_\kappa)$. Using the notation

$$\begin{aligned} A &:= (I \otimes A_a + (\mathbf{1}\mathbf{1}^\mathsf{T}) \otimes A_b) + \tilde{A}, \quad & B &:= (I \otimes B_a) + \tilde{B}, \\ x &:= [x_1^\mathsf{T}, \dots, x_\kappa^\mathsf{T}]^\mathsf{T}, & u &:= [u_1^\mathsf{T}, \dots, u_\kappa^\mathsf{T}]^\mathsf{T}, \end{aligned} \tag{2}$$

the interconnected network can be written as

$$\Sigma : \ \dot{x} = Ax + Bu. \tag{3}$$

Our objective is to develop a distributed learning algorithm for designing a distributed optimal controller for the system Σ in (3). We make the following assumptions.

Assumption 1: The matrices A_a, A_b, B_a, $\tilde{A}_{kl}$, and $\tilde{B}_k$ are unknown.

Assumption 2: The matrix A is Hurwitz.

Assumption 1 implies that although we know that the system of our interest is a network of κ subsystems coupled together via a graph we do not know its state-space model. Assumption 2 is usually satisfied in many real-world networks. Under these settings, we formulate a distributed learning problem in the next section.

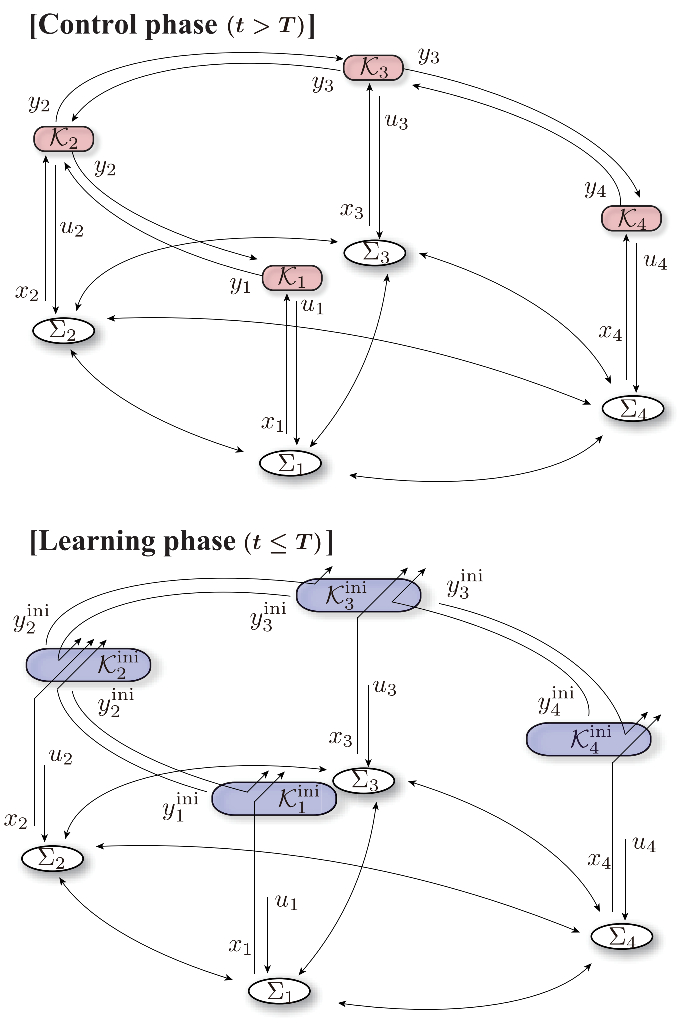
[Control phase ($t > T$)]
[Learning phase ($t \leq T$)]
$\mathcal{K}_1$, $\mathcal{K}_2$, $\mathcal{K}_3$, $\mathcal{K}_4$
Σ_1, Σ_2, Σ_3, Σ_4
y_1, y_2, y_3, y_4, u_1, u_2, u_3, u_4, x_1, x_2, x_3, x_4
$\mathcal{K}_1^{\rm ini}$, $\mathcal{K}_2^{\rm ini}$, $\mathcal{K}_3^{\rm ini}$, $\mathcal{K}_4^{\rm ini}$
$y_1^{\rm ini}$, $y_2^{\rm ini}$, $y_3^{\rm ini}$, $y_4^{\rm ini}$

Fig. 2. A schematic diagram of the problem for the case where $\boldsymbol{\kappa} = \mathbf{4}$, $\mathbb{L}_1 = \{1,2\}$, $\mathbb{L}_2 = \{1,2,3\}$, $\mathbb{L}_3 = \{2,3,4\}$ and $\mathbb{L}_4 = \{3,4\}$. In addition, $\mathcal{K}_k^{\rm ini}$ is defined in (9), and u_k is given in feedforward.

B. Problem Formulation

For $k \in \{1,\dots,\kappa\}$, let $\mathbb{L}_k \subseteq \{1,\dots,\kappa\}$ be a given index set of neighboring sub-controllers, such that the graph corresponding to $\{\mathbb{L}_k\}_{k\in\{1,\dots,\kappa\}}$ is connected and undirected. We impose this distribution structure to both the *learning phase* and *control phase*. In other words, each subsystem can only use its own and its neighbors' information for learning its sub-controller and for computing the input of the learned sub-controller. We suppose that both phases are performed within a single episode, i.e., the learning is performed for $t \in [0,T]$ with a given parameter T while the control is for $t > T$. A schematic picture of the two phases is shown in Fig. 2. Each phase is formulated as follows.

• *Control phase:* Let the k-th sub-controller be described as

$$\begin{bmatrix} u_k(t) \\ y_k(t) \end{bmatrix} = \mathcal{K}_k(x_k(t), \{y_j(t)\}_{j\in\mathbb{L}_k}), \quad t > T \tag{4}$$

where $y_k \in \mathbb{R}^r$ is the output used for communicating neighboring sub-controllers, x_k and u_k are defined in (1), and $\mathcal{K}_k$ is a dynamical map to be designed. The control objective of the set $\{\mathcal{K}_k\}_{k\in\{1,\dots,\kappa\}}$ is to minimize the cost function

$$J := \int_T^\infty x^\mathsf{T}(t)Qx(t) + u^\mathsf{T}(t)Ru(t)dt, \tag{5}$$

with

$$Q := I \otimes Q_a + (\mathbf{1}\mathbf{1}^\mathsf{T}) \otimes Q_b, \quad R := I \otimes R_a \tag{6}$$

where $Q_a \in \mathbb{R}^{n\times n}$, $Q_b \in \mathbb{R}^{n\times n}$, and $R_a \in \mathbb{R}^{m\times m}$ are given such that

$$Q_a \geq 0, \quad Q_a + \kappa Q_b \geq 0, \quad R_a > 0 \tag{7}$$

to ensure $Q \geq 0$ and $R > 0$. Using these structured Q and R, we can construct a distributed RL algorithm while appropriately controlling the behavior of the nearly-homogeneous network Σ.

• *Learning phase:* The requirements for designing $\mathcal{K}_k$ in (4) are twofolds: First, it should be data-driven due to the Assumption 1. The second requirement is that only neighboring information

$$\mathbb{D}_k := \{x_k(t), u_k(t), \{y_j^{\rm ini}(t)\}_{j\in\mathbb{L}_k}\}_{0\leq t\leq T} \tag{8}$$

should be available for the design, where $y_k^{\rm ini}$ is defined as

$$y_k^{\rm ini}(t) = \mathcal{K}_k^{\rm ini}(x_k(t), \{y_j^{\rm ini}(t)\}_{j\in\mathbb{L}_k}), \quad t \in [0,T] \tag{9}$$

for a given $\mathcal{K}_k^{\rm ini}$. Note that u_k is not generated by $\mathcal{K}_k^{\rm ini}$, but will be given in feedforward mode. In this setting, the controller design can be described as a map $\mathfrak{A}$ from the neighbors' data to the k-th sub-controller, i.e.,

$$\mathcal{K}_k = \mathfrak{A}(\mathbb{D}_k; Q, R), \quad k \in \{1,\dots,\kappa\} \tag{10}$$

where the weights Q and R are assumed to be shared among subsystems. In conclusion, our problem is summarized as follows.

Problem 1: Consider Σ in (3) and J in (5) with Q and R in (6), where Q_a, Q_b and R_a are given such that (7) holds. Let Assumptions 1-2 be held. Given $\{\mathbb{L}_k\}_{k\in\{1,\dots,\kappa\}}$, T and $\{u_k(t)\}_{k\in\{1,\dots,\kappa\}}$ for $t \in [0,T]$, find $\mathfrak{A}$ in (10) and $\{\mathcal{K}_k^{\rm ini}\}_{k\in\{1,\dots,\kappa\}}$ in (9) such that the resultant $\{\mathcal{K}_k\}_{k\in\{1,\dots,\kappa\}}$ makes J small as much as possible.

Owing to the distribution architecture of both learning and control phases as shown in Fig. 2, the complexity for computing $\mathcal{K}_k$ in (10) and $\{u_k, y_k\}$ in (4) is entirely independent from the number of subsystems. The main challenge of Problem 1 lies in the distributive nature of the learning phase.

III. Proposed Method

We first present a centralized RL algorithm to design a distributed optimal controller. Later, by distributing the algorithm, we show a solution to Problem 1. Throughout this section, for developing the algorithm, we impose the following assumption on Σ in addition to Assumptions 1-2.

Assumption 3: $\tilde{A} = 0$, $\tilde{B} = 0$ in (2).

Assumption 3 implies that the dynamics of individual subsystems are identical while their interconnections are symmetrically identical. In other words, under this assumption, Σ is a network of homogeneous subsystems coupled together via a complete graph. Later in Section IV, we will investigate the heterogeneous case.

A. Centralized RL Algorithm of Designing Distributed Optimal Controller

We start from the following lemma.

Lemma 1: Consider Σ in (3) with J in (5). Let $P_a > 0$ and $\bar{P} > 0$ be solutions of

$$A_a^\top P_a + P_a A_a - P_a B_a R_a^{-1} B_a^\top P_a + Q_a = 0 \quad (11)$$

$$\bar{A}^\top \bar{P} + \bar{P}\bar{A} - \bar{P} B_a R_a^{-1} B_a^\top \bar{P} + \bar{Q} = 0 \quad (12)$$

respectively, where

$$\bar{A} := A_a + \kappa A_b, \quad \bar{Q} := Q_a + \kappa Q_b. \quad (13)$$

Then the control law $u = -Kx$ with

$$K = I \otimes K_a + (\mathbf{1}\mathbf{1}^\top) \otimes \left(\frac{\bar{K} - K_a}{\kappa} \right) \quad (14)$$

where $K_a := R_a^{-1} B_a^\top P_a$ and $\bar{K} := R_a^{-1} B_a^\top \bar{P}$, minimizes J.

Proof: Note that the positive-definite solution of

$$\mathrm{Ric}: \ A^\top P + PA - PBR^{-1}B^\top P + Q = 0. \quad (15)$$

is unique because A is stable. We will show that the solution can be described as

$$P = I \otimes P_a + (\mathbf{1}\mathbf{1}^\top) \otimes P_b \quad (16)$$

where $P_b := (\bar{P} - P_a)/\kappa$. The positive-definiteness of P in (16) follows if $\bar{P} > 0$ and $P_a > 0$ because P is block-diagonally-dominant. The n-by-n block-diagonal and off-diagonal parts of Ric can be written as

$$\begin{aligned}&\mathrm{sym}((A_a + A_b)^\top (P_a + P_b) + (\kappa - 1) A_b^\top P_b) \\ &- (P_a + P_b) B_a R_a^{-1} B_a^\top (P_a + P_b) \\ &- (\kappa - 1) P_b B_a R_a^{-1} B_a^\top P_b + Q_a + Q_b = 0 \end{aligned} \quad (17)$$

and

$$\begin{aligned}&\mathrm{sym}((A_a + A_b)^\top P_b + A_b^\top (P_a + P_b) + (\kappa - 2) A_b^\top P_b \\ &-(P_a + P_b) B_a R_a^{-1} B_a^\top P_b) - (\kappa - 2) P_b B_a R_a^{-1} B_a^\top P_b + Q_b = 0\end{aligned} \quad (18)$$

respectively. We see that (17) is equivalent to the sum of (11) multiplied by $\kappa - 1$ and (12). Also, by subtracting (11) from (12), we have (18). Therefore, P in (16) is the unique solution of Ric. Using this P, the optimal gain PBR^{-1} can be written as (14), which completes the proof. ∎

This lemma means that the κn-by-κn Riccati equation Ric in (15) can be decomposed into two n-by-n Riccati equations (17)-(18), each of whose size is independent of the number of subsystems κ. In other words, constructing (14) via solving those two equations is scalable in κ. We note that the resultant controller $u = -Kx$ has a distributed structure that is compatible with the structure of Σ. To solve (17)-(18) in a model-free way, we define the arithmetic averages of x and u as

$$\bar{x} := \sum_{k=1}^{\kappa} \kappa^{-1} x_k \in \mathbb{R}^n, \quad \bar{u} := \sum_{k=1}^{\kappa} \kappa^{-1} u_k \in \mathbb{R}^m \quad (19)$$

and the differences from the averages as

$$\tilde{x}_k := x_k - \bar{x}, \quad \tilde{u}_k := u_k - \bar{u}, \quad k \in \{1, \dots, \kappa\}. \quad (20)$$

Then, taking the coordinate transformation of x and u to

$$\xi := [\bar{x}^\top, \tilde{x}_1^\top, \dots, \tilde{x}_{\kappa-1}^\top]^\top, \quad \mu := [\bar{u}^\top, \tilde{u}_1^\top, \dots, \tilde{u}_{\kappa-1}^\top]^\top,$$

the system Σ in (3) can be rewritten as

$$\dot{\xi} = \mathrm{diag}(\bar{A}, A_a, \dots, A_a)\xi + (I \otimes B_a)\mu. \quad (21)$$

Let us focus on the first n equations of (21). The Riccati equation of the dynamics with the cost $\bar{J} := \int_0^\infty \bar{x}^\top \bar{Q} \bar{x} + \bar{u}^\top R_a \bar{u} dt$ coincides with (12). Therefore, by applying any one-shot Q-learning algorithm such as Off-Policy Iteration (Off-PI) [23] to a finite-length data-set $\{\bar{x}, \bar{u}\}$ with $\bar{J}$, we can find $\bar{K}$ in (14) without knowing the system model. Similarly, for any choice of $k \in \{1, \dots, \kappa - 1\}$, we can find K_a by applying the same algorithm to the data of $\{\tilde{x}_k, \tilde{u}_k\}$ with the cost $J_a := \int_0^\infty \tilde{x}_k^\top Q_a \tilde{x}_k + \tilde{u}_k^\top R_a \tilde{u}_k dt$. Also, we can see that this result holds when $k = \kappa$ by redefining ξ and μ such that $\tilde{x}_\kappa$ and $\tilde{u}_\kappa$ are included. The brief summary of the design steps can be written as follows:

1. Collect a finite-length data-set $\{x, u\}$
2. Choose $k \in \{1, \dots, \kappa\}$. Compute $\{\bar{x}, \bar{u}\}$ and $\{\tilde{x}_k, \tilde{u}_k\}$ in (19)-(20).
3. Learn $\bar{K}$ and K_a by applying Off-PI to the data-sets with $\bar{J}$ and J_a, respectively.
4. Construct K in (14)

The details of the Off-PI and condition on data for guaranteeing convergence will be shown later in Section III-C. The above algorithm has a structure in which the information of all subsystems is sent to one place, and then the learned control gains are distributed to individual subsystems.

The above algorithm, however, is not completely scalable because it requires all-to-all communications among subsystems for computing $\bar{x}$ and $\bar{u}$ in (19). To overcome this difficulty, we aim to estimate the averages by a model-free consensus-type distributed observer in [24], which is described in the next subsection.

B. Average State Estimation

Unlike the original average-state observer in [24] estimating a scalar-valued average of multiple scalar valuables, we need to estimate the n-dimensional vector $\bar{x}$. Thus, for each n entry, we construct a distributed observer composed of κ sub-observers.

Let $x_k^i \in \mathbb{R}$ be the i-th entry of the k-th subsystem's state $x_k \in \mathbb{R}^n$ in (1). Our objective here is to estimate the average

$$\bar{x}^i := \sum_{k=1}^{\kappa} x_k^i / \kappa. \quad (22)$$

Let the graph structure among sub-observers be specified by $\{\mathbb{L}_k\}_{k \in \{1,\dots,\kappa\}}$ introduced in Section II-B. Also, let $L \in \mathbb{R}^{\kappa \times \kappa}$ be its graph-Laplacian matrix whose (k, j)-entry is defined as

$$L_{kj} := \begin{cases} -1, & \text{if } j \in \mathbb{L}_k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } j \neq k, \quad L_{kk} := -\sum_{j \neq k} L_{kj}. \quad (23)$$

Though the graph can be different for each i, for simplicity we suppose that the graphs of all distributed observers are

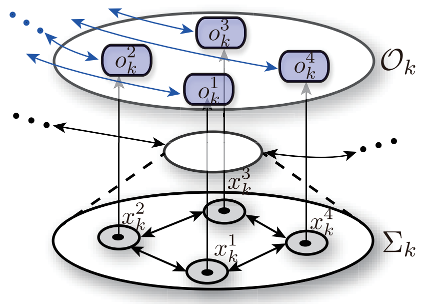

Fig. 3. An illustrative example of the sub-observer $\boldsymbol{\mathcal{O}_k}$ when $\boldsymbol{\Sigma_k}$ is 4-dimensional.

identical. Following [24], the i-th entry of the k-th sub-observer dynamics is described as

$$o_k^i : \begin{cases} \dot{q}_k^i &= -\gamma_k^i \mathrm{sgn}\left(\sum_{j=1}^{\kappa} L_{kj} p_j^i\right) \\ p_k^i &= x_k^i + \sum_{j\neq k} L_{kj} q_j^i \end{cases} \tag{24}$$

where $q_k^i \in \mathbb{R}$ is the state, $p_k^i \in \mathbb{R}$ is the output, γ_k^i is the observer gain. If γ_k^i satisfies

$$\gamma_k^i \geq \frac{\sqrt{n}\alpha^i}{\lambda_2(L)} + 1, \tag{25}$$

where $\alpha^i := \max\{|\dot{x}_1^i|, \dots, |\dot{x}_\kappa^i|\}$ and $\lambda_2(L)$ is the second smallest eigenvalue of L, then

$$p_1^i(t) \equiv \cdots \equiv p_\kappa^i(t) \equiv \bar{x}^i(t), \quad {}^\forall t \geq t_*^i \tag{26}$$

where

$$t_*^i = \sqrt{\sum_{k=1}^{\kappa} \left(p_k^i(0) - \bar{x}^i\right)^2} / \lambda_2(L). \tag{27}$$

The relation (26) implies that the outputs of all sub-observers for any t after the finite time t_*^i become identical to the true average; please see [24] for details of this proof.

Remark 1: The output p_j^i in reality contains chattering noise due to the sign function in (24). One way to mitigate the noise is to replace it by the sigmoid function

$$\mathrm{sig}(x;\sigma) := -1 + 2\left(1 + \exp(-\sigma x)\right)^{-1} \tag{28}$$

where σ is a sufficiently large parameter [24].

Finally, the set $\{o_k^i\}_{i\in\{1,\dots,n\}}$ can be described as

$$y_k = \mathcal{O}_k(x_k, \{y_j\}_{i\in\mathbb{L}_k}), \ \ y_k := \begin{bmatrix} p_k \\ q_k \end{bmatrix}, \ \begin{cases} p_k := [p_k^1, \dots, p_k^n]^\mathsf{T} \\ q_k := [q_k^1, \dots, q_k^n]^\mathsf{T}. \end{cases} \tag{29}$$

An illustrative example of $\mathcal{O}_k$ is shown in Fig. 3. The distributed observer $\{\mathcal{O}_k\}_{k\in\{1,\dots,\kappa\}}$ has a feature that

$$p_1(t) \equiv \cdots \equiv p_\kappa(t) \equiv \bar{x}(t), \quad {}^\forall t \geq t_* := \max\{t_*^1, \dots, t_*^n\}. \tag{30}$$

Remark 2: Although the exact computation of t_* in (30) is difficult because the true average $\bar{x}^i$ appears in (27), an upper bound of t_*, denoted as $\underline{T}$, can be estimated as follows. Let $q_k(0) = 0$ in (29). From a simple calculation, t_* is shown to satisfy $t_* \leq \max\{\|x^1(0)\|, \dots, \|x^n(0)\|\}/\lambda_2(L)$ where $x^i := [x_1^i, \dots, x_\kappa^i]^\mathsf{T}$. Even though $\|x^i(0)\|$ is still difficult to know, one would be possible to estimate its upper bound in advance. Assume $\|x^i(0)\| \leq \beta^i$ and β^i is a known parameter. Then we have

$$t_* \leq \max\{\beta^1, \dots, \beta^n\}/\lambda_2(L) =: \underline{T}. \tag{31}$$

Later in our simulations, we use $\underline{T}$ as the time of convergence.

Remark 3: The right-hand side of (24) changes significantly near the true value, even if the sgn function is replaced by the sigmoid function. Therefore, in numerical simulations, the step-size parameter of numerical integration methods to solve (24) should be small enough. Consequently, the computational complexity for numerical integration is often large; however, it is different from the computation cost for the learning algorithm.

Next, we show how to incorporate this distributed observer with the RL algorithm presented in the previous section.

C. Proposed Algorithm

This section presents a solution to Problem 1. By choosing

$$\mathcal{K}_k^{\mathrm{ini}} := \mathcal{O}_k, \quad y_k^{\mathrm{ini}} := y_k \tag{32}$$

where $\mathcal{O}_k$ and y_k are defined in (29), we can estimate $\bar{x}$ in a distributed manner, as shown in (30). Although we can estimate $\bar{u}$ by a similar procedure, for simplicity we assume that $\bar{u}$ is shared in advance among subsystems. This sharing can be done because u_k is a feedforward signal. From (30), the signal

$$\hat{x}_k(t) := x_k(t) - p_k(t), \tag{33}$$

coincides with $\tilde{x}_k(t)$ in (20) if $t \geq t_*$. This fact and (30) imply that $\{p_k, \hat{x}_k\}$, which is accessible in a distributed manner, is available as an alternative data for $\{\bar{x}, \hat{x}_k\}$ without errors. Therefore, replacing $\{\bar{x}, \tilde{x}_k\}$ by $\{p_k, \hat{x}_k\}$ in the centralized RL algorithm in Section III-A, we obtain a distributed RL algorithm, whose pseudo-code is summarized as follows.

Algorithm 1: Proposed Distributed Learning Algorithm

```
Let $Q_a, Q_b, R_a$ satisfy (7). Give $\{\mathbb{L}_k\}_{k\in\{1,\dots,\kappa\}}$ such that the graph is connected and undirected, $\{\beta^i\}_{i\in\{1,\dots,n\}}$ in Remark 2, and $\{u_k\}_{k\in\{1,\dots,\kappa\}}$. Compute $L$ in (23) and $\underline{T}$ in (31), and choose $T > \underline{T}$ and a natural number $N$. Share these information among subsystems in advance.
For $k \in \{1,\dots,\kappa\}$, do the following:
Initialization:
Give $\bar{u}$ and $\tilde{u}_k$ in (19)-(20). Construct $\mathcal{K}_k^{\mathrm{ini}}$ in (9) with (32) where $\gamma_k^i$ satisfies (25). Give $\delta \geq 0$ and $\{t_j\}_{j\in\{1,\dots,N\}}$ such that $T = t_N > \cdots > t_1 = \underline{T}$.
Data Collection:
  1. Start the estimation by $\mathcal{K}_k^{\mathrm{ini}}$ from $t = 0$.
  2. Collect data $\{x_k, u_k, \{y_j^{\mathrm{ini}}\}_{j\in\mathbb{L}_k}\}$ for $t \in [\underline{T}, T]$ and compute $p_k$ and $\hat{x}_k$ in (33).
Policy Improvement ($\mathfrak{A}$):
  3. Compute $\phi^\bullet$ for $\bullet \in \{p_k, \hat{x}_k\}$ and $\rho^{\bullet\circ}$ for $\{\bullet,\circ\} \in \{\{p_k,p_k\},\{p_k,\bar{u}\},\{\hat{x}_k,\hat{x}_k\},\{\hat{x}_k,\tilde{u}_k\}\}$ as the stacked versions of $\phi_j^\bullet$ and $\rho_j^{\bullet\circ}$ for $j \in \{1,\dots,N\}$ defined as
       $\phi_j^\bullet := (\bullet(t_j)\otimes\bullet(t_j) - \bullet(t_{j-1})\otimes\bullet(t_{j-1}))^\mathsf{T}$   (34)
       $\rho_j^{\bullet\circ} := \int_{t_{j-1}}^{t_j} (\bullet(t)\otimes\circ(t))^\mathsf{T} dt.$   (35)
```

4. Let $\bar{K}_k^{[0]} = 0$, $K_{a,k}^{[0]} = 0$ and $i \leftarrow 0$.
5. Find $\{\bar{P}_k^{[i]}, \bar{K}_k^{[i+1]}\}$ and $\{P_{a,k}^{[i]}, K_{a,k}^{[i+1]}\}$ minimizing

$$\left\| \bar{\Theta}_k^{[i]} \begin{bmatrix} \mathrm{vec}(\bar{P}_k^{[i]}) \\ \mathrm{vec}(\bar{K}_k^{[i+1]}) \end{bmatrix} + \rho^{p_k p_k} \mathrm{vec}(\bar{Q} + \bar{K}_k^{[i]\top} R_a \bar{K}_k^{[i]}) \right\| \tag{36}$$

$$\left\| \Theta_{a,k}^{[i]} \begin{bmatrix} \mathrm{vec}(P_{a,k}^{[i]}) \\ \mathrm{vec}(K_{a,k}^{[i+1]}) \end{bmatrix} + \rho^{\hat{x}_k \hat{x}_k} \mathrm{vec}(Q_a + K_{a,k}^{[i]\top} R_a K_{a,k}^{[i]}) \right\| \tag{37}$$

where

$$\bar{\Theta}_k^{[i]} := \left[\phi^{p_k} \quad -2\rho^{p_k \bar{u}}(I \otimes R_a) - 2\rho^{p_k p_k}(I \otimes \bar{K}_k^{[i]\top} R_a) \right]$$
$$\Theta_{a,k}^{[i]} := \left[\phi^{\hat{x}_k} \quad -2\rho^{\hat{x}_k \tilde{u}_k}(I \otimes R_a) - 2\rho^{\hat{x}_k \hat{x}_k}(I \otimes K_{a,k}^{[i]\top} R_a) \right].$$

6. Exit if $\|\bar{K}_k^{[i+1]} - \bar{K}_k^{[i]}\| \le \delta$ and $\|K_{a,k}^{[i+1]} - K_{a,k}^{[i]}\| \le \delta$, otherwise let $i \leftarrow i + 1$ and return to Step 5.
7. Construct

$$\mathcal{K}_k^{[i]} : \begin{cases} y_k &= \mathcal{O}_k(x_k, \{y_j\}_{i \in \mathbb{L}_k}) \\ u_k &= K_{a,k}^{[i]} x_k + (\bar{K}_k^{[i]} - K_{a,k}^{[i]}) c_k y_k \end{cases} \tag{38}$$

where $c_k := [I_n, 0]$. Return $\mathcal{K}_k^{[i]}$ as $\mathcal{K}_k$ in Problem 1.

In **Data Collection**, the learner collects $\mathbb{D}_k$ in (8) and computes $\{p_k, \hat{x}_k\}$ by communicating with only its neighbors. Clearly, **Policy Improvement** is the procedure $\mathfrak{A}$ in (10) because that outputs the controller $\mathcal{K}_k$ by processing the data based on $\mathbb{D}_k$. In **Policy Improvement**, steps 4-6 are equivalent to a data-driven implementation of Kleinman's algorithm [27], which is an iterative scheme for solving the Riccati equation of Σ. The reason why we construct $\mathcal{K}_k^{[i]}$ as (38) is as follows. Note that the control $u = -Kx$ with K in (14) can be rewritten as

$$u_k = -K_a x_k - \left(\bar{K} - K_a\right) \bar{x}. \tag{39}$$

If

$$\bar{K}_k^{[i]} = \bar{K}, \quad K_{a,k}^{[i]} = K_k, \tag{40}$$

we can replace $\bar{x}$ and $\{\bar{K}, K_a\}$ in (39) by p_k and $\{\bar{K}_k^{[i]}, K_{a,k}^{[i]}\}$ for $t \ge T$ while keeping the output u_k same. As a result, we obtain $\mathcal{K}_k^{[i]}$ in the form of (38). Next, we show that (40) is indeed satisfied when $i \to \infty$. In other words, we show the convergence of the algorithm and stability/optimality of the resultant closed-loop system.

Theorem 1: Consider Problem 1 under Assumptions 1-3, and consider **Algorithm 1**. If

$$\mathrm{rank}[\rho^{p_k p_k}, \rho^{p_k \bar{u}}] = \mathrm{rank}[\rho^{\hat{x}_k \hat{x}_k}, \rho^{\hat{x}_k \tilde{u}_k}] = \frac{n(n+1)}{2} + mn \tag{41}$$

then following statements are true:

i) The closed-loop $(\Sigma, \{\mathcal{K}_k^{[i]}\}_{k \in \{1,\ldots,\kappa\}})$ is globally asymptotically stable for any i.
ii) Let $\mathcal{K}_k^\star := \lim_{i \to \infty} \mathcal{K}_k^{[i]}$. The set $\{\mathcal{K}_k^\star\}_{k \in \{1,\ldots,\kappa\}}$ minimizes J in (5).

Proof: Under the condition (41), there exists a unique pair $\{\bar{P}_k^{[i]}, \bar{K}_k^{[i+1]}\}$ (resp. $\{P_{a,k}^{[i]}, K_{a,k}^{[i+1]}\}$) making the term (36) (resp. (37)) exactly zero [23]. Moreover, the solution satisfies

$$\mathrm{sym}(\bar{P}_k^{[i]} \bar{A}_k^{[i]}) = -(\bar{Q} + (\bar{K}_k^{[i]})^\top R_a \bar{K}_k^{[i]}), \ \bar{K}_k^{[i+1]} = R_a^{-1} B_a \bar{P}_k^{[i]}$$

and

$$\mathrm{sym}(P_{a,k}^{[i]} A_{a,k}^{[i]}) = -(Q_a + (K_{a,k}^{[i]})^\top R_a K_{a,k}^{[i]}), \ K_{a,k}^{[i+1]} = R_a^{-1} B_a P_{a,k}^{[i]}$$

respectively, where $\bar{A}_k^{[i]} := \bar{A} - B_a \bar{K}_k^{[i]}$ and $A_{a,k}^{[i]} := A_a - B_a K_{a,k}^{[i]}$. Thus, the solution is independent from k, i.e.,

$$P_{a,1}^{[i]} = \cdots = P_{a,\kappa}^{[i]} =: P_a^{[i]}, \qquad \bar{P}_1^{[i]} = \cdots = \bar{P}_\kappa^{[i]} =: \bar{P}^{[i]},$$
$$K_{a,1}^{[i]} = \cdots = K_{a,\kappa}^{[i]} =: K_a^{[i]}, \quad \bar{K}_1^{[i]} = \cdots = \bar{K}_\kappa^{[i]} =: \bar{K}^{[i]}.$$

Using these symbols we define

$$\begin{aligned} P^{[i]} &:= I \otimes P_a^{[i]} + (\mathbf{1}\mathbf{1}^\top) \otimes \left(\bar{P}^{[i]} - P_a^{[i]}\right)/\kappa, \\ K^{[i]} &:= I \otimes K_a^{[i]} + (\mathbf{1}\mathbf{1}^\top) \otimes \left(\bar{K}^{[i]} - K_a^{[i]}\right)/\kappa. \end{aligned} \tag{42}$$

From a procedure similar to the proof of Lemma 1, it is shown that $\{P^{[i]}, K^{[i]}, K^{[i+1]}\}$ satisfies

$$\mathrm{sym}(P^{[i]} A^{[i]}) = -(Q + (K^{[i]})^\top R K^{[i]}), \ K^{[i+1]} = R^{-1} B P^{[i]} \tag{43}$$

with $K^{[0]} = 0$, where $A^{[i]} := A - BK^{[i]}$. This is the i-th step of the Kleinman's algorithm [27] to Σ. Therefore, $A^{[i]}$ is stable for any i. Because the behavior of the closed-loop $(\Sigma, \{\mathcal{K}_k^{[i]}\}_{k \in \{1,\ldots,\kappa\}})$ coincides with $\dot{x} = A^{[i]} x$ for $t \ge t_*$, Property i) is proven. Following [27], Property ii) holds. This completes the proof. ■

It should be noted here that the proposed algorithm is *parallelly computable* among subsystems. In other words, its computational cost is scalable in κ, and yet another benefit is that the resultant fully distributed controller $\{\mathcal{K}_k\}_{k \in \{1,\ldots,\kappa\}}$ is shown to be optimal. Moreover, owing to the advantage of the Off-PI [23], the algorithm is *one-shot*, i.e., we do not need to re-collect data by using an updated control gain $K^{[i+1]}$.

Finally, we summarize the impact of the design parameters of **Algorithm 1** on the learning/control phases and a guideline for parameter selection as follows:

• Note that the Riccati equations (11)-(12) correspond to the behavior of the average $\bar{x}$ in (19) and $\tilde{x}_k$ in (20) for any k. Therefore, similarly to the standard optimal control, we can individually regulate $\{\tilde{x}_k\}_{k \in \{1,\ldots,\kappa\}}$ and $\bar{x}$ by tuning Q_a and Q_b with R_a while satisfying (7).
• The graph structure $\{\mathbb{L}_k\}_{k \in \{1,\ldots,\kappa\}}$ does not have any impact on the resultant control performance as long as the graph is strongly connected. However, as we can see in (27), the structure affects the time to start learning. When the graph is sparse (resp. dense), the second smallest eigenvalue of the associated Laplacian will be small (resp. large). As a result, t_* in (30) where t_*^i is defined in (27) becomes large (resp. small). Therefore, the sparser the communication graph is, the slower will be the time to learn. This implicit impact on control phase will be demonstrated in the later numerical simulations.
• The impact of choosing β^i in (31) only affects the time to start learning $\underline{T}$ used in Step 2. Note that β^i should satisfy $\|x^i(0)\| \le \beta^i$ as described in Remark 2. Therefore, one can choose a larger β^i depending on the confidence of the prior knowledge regarding $\|x^i(0)\|$. The tradeoff, however, is that $\underline{T}$ becomes slower in that case.
• The gain γ_k^i in (24) should satisfy only (25). Therefore, a guideline for choosing γ_k^i would be to assign it an adequately

high value. In practice, the larger the gain, the more significant the impact of chattering noise on the estimated average state. However, as shown in continuous-time numerical simulations in Section V-A, the choice of large γ_k^i has little impact on the learning results in continuous-time settings. For the case when **Algorithm 1** is time-discretized for implementation to digital computers, please see the next subsection.

• The choice of N, $\{t_j\}_{j\in\{1,\dots,N\}}$ and $\{u_k\}_{k\in\{1,\dots,N\}}$ used in Steps 2-3 does not have any impact on the learned controller as long as the condition (41) is satisfied. In **Algorithm 1**, the parameters are assumed to be given according to the setting of Problem 1. Determining these parameters, particularly for N, may appear to be difficult. However, they can be determined through a data-driven approach by incrementally repeating Steps 2-3 with increasing N until (41) is met, as it can only be verified using the estimated data.

• The parameter δ utilized in Step 6 should be chosen to be sufficiently small, such as 10^{-6}. As Kleinman's algorithm is known to exhibit quadratic convergence [28], selecting a sufficiently small δ will not significantly impact the iteration count for learning.

Remark 4: One can rewrite (36) (resp. (37)) as a linear matrix equation form because there exists a unique solution making the evaluation cost (36) (resp. (37)) exactly zero, as shown in the proof. However, the existence of such a solution is limited to the case for homogeneous networks. In the following section we consider applying the algorithm to general heterogeneous network systems, and therefore, we describe Step 6 as a minimization problem.

Remark 5: The number of data samples N to satisfy (41) is in order of n^2 while the computational cost for executing Step 6 is in order of n^6, where n is the dimension of each subsystem. Even if n is large, by preconditioning the data p_k and $\hat{x}_k$ based on the singular value decomposition, we can reduce the cost to the order of n^3; see [25] for more details.

D. Time-Discretization for Implementation

Algorithm 1 contains two continuous-time calculations, i.e., the continuous-time evolution of o_k^i in (24) used for $\mathcal{K}_k^{\rm ini}$ and $\mathcal{K}_k$ and the continuous-time integral (35). For implementing this algorithm in a digital computer it will be necessary to discretize. One approach is replacing o_k^i with the time-discretized sub-observer with the sampler and zero-order-hold (ZOH), described as

$$\mathsf{o}_k^i:\left\{\begin{array}{rcl}\mathsf{q}_k^i(\mathsf{t}+1)&=&\mathsf{q}_k^i(\mathsf{t})-\Delta\gamma_k^i\mathrm{sig}\left(\sum_{j=1}^{\kappa}L_{kj}\mathsf{p}_j^i(\mathsf{t});\sigma\right)\\ \mathsf{p}_k^i(\mathsf{t})&=&x_k^i(\mathsf{t}\Delta)+\sum_{j\neq k}L_{kj}\mathsf{q}_j^i(\mathsf{t})\\ p_k^i(t)&=&\mathsf{p}_k^i(\mathsf{t}),\quad\forall t\in[\mathsf{t}\Delta,(\mathsf{t}+1)\Delta)\end{array}\right. \tag{44}$$

where $\Delta>0$ is the sampling interval and $\mathsf{t}\in\mathbb{Z}$. Note that the sign function in (24) is replaced with the sigmoid function defined as (28). In addition, one may replace (35) by its trapezoidal approximant

$$\varrho_j^{\bullet\circ}:=\frac{\Delta}{2}\left(\bullet(\mathsf{t}\Delta)\otimes\circ(\mathsf{t}\Delta)+\bullet(\mathsf{t}\Delta+\Delta)\otimes\circ(\mathsf{t}\Delta+\Delta)\right)^{\mathsf{T}}. \tag{45}$$

When o_k^i and $\rho_j^{\bullet\circ}$ in **Algorithm 1** are replaced with o_k^i and $\varrho_j^{\bullet\circ}$, we refer to the revised algorithm as **Algorithm 1D**.

Due to the replacement of o_k^i by o_k^i, the estimated signal p_k^i will be chattering around the true average, which may result in failure of learning. One way to avoid this phenomenon is to choose an appropriately small γ_k^i by monitoring the level of chattering noise on q_k^i. One heuristic procedure of choosing γ_k^i is as follows: When the estimate p_k^i is around the true average, $\sum_{j=1}^{\kappa}L_{kj}\mathsf{p}_k^i\approx 0$ holds. Therefore, one can see from (44) that $\mathsf{q}_k^i(\mathsf{t}+1)-\mathsf{q}_k^i(\mathsf{t})$ is chattering depending on the value of γ_k^i, and vise versa. Based on this observation, we can tune γ_k^i such that the chattering noise of $\mathsf{q}_k^i(\mathsf{t}+1)-\mathsf{q}_k^i(\mathsf{t})$ is small.

The trapezoidal approximation (45) leads to a Δ-dependent error between $\rho_j^{\bullet\circ}$ and $\varrho_j^{\bullet\circ}$. As the error become smaller as Δ becomes smaller, we can expect that stability and sub-optimality by the learned controller will be guaranteed as long as Δ is sufficiently small. The effectiveness of the discrete-time algorithm **Algorithm 1D** as well as selecting appropriate value of γ_k^i will be verified through a numerical simulation in Section V-C. A rigorous robustness analysis necessitates the extension of our result to an entirely discrete-time version based on the discrete-time consensus observer [29] and the discrete-time RL method [30], which is out of the scope of this paper.

IV. Robustness Analysis for Heterogeneous NDSs

In this section, we consider a general NDSs without imposing Assumption 3 on Σ in (3). For the following arguments, we define

$$\boldsymbol{A}:=I\otimes A_a+(\mathbf{11})^{\mathsf{T}}\otimes A_b,\quad \boldsymbol{B}:=I\otimes B_a \tag{46}$$

and let a homogeneous network system be denoted as

$$\boldsymbol{\Sigma}:\ \dot{\boldsymbol{x}}=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}u \tag{47}$$

where $\boldsymbol{x}:=[\boldsymbol{x}_1^{\mathsf{T}},\dots,\boldsymbol{x}_\kappa^{\mathsf{T}}]^{\mathsf{T}}$. We assume Assumptions 1-2 and the following.

Assumption 4: The matrix $\boldsymbol{A}$ is Hurwitz.

Assumptions 2 and 4 imply that both of heterogeneous and homogeneous networks are stable. Let

$$\epsilon:=\max\{\|\tilde{A}\|,\|\tilde{B}\|\} \tag{48}$$

denote the degree of heterogeneity. If ϵ is small, one can expect that applying the proposed algorithm to the heterogeneous network Σ will yield a sub-optimal distributed controller. In the remainder of this section, we verify this expectation mathematically. To this end, we introduce the following lemma.

Lemma 2: Consider Σ in (3) and $\boldsymbol{\Sigma}$ in (47). Let Assumptions 1-2 and 4 be held. Define

$$\bar{\boldsymbol{x}}:=\sum_{k=1}^{\kappa}\kappa^{-1}\boldsymbol{x}_k,\quad \tilde{\boldsymbol{x}}_k:=\boldsymbol{x}_k-\bar{\boldsymbol{x}}. \tag{49}$$

Additionally, consider $\mathcal{O}_k$ in (29), and define $\hat{x}_k$ in (33). Then, it follows that

$$\circ(t)=\bullet(t)+O(\epsilon),\quad {}^\forall t\geq t_* \tag{50}$$

where $\{\circ,\bullet\}\in\{\{p_k,\bar{\boldsymbol{x}}\},\{\hat{x}_k,\tilde{\boldsymbol{x}}_k\}\}$, for any k.

Proof: From a simple calculation, we have

$$\begin{aligned} x(t) &= \int_0^t e^{(\boldsymbol{A}+\tilde{A})\tau}(\boldsymbol{B}+\tilde{B})u(t-\tau)d\tau \\ &= \int_0^t (I+\tilde{A}\tau+\cdots)e^{\boldsymbol{A}\tau}(\boldsymbol{B}+\tilde{B})u(t-\tau)d\tau \\ &= \boldsymbol{x}(t)+\int_0^t e^{\boldsymbol{A}\tau}\tilde{B}u(t-\tau)d\tau \\ &\quad + \tilde{A}\int_0^t \tau e^{\boldsymbol{A}\tau}(I+\tfrac{1}{2}\tilde{A}\tau+\cdots)Bu(t-\tau)d\tau. \end{aligned} \tag{51}$$

Due to Assumptions 2 and 4, the signals $x(t)$ and $\boldsymbol{x}(t)$ are bounded. Therefore, the second and third terms in the RHS of (51) are also bounded. Thus, from (48), we have $\boldsymbol{x}(t) = x(t)+O(\epsilon)$. Moreover, from the definition (49), we have (50) for $\{\circ,\bullet\} \in \{\{x,\boldsymbol{x}\},\{\bar{x},\bar{\boldsymbol{x}}\},\{\tilde{x}_k,\tilde{\boldsymbol{x}}_k\}\}$ and for any t, where $\bar{x}$ and $\tilde{x}_k$ are defined in (19)-(20). Finally, from (30), the claim follows. ■

This lemma characterizes the average estimate p_k (resp. the difference estimate $\hat{x}_k$) of the heterogeneous system Σ by the actual average $\bar{\boldsymbol{x}}$ (resp. the actual difference $\tilde{\boldsymbol{x}}$) of the homogeneous system $\boldsymbol{\Sigma}$ with the degree of heterogeneity ϵ. Since p_k and $\hat{x}_k$ are used for the k-th learner, the learning results will be characterized by heterogeneity. This can be summarized in the theorem below, where we denote the data matrices $\phi^\bullet$ and $\rho^{\circ\bullet}$ in (37) constructed from $\{\boldsymbol{x},u\}$ in the bold font, i.e., $\boldsymbol{\phi}^\bullet$ and $\boldsymbol{\rho}^{\circ\bullet}$, for clarifying the representation.

Theorem 2: Consider Σ in (3), $\boldsymbol{\Sigma}$ in (47), J in (5), and Proposed Algorithm. Assume Assumptions 1-4, (41), and

$$\mathrm{rank}[\boldsymbol{\rho}^{\bar{x}\bar{x}},\boldsymbol{\rho}^{\bar{x}\bar{u}}] = \mathrm{rank}[\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k},\boldsymbol{\rho}^{\tilde{x}_k\tilde{u}_k}] = \frac{n(n+1)}{2}+mn \tag{52}$$

where $\bar{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}$ are defined in (49). Consider $\boldsymbol{P}^{[i]}>0$ and $\boldsymbol{K}^{[i]}$ satisfying

$$\mathrm{sym}(\boldsymbol{P}^{[i]}\boldsymbol{A}^{[i]}) = -(Q+(\boldsymbol{K}^{[i]})^\top R\boldsymbol{K}^{[i]}),\ \boldsymbol{K}^{[i+1]} = R^{-1}\boldsymbol{B}\boldsymbol{P}^{[i]} \tag{53}$$

where, $\boldsymbol{A}^{[i]} := \boldsymbol{A}-\boldsymbol{B}\boldsymbol{K}^{[i]}$ and $\boldsymbol{K}^{[0]}=0$. Then, the followings hold.

i) The Lyapunov function candidate $V^{[i]}(x) := x^\top\boldsymbol{P}^{[i]}x$ satisfies

$$\dot{V}^{[i]} = x^\top\Big(-(\tilde{\boldsymbol{K}}^{[i]})^\top R\tilde{\boldsymbol{K}}^{[i]} - \boldsymbol{P}^{[i]}\boldsymbol{B}^\top R^{-1}\boldsymbol{B}\boldsymbol{P}^{[i]} - Q + O(\epsilon)\Big)x \tag{54}$$

where x follows $(\Sigma,\{\mathcal{K}_k^{[i]}\}_{k\in\{1,\dots,\kappa\}})$, and $\tilde{\boldsymbol{K}}^{[i]} := \boldsymbol{K}^{[i]}-\boldsymbol{K}^{[i+1]}$.

ii) Let $J^{[i]}$ be J when the controller $\{\mathcal{K}_k^{[i]}\}_{k\in\{1,\dots,\kappa\}}$ is used. If ϵ is sufficiently small such that $\dot{V}^{[i]}<0$ for any $x\neq 0$, then we have

$$J^{[i]} = \boldsymbol{J}^{[i]}+O(\epsilon), \quad \boldsymbol{J}^{[i]} := x(\underline{T})^\top\boldsymbol{P}^{[i]}x(\underline{T}). \tag{55}$$

Proof: First, we show

$$P_{a,k}^{[i]} = \boldsymbol{P}_{a,k}^{[i]}+O(\epsilon), \quad K_{a,k}^{[i+1]} = \boldsymbol{K}_{a,k}^{[i+1]}+O(\epsilon) \tag{56}$$

where $\{P_{a,k}^{[i]},K_{a,k}^{[i+1]}\}$ is a solution minimizing (37) while $\{\boldsymbol{P}_{a,k}^{[i]},\boldsymbol{K}_{a,k}^{[i+1]}\}$ is a solution satisfying

$$\boldsymbol{\Theta}_{a,k}^{[i]}\begin{bmatrix}\mathrm{vec}(\boldsymbol{P}_{a,k}^{[i]})\\ \mathrm{vec}(\boldsymbol{K}_{a,k}^{[i+1]})\end{bmatrix} = -\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}\mathrm{vec}(Q_a+\boldsymbol{K}_{a,k}^{[i]\top}R_a\boldsymbol{K}_{a,k}^{[i]}) \tag{57}$$

with $\boldsymbol{\Theta}_{a,k}^{[i]} = [\boldsymbol{\phi}^{\tilde{x}_k}\ -2\boldsymbol{\rho}^{\tilde{x}_k\tilde{u}_k}(I\otimes R_a) - 2\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}(I\otimes\boldsymbol{K}_{a,k}^{[i]\top}R_a)]$. Note here that (57) has a unique solution for any i because of (52). From Lemma 2, $\phi^{\hat{x}_k}$, $\rho^{\hat{x}_k\tilde{u}_k}$ and $\rho^{\hat{x}_k\hat{x}_k}$ satisfy (50) for $\{\circ,\bullet\} = \{\{\phi^{\hat{x}_k},\boldsymbol{\phi}^{\tilde{x}_k}\},\{\rho^{\hat{x}_k\tilde{u}_k},\boldsymbol{\rho}^{\tilde{x}_k\tilde{u}_k}\},\{\rho^{\hat{x}_k\hat{x}_k},\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}\}\}$. Therefore, for $i=0$, (37) can be written as

$$\left\|(\boldsymbol{\Theta}_{a,k}^{[0]}+O(\epsilon))\begin{bmatrix}\mathrm{vec}(P_{a,k}^{[0]})\\ \mathrm{vec}(K_{a,k}^{[1]})\end{bmatrix} + (\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}+O(\epsilon))\mathrm{vec}(Q_a)\right\|. \tag{58}$$

Note here that $K_{a,k}^{[0]}=0$ was used for deriving (58). The solution minimizing (58) can be written as

$$\begin{aligned}\begin{bmatrix}\mathrm{vec}(P_{a,k}^{[0]})\\ \mathrm{vec}(K_{a,k}^{[1]})\end{bmatrix} &= -\Big((\boldsymbol{\Theta}_{a,k}^{[0]}+O(\epsilon))^\top(\boldsymbol{\Theta}_{a,k}^{[0]}+O(\epsilon))\Big)^{-1} \\ &\quad\times(\boldsymbol{\Theta}_{a,k}^{[0]}+O(\epsilon))(\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}+O(\epsilon))\mathrm{vec}(Q_a).\end{aligned} \tag{59}$$

From a simple calculation, for any two matrices A and A_ϵ whose sizes are identical, we have

$$\big((A+A_\epsilon)^\top(A+A_\epsilon)\big)^{-1} = (A^\top A)^{-1}-(A^\top A)^{-1}(I+X_\epsilon)^{-1}X_\epsilon$$

where $X_\epsilon := A^\top A_\epsilon + A_\epsilon^\top A + A_\epsilon^\top A_\epsilon$. By applying this formula to (59), we have

$$\begin{bmatrix}\mathrm{vec}(P_{a,k}^{[0]})\\ \mathrm{vec}(K_{a,k}^{[1]})\end{bmatrix} = -(\boldsymbol{\Theta}_{a,k}^{[0]})^\dagger\boldsymbol{\rho}^{\tilde{x}_k\tilde{x}_k}\mathrm{vec}(Q_a)+O(\epsilon),$$

which implies that the pair $\{P_{a,k}^{[0]},K_{a,k}^{[1]}\}$ minimizing (58) is shown to satisfy (56) for $i=0$. By repeating the above procedure, we have (56) for any i. Similarly, we have

$$\bar{P}_k^{[i]} = \bar{\boldsymbol{P}}_k^{[i]}+O(\epsilon), \quad \bar{K}_k^{[i+1]} = \bar{\boldsymbol{K}}_k^{[i+1]}+O(\epsilon) \tag{60}$$

where $\{\bar{P}_k^{[i]},\bar{K}_k^{[i+1]}\}$ is a solution minimizing (36) while $\{\bar{\boldsymbol{P}}_k^{[i]},\bar{\boldsymbol{K}}_k^{[i+1]}\}$ is that when $\epsilon=0$.

We now show the claim i). We define

$$F^{[i]} = \mathrm{diag}(K_{a,1}^{[i]},\dots,K_{a,k}^{[i]}) + \begin{bmatrix}\frac{\bar{K}_1^{[i]}-K_{a,1}^{[i]}}{\kappa} & \cdots & \frac{\bar{K}_1^{[i]}-K_{a,1}^{[i]}}{\kappa}\\ \vdots & \ddots & \vdots\\ \frac{\bar{K}_\kappa^{[i]}-K_{a,\kappa}^{[i]}}{\kappa} & \cdots & \frac{\bar{K}_\kappa^{[i]}-K_{a,\kappa}^{[i]}}{\kappa}\end{bmatrix} \tag{61}$$

Let $\mathcal{K}_k^{[i]}$ be given by (38) with the replacement of $K_{a,k}$ and $\bar{K}_k$ by $K_{a,k}^{[i]}$ and $\bar{K}_k^{[i]}$. For $t\geq t_*$, it follows from (30) that $u=[u_1^\top,\dots,u_\kappa^\top]^\top$ where u_k is given by $\mathcal{K}_k^{[i+1]}$ coincides with $-F^{[i+1]}x$. Thus, for the following stability analysis, we focus on the closed-loop $\dot{x}=(A-BF^{[i+1]})x$. It follows from (56), (60) and the proof of Theorem 1 that $F^{[i]}=\boldsymbol{K}^{[i]}+O(\epsilon)$. Let $A_{F^{[i]}} := A-BF^{[i]}$. Note that

$$\begin{aligned}\boldsymbol{P}^{[i]}A_{F^{[i+1]}} &= \boldsymbol{P}^{[i]}(\boldsymbol{A}+\tilde{A}+(\boldsymbol{B}+\tilde{B})(\boldsymbol{K}^{[i+1]}+\tilde{K}^{[i+1]}))\\ &= \boldsymbol{P}^{[i]}(\boldsymbol{A}^{[i]}+\boldsymbol{B}\boldsymbol{K}^{[i]}-\boldsymbol{B}\boldsymbol{K}^{[i+1]}+O(\epsilon))\\ &= \boldsymbol{P}^{[i]}\boldsymbol{A}^{[i]}+(\boldsymbol{K}^{[i+1]})^\top R\boldsymbol{K}^{[i]}-(\boldsymbol{K}^{[i+1]})^\top R\boldsymbol{K}^{[i+1]}+O(\epsilon).\end{aligned}$$

By the technique so-called completing the square, we have

$$\begin{aligned}\mathrm{sym}(\boldsymbol{P}^{[i]}A_{F^{[i+1]}}) &= -(\boldsymbol{K}^{[i]}-\boldsymbol{K}^{[i+1]})^\top R(\boldsymbol{K}^{[i]}-\boldsymbol{K}^{[i+1]})\\ &\quad -\boldsymbol{P}^{[i]}\boldsymbol{B}R^{-1}\boldsymbol{B}^\top\boldsymbol{P}^{[i]}-Q+O(\epsilon).\end{aligned} \tag{62}$$

Thus, Property i) follows. Next, note that the cost J when Assumption 3 holds coincides with $\boldsymbol{J}^{[i]}$. Thus, Property ii) follows. This completes the proof. ■

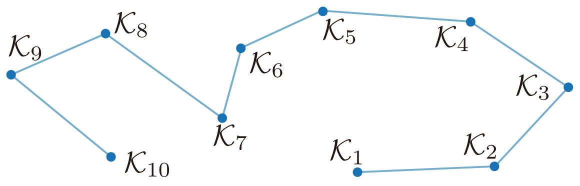

Fig. 4. The graph structure associated with $\boldsymbol{L}$

Property i) implies that the stability of the closed-loop system is guaranteed when ϵ is sufficiently small if $O(\epsilon)$ is negligible compared to the other terms in the RHS of (54). Property ii) guarantees sub-optimality achieved by the learned controller. As ϵ decreases, the resultant control performance of the heterogeneous network gets closer to that for homogeneous cases.

V. Numerical Simulation

A. Homogeneous Case

We first consider a homogeneous NDS composed of 10 subsystems, each of which is a three-dimensional single-input linear system. Thus, $\kappa = 10$, $n = 3$ and $m = 1$. The k-th subsystem dynamics is described as (1) where A_a, A_b and B_a are

$$A_a=\begin{bmatrix}0.5 & -6.0 & -3.5\\ -1.5 & -2.5 & 1\\ 6.0 & 1.5 & -2.5\end{bmatrix},\ A_b=\begin{bmatrix}-0.5 & -0.5 & 0\\ 0.5 & -2.0 & 0\\ 1 & 1.5 & 0.5\end{bmatrix},\ B_a=\begin{bmatrix}1\\0\\1\end{bmatrix},$$

$\tilde{B}_k = 0$ and $\tilde{A}_{kl} = 0$ for any l. Each entry of $x(0)$ is assumed to be randomly chosen from the range $[-0.5, 0.5]$. We construct Q and R from (6) where

$$Q_a=\begin{bmatrix}10 & 0 & 0\\ 0 & 10 & 0\\ 0 & 0 & 10\end{bmatrix},\ Q_b=\begin{bmatrix}2 & 0 & 0\\ 0 & 3 & 0\\ 0 & 0 & 2\end{bmatrix},\ R_a=1. \tag{63}$$

Note that $Q \geq 0$ and $R > 0$ hold. When the model information was available, we can construct the optimal control gain by (14) where

$$K_a=[1.9439, 3.3272, 3.4669],\ \bar{K}=[7.3815, 1.5936, 6.0814]. \tag{64}$$

However, because the model information is not available, we find the gain by applying **Algorithm 1** to this network.

Let the structure among sub-observers be a path graph, i.e.,

$$\mathbb{L}_1=\{1,2\},\ \mathbb{L}_\kappa=\{\kappa-1,\kappa\},\ \mathbb{L}_k=\{k-1,k,k+1\} \tag{65}$$

for $k \in \{2, \ldots, \kappa-1\}$. The graph structure is shown in Fig. 4. L is constructed using (23), yielding $\lambda_2(L) = 0.0979$. We use the exploration input as $u_k(t) = \sum_{h=1}^{100} \sin(\omega_{k,h}t)/10$ with $\omega_{k,h}$ randomly chosen from the range $(-50, 50)$ for $h \in \{1, \ldots, 100\}$. Also, let $\beta^i = 1$ in (31) for any i. Then, $\underline{T} = 10.22$. For $k \in \{1, \ldots, \kappa\}$, we construct a sub-observer $\mathcal{O}_k := \{o_k^i\}_{i\in\{1,\ldots,n\}}$ in (29), where o_k^i is given by (24) with the replacement of sgn by sig defined in (28) to mitigate chattering noise. Let $\gamma_k^i = 249$ and $\sigma = 10000$. We will later investigate how these values impact the learning results. Additionally, we let $\delta = 10^{-8}$, $N = 200$, $t_j := \underline{T} + (j-1)(T - \underline{T})/N$, and $T = 12.22$. From $t \geq 0$ we run $\mathcal{K}_k^{\rm ini} := \mathcal{O}_k$, and subsequently, for $t \in [\underline{T}, T]$, we collect $\{x_k, u_k, \{y_j^{\rm ini}\}_{j\in\mathbb{L}_k}\}$. In this case, the condition (41) is satisfied. The **Policy Improvement** in the algorithm was over at $i = 9$. All codes were developed in Matlab 2021b and run in an Intel(R) Core(TM) i9-9900K 3.60GHz, RAM 64.0GB computer. The total computational time for running all the iterations of **Algorithm 1** was found to be only 0.00014 (sec), which is almost negligible. The resultant control gains are

$$\begin{aligned}K_{a,1}^{[*]} \approx \cdots \approx K_{a,\kappa}^{[*]} &= [1.9421, 3.3016, 3.4634],\\ \bar{K}_1^{[*]} \approx \cdots \approx \bar{K}_\kappa^{[*]} &= [7.3810, 1.5962, 6.0813].\end{aligned}$$

Though differences among $K_{a,1}^{[*]} \cdots K_{a,\kappa}^{[*]}$ and $\bar{K}_1^{[*]} \cdots \bar{K}_\kappa^{[*]}$ exist due to the chattering noise of the estimation, those differences are almost negligible. By comparing the resultant gains with (64), we can see that the optimal controller is successfully obtained. In Fig. 5, the blue solid and yellow dotted lines depict $x_1(t) \in \mathbb{R}^3$ (i.e., the state of the first subsystem) when $\{\mathcal{K}_k\}_{k\in\{1,\ldots,\kappa\}}$ in (38) and the above model-based controller are actuated at $t = T$, respectively. For comparison, we show the case when no input is applied (i.e., $u(t) \equiv 0$ for $t \geq 0$) by the red chained lines. The following two observations can be made from this result. (i) The proposed method can learn the optimal controller in a decentralized manner. (ii) However, the performance improvement compared to the free-response is limited due to the long time required for learning.

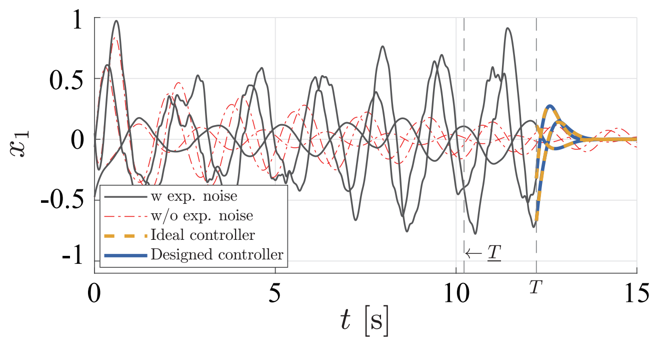

Fig. 5. Trajectory of $\boldsymbol{x_1}$ when no input is applied for $\boldsymbol{t \geq 0}$ (red chained), the exploration noise is applied for $\boldsymbol{t \in [0, \underline{T})}$ (black solid), the model-based optimal controller (yellow dotted) and the designed controller $\boldsymbol{\{\mathcal{K}_k\}_{k\in\{1,\ldots,\kappa\}}}$ are actuated at $\boldsymbol{t \geq T}$. Also, $\boldsymbol{\underline{T}}$ denotes the time to start learning.

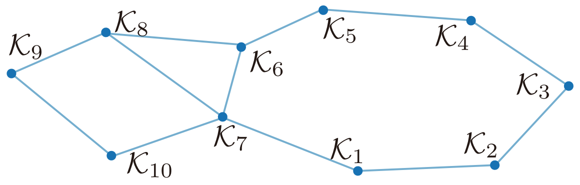

Fig. 6. The graph structure associated with $\boldsymbol{L'}$

For demonstrating the impact of choosing the graph structure on the control performance, we consider $\{\mathbb{L}'_k\}_{k\in\{1,\ldots,\kappa\}}$ whose graph Laplacian L' has the sparsity pattern as shown in Fig. 6. Because the graph is more dense than $\mathbb{L}_k$ given as (65), $\lambda_2(L')$ was smaller than $\lambda_2(L) = 0.0979$, i.e., $\lambda_2(L') = 0.4023$. As a result, the time to start learning becomes $\underline{T}' = 2.49$, which is faster than $\underline{T} = 10.22$. As a

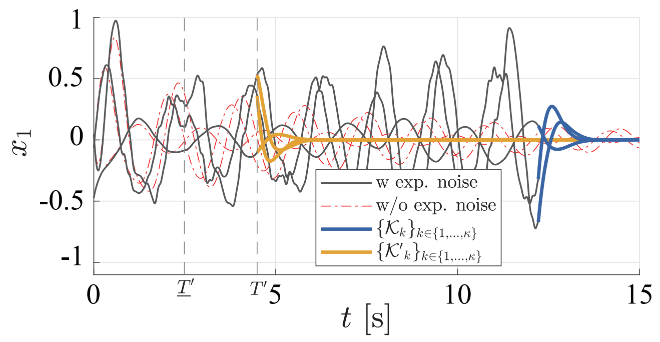

Fig. 7. Trajectory of x_1 of the homogeneous network when no input is applied for $t \geq 0$ (red chained), the exploration noise is applied for $t \in [0, T)$ (black solid), $\{\mathcal{K}'_k\}_{k\in\{1,\dots,\kappa\}}$ (yellow dotted) is actuated at $t = T'$, and $\{\mathcal{K}_k\}_{k\in\{1,\dots,\kappa\}}$ is actuated at $t = T$. Also, $\underline{T}'$ denotes the time to start learning $\{\mathcal{K}'_k\}_{k\in\{1,\dots,\kappa\}}$.

TABLE I

COST ACHIEVED BY THE RESULTANT CONTROLLER.

(γ_k^i, σ)	$(62, 10^3)$	$(62, 10^4)$	$(302, 10^4)$	$(1204, 10^4)$
$x^\top(0)Px(0)$	14.6286	14.9178	14.9227	14.9394

result, $T' = 4.49$. Let the newly learned controller be denoted as $\{\mathcal{K}'_k\}_{k\in\{1,\dots,\kappa\}}$. The closed-loop dynamic response with this controller is almost identical to that with $\{\mathcal{K}_k\}_{k\in\{1,\dots,\kappa\}}$ because of Theorem 1. However, owing to the fast learning, the performance achieved by $\{\mathcal{K}'_k\}_{k\in\{1,\dots,\kappa\}}$ becomes more damped than $\{\mathcal{K}_k\}_{k\in\{1,\dots,\kappa\}}$, as shown in Fig. 7 compared to Fig. 5. Therefore, we can see a trade-off between communication cost and the learning speed.

Next, we investigate how the choice of γ_k^i and σ affects the learning results. Let the graph structure be the one shown in Fig. 6. For simplicity we let γ_k^i is identical for any k and i. Table I shows the cost achieved by the resultant controller for four different choices of (γ_k^i, σ), respectively. We can see that the resultant cost $x^\top(0)Px(0)$ is almost same whereas γ_k^i and σ change. This result implies that the values of γ_k^i and σ have less impact on the learning. In this simulation, the RHS of (25) for $t \leq T$ was 61.1268. One may think that the model information is needed for choosing γ_k^i to satisfy (25); however, the fact that choosing γ_k^i as approximately twenty times the RHS value does not affect the results implies that such an appropriate choice is not necessary.

B. Heterogeneous Case

We next investigate the heterogeneous case. Let A_a, A_b and B_a be same as shown in the previous section while $A_{[kl]}$ and B_k are given such that every their element is randomly chosen from a range $\mathbb{D}$. We take $\mathbb{L}_k$ as the one corresponding to L' in Fig. 6, $R_a = 1$, $\gamma_k^i = 62$, $\sigma = 1000$. Let $Q = q_{\rm gain}(I \otimes Q_a + (\mathbf{1}\mathbf{1}^\top) \otimes Q_b)$ where Q_a and Q_b are given in (63). Let other parameters be identical in Section V-A. Table V-A shows whether the resultant closed-loop system is stable or not for several choices of $\mathbb{D}$ and $q_{\rm gain}$. Due to the reason as described in Remark 3, only in this simulation, we have utilized the true average $\bar{x}$ for learning. We can see that the closed-loop is stabilized when $q_{\rm gain}$ and the degree of heterogeneity are small. The closed-loop response of x_1 is shown in Fig. 8 for $q_{\rm gain} = 1$ and $\mathbb{D} = [-0.001, 0.001]$. The average-state estimation is used where the graph Laplacian is L'. In this case, the learned controller exhibits similarly to the ideal model-based controller. On the other hand, one can also see that the closed-loop is closer to unstable as $q_{\rm gain}$ or the degree of heterogeneity are larger. This is because the high control gains trigger higher levels of uncertainty for which the robustness guarantees of **Algorithm 1** can no longer catch up. Numerical computation of the allowable ranges for $q_{\rm gain}$ and the degree of heterogeneity as well as the interdependence between the two are open questions that will be explored in our future work.

TABLE II

THE STABILITY/INSTABILITY BY THE RESULTANT CONTROLLER, DEPENDING ON Q AND $\mathbb{D}$. THE SIGN ∘ (RESP. —) SHOWS THAT THE CORRESPONDING CLOSED-LOOP IS STABLE (RESP. UNSTABLE).

$\mathbb{D}$ \ $q_{\rm gain}$	0	0.01	0.1	1	10
$[-0.001, 0.001]$	∘	∘	∘	∘	—
$[-0.005, 0.005]$	∘	∘	∘	—	—
$[-0.01, 0.01]$	∘	∘	∘	—	—
$[-0.05, 0.05]$	∘	—	—	—	—
$[-0.1, 0.1]$	∘	—	—	—	—

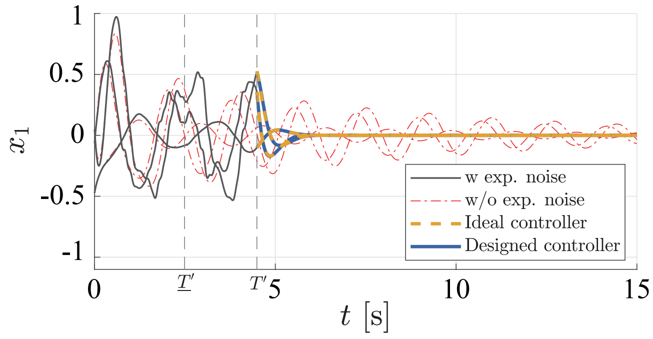

Fig. 8. Trajectory of x_1 of the heterogeneous network, where the notations are same as in Fig. 5.

C. Discrete-time Simulation for Heterogeneous Second-Order Oscillatory Networks

We show an effectiveness of the discrete-time version of the proposed algorithm **Algorithm 1D** in Section III-D for a larger complex heterogeneous NDS, where multiple second-order oscillators are interconnected via a complete undirected graph. Let $\kappa = 100$. For $k \in \{1, \dots, \kappa\}$, let the k-th oscillator dynamics be described as

$$\begin{bmatrix} \dot{\theta}_k \\ \dot{\omega}_k \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{f_k + \sum_j g_{kj}}{m_k} & -\frac{d_k}{m_k} \end{bmatrix} \begin{bmatrix} \theta_k \\ \omega_k \end{bmatrix} + \sum_{j=1}^{\kappa} \begin{bmatrix} 0 & 0 \\ \frac{g_{kj}}{m_j} & 0 \end{bmatrix} \begin{bmatrix} \theta_j \\ \omega_j \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_k} \end{bmatrix} u_k \tag{66}$$

where m_k and d_k are the inertia and the damping coefficient, g_{kj} is the stiffness of the connection between the k-th and j-th oscillators, and f_k is the proportional gain introduced for stabilizing $\bar{A}$ in (13). Suppose $g_{kj} = g_{jk}$. Let the values of m_k, d_k, and g_{kj} are randomly chosen from the ranges $[4.9, 5.1]$, $[1.4, 1.6]$, and $[0.099, 0.101]$, respectively. Let $\mathbb{L}_k$ be given as a strongly-connected Erdos-Renyi random graph with 300 edges. Fig. 9 shows the graph structure of a closed-loop system. Let $Q_a = 10$, $Q_b = 1$, $R_a = 1$, $\sigma = 10^6$,

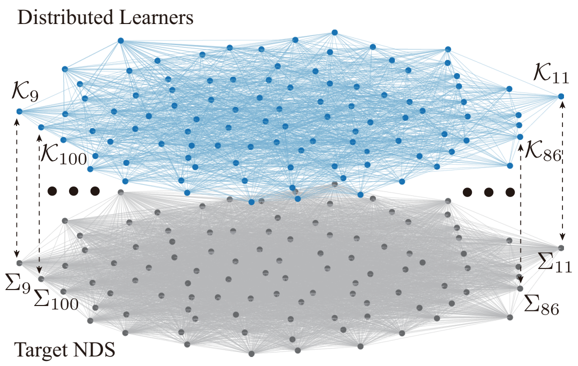

Fig. 9. The graph structure of the closed-loop system $(\boldsymbol{\Sigma}, \{\mathcal{K}_k\}_{k\in\{1,\ldots,100\}})$, where $\boldsymbol{\Sigma}$ consists of 100 oscillators.

TABLE III
THE ITERATION-WISE LEARNING TIME [SEC] OF **Cent-RL** AND **Algorithm 1D**. THE SIGN — SHOWS THAT THE TIME COULD NOT BE MEASURED DUE TO THE MASSIVE AMOUNT OF COMPUTATION.

κ	30	40	100
Cent-RL	58.1	283.1	–
Algorithm 1D	9.54×10^{-5}	7.41×10^{-5}	7.46×10^{-5}

the sampling interval of **Algorithm 1D** in Section III-D be $\Delta = 0.01$(msec), $\beta^i = 3$ in (31), and $N = 400$. The value of $\underline{T}$ defined as (31) becomes $\underline{T} = 0.16$, and the value of T is $T = 4.16$.

For selecting an appropriately small value of γ_k^i, we plot $\mathsf{p}_1^1(\mathsf{t})$ and $\mathsf{q}_1^1(\mathsf{t}+1) - \mathsf{q}_1^1(\mathsf{t})$ in Fig. 10 for two cases where $\gamma_k^i = 5$ and 2. For reference, we plot the true average $\bar{x}^1$ in Figs. 10(a-b). We can see from Figs. 10(a-b) that the estimate successfully track the true average for $t > \underline{T}$ in both cases. Furthermore, Fig. 10(c) shows that chattering noise occurs in $\mathsf{q}_1^1(\mathsf{t}+1)-\mathsf{q}_1^1(\mathsf{t})$ after $t > \underline{T}$ in both cases, and the magnitude of the noise when $\gamma_k^i = 2$ is smaller than when $\gamma_k^i = 5$. As we can expect that smaller magnitude of noise will yield better learning results, we choose $\gamma_k^i = 2$ for the following demonstration.

Let $\gamma_k^i = 2$. Fig. 11 shows the closed-loop response of ω_1 when **Algorithm 1D** is applied. This figure implies that the sub-optimal controller is successfully obtained by the discrete-time version of the proposed method even for larger and more complex NDS in Fig. 9.

Finally, we show an advantage of **Algorithm 1D** over a time-discretized version of the centralized RL algorithm in [23], which we refer to as **Cent-RL**. To compare the two algorithms, we consider three second-order oscillator networks each of which consists of 30, 40, and 100 oscillators. Table III shows the *iteration-wise learning time*, defined as the time required for updating the controller once. The learning time for **Cent-RL** when $\kappa = 100$ could not be measured due to the massive amount of computation. The computational complexity of **Cent-RL** is known to be of order $(\kappa n)^6$. Indeed, as shown in Table III, the learning time increases notably as the number of subsystems κ increases. In contrast, the learning time of the discrete-time algorithm **Algorithm 1D** remains constant as κ increases, implying the scalability of the proposed algorithm.

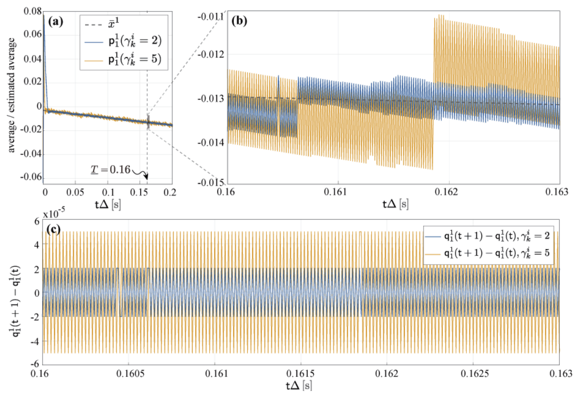

Fig. 10. Trajectories of $\mathbf{p}_1^1(\mathbf{t})$ and $\bar{\boldsymbol{x}}^1$ (a), its enlarged figure in the area $\mathbf{t}\boldsymbol{\Delta} \in [\mathbf{0.16}, \mathbf{0.163}]$ (b), and $\mathbf{q}_1^1(\mathbf{t}+1) - \mathbf{q}_1^1(\mathbf{t})$ in (44) (c) when $\gamma_k^i = \mathbf{5}, \mathbf{2}$ for any $\boldsymbol{k}$ and $\boldsymbol{i}$.

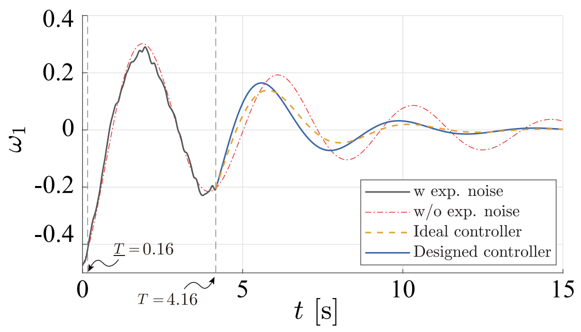

Fig. 11. Trajectory of $\boldsymbol{\omega_1}$ of the oscillatory network when $\boldsymbol{\kappa} = \mathbf{100}$.

VI. Conclusion

We have proposed a scalable algorithm for learning distributed optimal controllers for generic NDSs composed of nearly homogeneous subsystems under information constraints that follow a given arbitrary graph structure. Each sub-controller consists of an observer, which estimates the mean and differential states in a distributed manner, and the static controllers that feedback those. We have theoretically shown the convergence of the proposed algorithm and the optimality of the learned controller for the case where individual subsystems are identical. Furthermore, we have demonstrated the robustness of that analysis to general heterogeneous networks. The effectiveness of the proposed algorithm has been verified through numerical simulations. An interesting observation is that when the proposed algorithm is executed in discrete-time with a large sampling time (in the order of $\Delta = 1$ msec), it could not learn a stabilizing controller. Resolving this numerical issue would require a rigorous extension of this result by incorporating the discrete-time consensus observer [29] and the discrete-time RL method [30], which will be addressed in our future work.

References

[1] J. Shamma, *Cooperative control of distributed multi-agent systems*. John Wiley & Sons, 2008.
[2] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.
[3] L. Buşoniu, R. Babuška, and B. D. Schutter, "Multi-agent reinforcement learning: An overview," *Innovations in multi-agent systems and applications-1*, pp. 183–221, 2010.

[4] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks*. Princeton University Press, 2009.

[5] R. Hegselmann and U. Krause, "Opinion dynamics and bounded confidence models, analysis, and simulation," *Journal of artificial societies and social simulation*, vol. 5, no. 3, 2002.

[6] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.

[7] M. Yazdanian and A. Mehrizi-Sani, "Distributed control techniques in microgrids," *IEEE Transactions on Smart Grid*, vol. 5, no. 6, pp. 2901–2909, 2014.

[8] C. De Persis and P. Tesi, "Formulas for data-driven control: Stabilization, optimality, and robustness," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 909–924, 2019.

[9] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[10] Z. Qin, W. Li, and F. Janoos, "Sparse reinforcement learning via convex optimization," in *International Conference on Machine Learning*. PMLR, 2014, pp. 424–432.

[11] A. F. Dizche, A. Chakrabortty, and A. Duel-Hallen, "Sparse wide-area control of power systems using data-driven reinforcement learning," in *Proc. of American Control Conference*, 2019, pp. 2867–2872.

[12] T. Hoshiya and T. Sadamoto, "Fast online reinforcement learning of distributed optimal controller for large-scale network systems," in *Proc. of Conference on Control Technology and Applications*, 2021, pp. 1135–1141.

[13] C. Langbort and J. C. Delvenne, "Distributed design methods for linear quadratic control and their limitations," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2085–2093, 2010.

[14] Y. Lin, G. Qu, L. Huang, and A. Wierman, "Distributed reinforcement learning in multi-agent networked systems," *arXiv*, 2020.

[15] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *Acm computing surveys (csur)*, vol. 53, no. 2, pp. 1–33, 2020.

[16] G. Weiß, "Distributed reinforcement learning," in *The Biology and technology of intelligent autonomous agents*. Springer, 1995, pp. 415–428.

[17] G. Jing, H. Bai, J. George, and A. Chakrabortty, "Decomposability and parallel computation of multi-agent LQR," in *Proc. of American Control Conference*, 2021, pp. 4527–4532.

[18] H. Bai, J. George, and A. Chakrabortty, "Hierarchical control of multi-agent systems using online reinforcement learning," in *Proc. of American Control Conference*, 2020, pp. 340–345.

[19] G. Qu and N. Li, "Exploiting fast decaying and locality in multi-agent MDP with tree dependence structure," in *Proc. of Conference on Decision and Control*, 2019, pp. 6479–6486.

[20] G. Qu, A. Wierman, and N. Li, "Scalable reinforcement learning of localized policies for multi-agent networked systems," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 256–266.

[21] D. Tsubakino, T. Yoshioka, and S. Hara, "An algebraic approach to hierarchical LQR synthesis for large-scale dynamical systems," in *Proc. of Asian Control Conference*, 2013, pp. 1–6.

[22] M. K. Sundareshan and R. M. Elbanna, "Qualitative analysis and decentralized controller synthesis for a class of large-scale systems with symmetrically interconnected subsystems," *Automatica*, vol. 27, no. 2, pp. 383–388, 1991.

[23] Y. Jiang and Z. Jiang, *Robust adaptive dynamic programming*. Wiley IEEE Press, 2017.

[24] J. George, R. A. Freeman, and K. M. Lynch, "Robust dynamic average consensus algorithm for signals with bounded derivatives," in *Proc. of American Control Conference*, 2017, pp. 352–357.

[25] T. Sadamoto, A. Chakrabortty, and J. Imura, "Fast online reinforcement learning control using state-space dimensionality reduction," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 1, pp. 342–353, 2020.

[26] D. Zelazo and M. Bürger, "On the robustness of uncertain consensus networks," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 170–178, 2015.

[27] P. Dorato and A. Levis, "Optimal linear regulators: The discrete-time case," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 613–620, 1971.

[28] D. Kleinman, "On an iterative technique for riccati equation computations," *IEEE Transactions on Automatic Control*, vol. 13, no. 1, pp. 114–115, 1968.

[29] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch, and S. Martinez, "Tutorial on dynamic average consensus: The problem, its applications, and the algorithms," *IEEE Control Systems Magazine*, vol. 39, no. 3, pp. 40–72, 2019.

[30] T. Sadamoto and A. Chakrabortty, "Fast real-time reinforcement learning for partially-observable large-scale systems," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 3, pp. 206–218, 2020.

Tomonori Sadamoto (M'15) received the Ph.D. degree from the Tokyo Institute of Technology in 2015. From 2015 to 2016, he was a visiting researcher at School of Electrical Engineering, Royal Institute of Technology, Stockholm, Sweden. From 2016 to November 2018, he was a specially appointed assistant professor with the Department of Systems and Control Engineering, Graduate School of Engineering, Tokyo Institute of Technology. Since November 2018, hehas been assistant professor with Department of Mechanical and Intelligent Systems Engineering in the University of Electro-Communications. In 2014, he was named as a finalist of the 13st European Control Conference Best Student-Paper Award. In 2020, he received IEEE Control Systems Magazine Outstanding Paper Award.

Ayafumi Kikuya received the bachelor degree from School of Informatics and Engineering, The University of Electro-Communications in 2022. His research interest includes the interdisciplinary research among machine learning and control theory.

Aranya Chakrabortty (SM'15) received the Ph.D. degree in Electrical Engineering from Rensselaer Polytechnic Institute, NY in 2008. From 2008 to 2009 he was a postdoctoral research associate at University of Washington, Seattle, WA. From 2009 to 2010 he was an assistant professor at Texas Tech University, Lubbock, TX. Since 2010 he has joined the Electrical and Computer Engineering department at North Carolina State University, Raleigh, NC, where he is currently a Professor. His research interests are in all branches of control theory with applications to electric power systems. He was an associate editor for IEEE transactions on Control System Technology from 2016 to 2020, and currently serves as an editor for IEEE Transactions on Power Systems. He received the NSF CAREER award in 2011. He was named as a University Faculty Scholar by the NC State Provost's office in 2019.