Asynchronous Distributed Reinforcement Learning for LQR Control via Zeroth-Order Block Coordinate Descent

Gangshan Jing, He Bai, Jemin George, Aranya Chakrabortty and Piyush K. Sharma

Abstract

Recently introduced distributed zeroth-order optimization (ZOO) algorithms have shown their utility in distributed reinforcement learning (RL). Unfortunately, in the gradient estimation process, almost all of them require random samples with the same dimension as the global variable and/or require evaluation of the global cost function, which may induce high estimation variance for large-scale networks. In this paper, we propose a novel distributed zeroth-order algorithm by leveraging the network structure inherent in the optimization objective, which allows each agent to estimate its local gradient by local cost evaluation independently, without use of any consensus protocol. The proposed algorithm exhibits an asynchronous update scheme, and is designed for stochastic non-convex optimization with a possibly non-convex feasible domain based on the block coordinate descent method. The algorithm is later employed as a distributed model-free RL algorithm for distributed linear quadratic regulator design, where a learning graph is designed to describe the required interaction relationship among agents in distributed learning. We provide an empirical validation of the proposed algorithm to benchmark its performance on convergence rate and variance against a centralized ZOO algorithm.

Index Terms

Reinforcement learning, distributed learning, zeroth-order optimization, linear quadratic regulator, multi-agent systems

I. INTRODUCTION

Zeroth-order optimization (ZOO) algorithms solve optimization problems with implicit function information by estimating gradients via zeroth-order observation (function evaluations) at judiciously chosen points [1]. They have been extensively employed as model-free reinforcement learning (RL) algorithms for black-box off-line optimization [2], online optimization [3], neural networks training [4], and model-free linear quadratic regulator (LQR) problems [5]–[8]. Although, ZOO algorithms are applicable to a broad range of problems, they suffer from high variance and slow convergence rate especially when the problem is of a large size. To improve the performance of ZOO algorithms, besides the classical one-point feedback [3], [6], other types of estimation schemes have been proposed, e.g., average of multiple one-point feedback [5], two-point feedback [2], [4], [6], [9], [10], average of multiple two-point feedback [8], and one-point residual feedback [11]. These methods have been shown to reduce variance and increase convergence rate efficiently. However, all these methods are still based on the evaluation of the global optimization objective, which may cause significantly large errors and high variance of gradient estimates when encountering a large-size problem with a high dimensional variable.

Multi-agent networks are one of the most representative systems that have broad applications and usually induce large-size optimization problems [12]. In recent years, distributed zeroth-order convex and non-convex optimizations on multi-agent networks have been extensively studied, e.g., [13]–[17], all of which decompose the original cost function into multiple functions and assign them to the agents. Unfortunately, the variable dimension for each agent is the same as that for the original problem. As a result, the agents are expected to eventually reach consensus on the optimization variables. Such a setting indeed divides the global cost into local costs, but the high variance of the gradient estimate caused by the high variable dimension remains unchanged. In [18], each agent has a lower dimensional variable. However, to estimate the local gradient, the authors employed a consensus protocol for each agent to estimate the global cost function value. Such a framework maintains a similar gradient estimate to that of the centralized ZOO algorithm, which still suffers a high variance when the network is of a large scale.

Block coordinate descent (BCD) are known to be efficient for large-size optimization problems due to the low per-iteration cost [19], [20]. The underlying idea is to solve optimizations by updating only partial components (one block) of the entire variable in each iteration, [19]–[24]. Recently, zeroth-order BCD algorithms have been proposed to craft adversarial attacks for neural networks [4], [25], where the convergence analysis was given under the convexity assumption on the objective function in [25]. A non-convex optimization was addressed by a zeroth-order BCD algorithm in [26], whereas the feasible domain was assumed to be convex.

In this paper, we consider a general stochastic locally smooth non-convex optimization with a possibly non-convex feasible domain, and propose a distributed ZOO algorithm based on the BCD method. Out of the aforementioned situations, we design local cost functions involving only partial agents by utilizing the network structure embedded in the optimization objective over a multi-agent network. This is reasonable in model-free control problems because although the dynamics of each agent

G. Jing is with Chongqing University, Chongqing, 400044, PRC. jinggangshan@cqu.edu.cn

A. Chkarabortty is with North Carolina State University, Raleigh, NC 27695, USA. achakra2@ncsu.edu

H. Bai is with Oklahoma State University, Stillwater, OK 74078, USA. he.bai@okstate.edu

J. George and P. Sharma are with the DEVCOM Army Research Laboratory, Adelphi, MD 20783, USA. {jemin.george.civ,piyush.k.sharma.civ}@army.mil

is unknown, the coupling structure between different agents may be known and the control objective with an inherent network structure is usually artificially designed. Our algorithm allows each agent to update its optimization variable by evaluating a local cost independently, without requiring agents to reach consensus on their optimization variables. This formulation is applicable to the distributed LQR control and many multi-agent cooperative control problems such as formation control [27], network localization [28] and cooperative transportation [29], where the variable of each agent converges to a desired local set corresponding to a part of the minimizer of the global cost function. Our main contributions are listed as follows.

(i). We propose a distributed accelerated zeroth-order algorithm with asynchronous sample and update schemes (Algorithm 2). The multi-agent system is divided into multiple clusters, where agents in different clusters evaluate their local costs asynchronously, and the evaluations for different agents are completely independent. It is shown that with appropriate parameters, the algorithm converges to an approximate stationary point of the optimization with high probability. The sample complexity is polynomial with the reciprocal of the convergence error, and is dominated by the number of clusters (see Theorem 1). When compared to ZOO based on the global cost evaluation, our algorithm produces gradient estimates with lower variance and thus results in faster convergence (see the variance analysis in Subsection III-E).

(ii). We further consider a model-free multi-agent distributed LQR problem, where multiple agents with decoupled dynamics[1] cooperatively minimize a cost function involving all the agents. The optimal LQR controller is desired to be distributed, i.e., the control of each agent involves only its neighbors. We describe the couplings in the cost function and the structural constraint by a *cost graph* and a *sensing graph*, respectively. To achieve distributed learning, we design a local cost function for each agent, and introduce a *learning graph* that describes the required agent-to-agent interaction relationship for local cost evaluations. Specifically, the local cost function for each agent is designed such that its gradient w.r.t. the local control gain is the same as that of the global cost function. The learning graph is determined by the cost and sensing graphs and is needed only during the learning stage. By implementing our asynchronous RL algorithm, the agents optimize their local costs and are able to learn a distributed controller corresponding to a stationary point of the optimization distributively. The design of local costs and the learning graph plays the key role in enabling distributed learning. The learning graph is typically denser than the cost and sensing graphs, which is a trade-off for not using a consensus algorithm.

The comparisons between our work and existing related results are listed as follows.

(i). In comparison to centralized ZOO [2]–[4], [9], [11] and distributed ZOO [13]–[17], we reduce the variable dimension for each agent, and construct local cost involving only partial agents. Such a framework avoids the influence of the convergence error and convergence rate of the consensus algorithm, and results in reduced variance and high scalability to large-scale networks. Moreover, since the agents do not need to reach agreement on any global index, our algorithm benefits for privacy preserving.

(ii). In comparison to ZOO algorithms for LQR, e.g., [5], [6], [8], our problem has a structural constraint on the desired control gain, thus the cost function is not gradient dominated.

(iii) Existing BCD algorithms in [19]–[24] always require global smoothness of the objective function. When only zeroth-order information is available, existing BCD algorithms require convexity of either the objective function [25] or the feasible domain [26]. In contrast, we propose a zeroth-order BCD algorithm for a stochastic non-convex optimization with a locally smooth objective and a possibly non-convex feasible set. A clustering strategy and a cluster-wise update scheme are proposed as well, which further significantly improve the algorithm convergence rate.

(iv) The distributed LQR problem is complicated and challenging because both the objective function and the feasible set are non-convex, and the number of locally optimal solutions grow exponentially in system dimension [30], [31]. Existing results for distributed LQR include model-based centralized approaches [32], [33], model-free centralized approaches [7], [34], and model-free distributed approaches [18], [35]. All of them focus on finding a sub-optimal distributed controller for an approximate problem or converging to a stationary point of the optimization via policy gradient. Therefore, the obtained solutions may be a sub-optimal solution to the distributed LQR problem. Our algorithm is a derivative-free distributed policy gradient algorithm, which also seeks a stationary point of the problem and yields a sub-optimal solution.

(v). The sample complexity of our algorithm is higher than that in [5], [6], [8] because their optimization problems have the gradient domination property, and [5], [6] assume that the infinite horizon LQR cost can be obtained perfectly. The sample complexity of our algorithm for LQR is slightly lower than that in [18]. However, two consensus algorithms are employed in [18] for distributed sampling and global cost estimation, respectively, which may slow down the gradient estimation process especially for large-scale problems. Moreover, the gradient estimation in [18] is essentially based on global cost evaluation, while we adopt local cost evaluation, which improves the scalability to large-scale networks significantly.

(vi). Distributed RL has also been extensively studied via a Markov decision process (MDP) formulation, e.g., [36]–[40]. However, in comparison to our work, they require more information for each agent or are essentially based on global cost evaluation. More specifically, in [36], [38]–[40], the global state is assumed to be available for all the agents. It has been shown in [38] that naively applying policy gradient methods to multi-agent settings exhibits high variance gradient estimates. [37] describes a distributed learning framework with partial observations, but only from an empirical perspective.

---

[1]Our algorithm applies to multi-agent systems with coupled dynamics. Please refer to Remark 4 for more details.

This paper is structured as follows. Section II describes the optimization problem. Section III presents our zeroth-order BCD algorithm with convergence analysis. Section IV introduces the application of our algorithm to the model-free multi-agent LQR problem. Section V shows asimulation example. Section VI concludes the whole paper.

Notations. Given function $f(x)$ with $x = (x_1^\top, ..., x_N^\top)^\top$, denote $f(y_i, x_{-i}) = f(x')$ with $x' = (x_1'^\top, ..., x_N'^\top)^\top$, $x_i' = y_i$ and $x_j' = x_j$ for all $j \neq i$. Let $\mathbb{R}^n$ and $\mathbb{R}_{\geq 0}$ denote the $n-$dimensional Euclidean space and the space of nonnegative real numbers. The norm $\|\cdot\|$ denotes the $l2$-norm for vectors, and denotes the spectral norm for matrices, $\|\cdot\|_F$ is the Frobenius norm. The pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes a directed or an undirected graph, where $\mathcal{V}$ is the set of vertices, $\mathcal{E} \subset \mathcal{V}^2$ is the set of edges. A pair $(i, j) \in \mathcal{E}$ implies that there exists an edge from i to j. A path from vertex i to vertex j in $\mathcal{G}$ is a sequence of pairs (i, i_1), (i_1, i_2), ..., $(i_s, j) \in \mathcal{E}$. The symbol $\mathbb{E}[\cdot]$ denotes expectation, $\mathrm{Cov}(x)$ denotes the covariance matrix of vector x. Given a set S, $\mathrm{Uni}(S)$ represents the uniform distribution in S. Given two matrices X and Y, let $\langle X, Y \rangle = \mathrm{trace}(X^\top Y)$ be the Frobenius inner product.

II. STOCHASTIC ZOO VIA MULTI-AGENT NETWORKS

In this section, we review the zeroth-order stochastic non-convex optimization problem and describe the optimization we aim to solve in this paper.

A. Stochastic ZOO

Consider a general optimization problem

$$\underset{x \in \mathcal{X}}{\mathrm{minimize}}\, f(x), \tag{1}$$

where

$$f(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[h(x, \xi)] \tag{2}$$

is continuously differentiable, $x \in \mathbb{R}^q$ is the optimization variable, $\mathcal{X}$ is its feasible domain and possibly non-convex, $\xi \in \mathbb{R}^p$ is a random variable and may denote the noisy data with distribution $\mathcal{D}$ in real applications, and $h(\cdot, \cdot): \mathbb{R}^q \times \mathbb{R}^p \to \mathbb{R}_{\geq 0}$ is a mapping such that the following assumptions hold:

Assumption 1: We assume the following statements hold for f.

A: The function f is (λ_x, ζ_x) locally Lipschitz in $\mathcal{X}$, i.e., for any $x \in \mathcal{X}$, if $\|x' - x\| \leq \zeta_x$ for $x' \in \mathbb{R}^q$, then

$$\|f(x') - f(x)\| \leq \lambda_x \|x' - x\|, \tag{3}$$

where λ_x and ζ_x are both continuous in x.

B: The function f has a (ϕ_x, β_x) locally Lipschitz gradient in $\mathcal{X}$, i.e., for any $x \in \mathcal{X}$, if $\|x' - x\| \leq \beta_x$ for $x' \in \mathbb{R}^q$, it holds that

$$\|\nabla_x f(x') - \nabla_x f(x)\| \leq \phi_x \|x' - x\|, \tag{4}$$

where ϕ_x and β_x are both continuous in x.

C: The function f is coercive, i.e., $f(x)$ goes to infinity when x approaches the boundary of $\mathcal{X}$.

The stochastic zeroth-order algorithm aims to solve the stochastic optimization (1) in the bandit setting, where one only has access to a noisy observation, i.e., the value of $h(x, \xi)$, while the detailed form of $h(x, \xi)$ is unknown. Since the gradient of $f(x)$, i.e., $\nabla_x f(x)$, can no longer be computed directly, it will be estimated based on the function value.

In the literature, the gradient can be estimated by the noisy observations with one-point feedback [3], [5], [6] or two-point feedback [2], [4], [9]. In what follows, we introduce the one-point estimation approach, based on which we will propose our algorithm. The work in this paper is trivially extendable to the two-point feedback case.

Define the unit ball and the $(d-1)$-dimensional sphere (surface of the unit ball) in $\mathbb{R}^d$ as

$$\mathbb{B}_d = \{y \in \mathbb{R}^d : \|y\| \leq 1\}, \tag{5}$$

and

$$\mathbb{S}_{d-1} = \{y \in \mathbb{R}^d : \|y\| = 1\}, \tag{6}$$

respectively. Given a sample $v \sim \mathrm{Uni}(\mathbb{B}_q)$, define

$$\hat{f}(x) = \mathbb{E}_{v \in \mathbb{B}_q}[f(x + rv)], \tag{7}$$

where $r > 0$ is called the smoothing radius. It is shown in [3] that

$$\nabla_x \hat{f}(x) = \mathbb{E}_{u \in \mathbb{S}_{q-1}}[f(x + ru)u]q/r, \tag{8}$$

which implies that $f(x + ru)uq/r$ is an unbiased estimate for $\nabla_x \hat{f}(x)$. Based on this estimation, a first-order stationary point of (1) can be obtained by using gradient descent.

B. Multi-Agent Stochastic Optimization

In this paper, we consider the scenario where optimization (1) is formulated based on a multi-agent system with N agents $\mathcal{V} = \{1, ..., N\}$. More specifically, let $x = (x_1^\top, ..., x_N^\top)^\top$, where $x_i \in \mathbb{R}^{q_i}$ is the state of each agent i, and $\sum_{i=1}^N q_i = q$. Similarly, let $\xi = (\xi_1^\top, ..., \xi_N^\top)^\top$, where $\xi_i \in \mathbb{R}^{p_i}$ denotes the noisy exploratory input applied to agent i, and $\sum_{i=1}^N p_i = p$. Different agents may have different dimensional states and noise vectors, and each agent i only has access to a local observation $h_i(x_{\mathcal{N}_i}, \xi_{\mathcal{N}_i})$, $i = 1, ..., N$. Here $x_{\mathcal{N}_i} = \{x_j, j \in \mathcal{N}_i\}$ and $\xi_{\mathcal{N}_i} = \{\xi_j, j \in \mathcal{N}_i\}$ are the vectors composed of the state and noise information of the agents in $\mathcal{N}_i$, respectively, where $\mathcal{N}_i = \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$ contains its neighbors determined by the graph[2] $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Note that each vertex in graph $\mathcal{G}$ is considered to have a self-loop, i.e., $i \in \mathcal{N}_i$. At this moment we do not impose other conditions for $\mathcal{G}$, instead, we give Assumption 2 on those local observations, based on which we propose our distributed RL algorithm. In Section IV, when applying our RL algorithm to a model-free multi-agent LQR problem, the details about how to design the inter-agent interaction graph for validity of Assumption 2 will be introduced.

Assumption 2: There exist local cost functions $f_i(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[h_i(x_{\mathcal{N}_i}, \xi_{\mathcal{N}_i})]$ such that for any $i \in \mathcal{V}$ and $x \in \mathcal{X}$, $h_i(\cdot, \cdot)$: $\mathbb{R}^{\sum_{j \in \mathcal{N}_i} q_j} \times \mathbb{R}^{\sum_{j \in \mathcal{N}_i} p_j} \to \mathbb{R}_{\geq 0}$ satisfies

$$h_i(x_{\mathcal{N}_i}, \xi_{\mathcal{N}_i}) \leq c f_i(x), \quad \text{almost surely (a.s.),} \tag{9}$$

and

$$\nabla_{x_i} f(x) = \nabla_{x_i} f_i(x). \tag{10}$$

Remark 1: Inequality (9) is used to build a relationship between $h_i(x, \xi)$ and its expectation w.r.t. ξ. Note that (9) is the only condition for the random variable ξ, and ξ does not necessarily have a zero mean. If $h_i(x, \xi)$ is continuous in ξ for any $x \in \mathcal{X}$, the assumption (9) holds when the random variable ξ is bounded. When ξ is unbounded, for example, ξ follows a sub-Gaussian distribution, (9) may hold with a high probability, see [41]. A truncation approach can be used when evaluating the local costs to ensure the boundedness of the observation if ξ follows a standard Gaussian distribution.

To show that Assumption 2 is reasonable, we give a class of feasible examples. Consider

$$h(x, \xi) = \sum_{(i,j) \in \mathcal{E}} F_{ij}(x_i, x_j, \xi_i, \xi_j), \quad \xi_i \sim \mathcal{D}_i, \tag{11}$$

where $F_{ij}(\cdot, \cdot, \cdot, \cdot) : \mathbb{R}^{q_i} \times \mathbb{R}^{q_j} \times \mathbb{R}^{p_i} \times \mathbb{R}^{p_j}$ is locally Lipschitz continuous and has a locally Lipschitz continuous gradient w.r.t. x_i and x_j, $\mathcal{E}$ is the edge set of a graph characterizing the inter-agent coupling relationship in the objective function, and $\mathcal{D}_i$ for each $i \in \mathcal{V}$ is a bounded distribution with zero mean. By setting $h_i(x_{\mathcal{N}_i}, \xi_{\mathcal{N}_i}) = \sum_{j \in \mathcal{N}_i} F_{ij}(x_i, x_j, \xi_i, \xi_j)$, Assumption 2 is satisfied. If we define the domain $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_N$ as a strategy set, the formulation (1) with (11) can be viewed as a cooperative networking game [42], where x_i is the decision variable of player i, each pair of players aim to minimize F_{ij}, and ξ_i is the random effect on the observation of each player. Moreover, the objective function (11) has also been employed in many multi-agent coordination problems [43] such as consensus [44] and formation control [27]. As a specific example, consider $F_{ij}(x_i, x_j, \xi_i, \xi_j) = \|x_i + \xi_i - x_j - \xi_j\|^2$. In this case, the objective is consensus where ξ_i and ξ_j model bounded noise effects in the measurement of x_i and x_j, respectively.

A direct consequence of Assumption 2 is that $f_i(x)$ is locally Lipschitz continuous and has a locally Lipschitz continuous gradient w.r.t. x_i. Moreover, the two Lipschitz constants are the same as those of $f(x)$.

In this paper, with the help of Assumption 2, we will propose a novel distributed zeroth-order algorithm, in which the local gradient of each agent is estimated based upon the local observations h_i's directly. The problem we aim to solve is summarized as follows:

Problem 1: Under Assumptions 1-2, given an initial state $x^0 \in \mathcal{X}$, design a distributed algorithm for each agent i based on the local observation[3] $h_i(x, \xi)$ such that by implementing the algorithm, the state x converges to a stationary point of f.

III. DISTRIBUTED ZOO WITH ASYNCHRONOUS SAMPLES AND UPDATES

In this section, we propose a distributed zeroth-order algorithm with asynchronous samples and updates based on an accelerated zeroth-order BCD algorithm.

A. Block Coordinate Descent

A BCD algorithm solves for the minimizer $x = (x_1^\top, ..., x_N^\top)^\top \in \mathbb{R}^q$ by updating only one block $x_i \in \mathbb{R}^{q_i}$ in each iteration. More specifically, let $b_k \in \mathcal{V}$ be the block to be updated at step k, and x^k be the value of x at step k. An accelerated BCD algorithm is shown below:

$$\begin{cases} x_i^{k+1} = x_i^k, & i \neq b_k, \\ x_i^{k+1} = \hat{x}_i^k - \eta \nabla_{x_i} f(\hat{x}_i^k, x_{-i}^k), & i = b_k, \end{cases} \tag{12}$$

---

[2] Here graph $\mathcal{G}$ can be either undirected or directed. In Section IV where our algorithm is applied to the LQR problem, graph $\mathcal{G}$ corresponds to the learning graph.

[3] Since $x_{\mathcal{N}_i}$ is composed of partial elements of x, for symbol simplicity, we write the local cost function for each agent i as $h_i(x, \xi)$ and treat $h_i(\cdot, \cdot)$ as a mapping from $\mathbb{R}^q \times \mathbb{R}^p$ to $\mathbb{R}_{\geq 0}$ while keeping in mind that $h_i(x, \xi)$ only involves agent i and its neighbors.

where $\eta > 0$ is the step-size, $\hat{x}_i^k$ is the extrapolation determined by

$$\hat{x}_i^k = x_i^k + w_i^k(x_i^k - x_i^{k_{prev}}), \tag{13}$$

here $w_i^k \geq 0$ is the extrapolation weight to be determined, $x_i^{k_{prev}}$ is the value of x_i before it was updated to x_i^k.

Note that w_i^k can be simply set as 0. However, it has been empirically shown in [23], [24] that having appropriate positive extrapolation weights helps significantly accelerate the convergence speed of the BCD algorithm, which will also be observed in our simulation results.

To avoid using the first-order information, which is usually absent in reality, we estimate $\nabla_{x_i} f(\hat{x}_i^k, x_{-i}^k)$ in (12) based on the observation $f_i(\hat{x}_i^k, x_{-i}^k)$, see the next subsection.

B. Gradient Estimation via Local Cost Evaluation

In this subsection, we introduce given $x \in \mathcal{X}$, how to estimate $\nabla_{x_i} f(x)$ based on $f_i(x)$. In the ZOO literature, $\nabla_x f(x)$ is estimated by perturbing state x with a vector randomly sampled from $\mathbb{S}_{q-1}$. In order to achieve distributed learning, we expect different agents to sample their own perturbation vectors independently. Based on Assumption 2, we have

$$\nabla_x f(x) = \begin{pmatrix} \nabla_{x_1} f(x) \\ \vdots \\ \nabla_{x_N} f(x) \end{pmatrix} = \begin{pmatrix} \nabla_{x_1} f_1(x) \\ \vdots \\ \nabla_{x_N} f_N(x) \end{pmatrix}. \tag{14}$$

Let

$$\hat{f}_i(x) = \mathbb{E}_{v_i \in \mathbb{B}_{q_i}}[f_i(x_i + r_i v_i, x_{-i})]$$
$$= \frac{\int_{r_i \mathbb{B}_{q_i}} f_i(x_i + v_i, x_{-i}) dv_i}{V(r_i \mathbb{B}_{q_i})}, \tag{15}$$

where $V(r_i \mathbb{B}_{q_i})$ is the volume of $r_i \mathbb{B}_{q_i}$. Here $\hat{f}_i(x)$ is always differentiable even when $f_i(x)$ is not differentiable. To approximate $\nabla_{x_i} f_i(x)$ for each agent i, we approximate $\nabla_{x_i} \hat{f}_i(x)$ by the following one-point feedback:

$$g_i(x, u_i, \xi) = \frac{q_i}{r_i} h_i(x_i + r_i u_i, x_{-i}, \xi) u_i, \tag{16}$$

$u_i \in \text{Uni}(\mathbb{S}_{q_i-1})$, $\xi \in \mathbb{R}^p$ is a random variable following the distribution $\mathcal{D}$. Note that according to the definition of the local cost function $h_i(x, \xi)$, $g_i(x, u_i, \xi)$ may be only affected by partial components of ξ.

The following lemma shows that $g_i(x, u_i, \xi)$ is an unbiased estimate of $\nabla_{x_i} \hat{f}_i(x)$.

Lemma 1: Given $r_i > 0$, $i = 1, ..., N$, the following holds

$$\nabla_{x_i} \hat{f}_i(x) = \mathbb{E}_{u_i \in \mathbb{S}_{q_i-1}} \mathbb{E}_{\xi \sim \mathcal{D}}[g_i(x, u_i, \xi)]. \tag{17}$$

Although $\nabla_{x_i} \hat{f}_i(x) \neq \nabla_{x_i} f(x)$, their error can be quantified using the smoothness property of $f(x)$, as shown below.

Lemma 2: Given a point $x \in \mathbb{R}^q$, if $r_i \leq \beta_x$, then

$$\|\nabla_{x_i} \hat{f}_i(x) - \nabla_{x_i} f(x)\| \leq \phi_x r_i. \tag{18}$$

Remark 2: Compared with gradient estimation based on one-point feedback, the two-point feedback

$$g_i(x, u_i, \xi_i) = \frac{q_i}{r_i} \left[h_i(x_i + r_i u_i, x_{-i}, \xi_i) - h_i(x_i - r_i u_i, x_{-i}, \xi_i) \right] u_i$$

is recognized as a more robust algorithm with a smaller variance and a faster convergence rate [2], [4], [9]. In this work, we mainly focus on how to solve an optimization via networks by a distributed zeroth-order BCD algorithm. Since the expectation of the one-point feedback is equivalent to the expectation of the two-point feedback, our algorithm in this paper is extendable to the two-point feedback case. Note that in the gradient estimation, the two-point feedback requires two times of policy evaluation with the same noise vector, which may be unrealistic in practical applications.

C. Distributed ZOO Algorithm with Asynchronous Samplings

In this subsection, we propose a distributed ZOO algorithm with asynchronous sample and update schemes based on the BCD algorithm (12) and the gradient approximation for each agent i. According to (16), we have the following approximation for each agent i at step k:

$$g_i(\hat{x}_i^k, x_{-i}^k, u_i^k, \xi) = \frac{q_i}{r_i} h_i(\hat{x}_i^k + r_{ik} u_i^k, x_{-i}^k, \xi^k) u_i^k, \tag{19}$$

where u_i^k is uniformly randomly sampled from $\mathbb{S}_{q_i-1}$.

In fact, since $h_i(x, \xi)$ only involves the agents in $\mathcal{N}_i$, it suffices to maintain the variables of the agents in $\mathcal{N}_i \setminus \{i\}$ invariant when estimating the gradient of agent i. That is, in our problem, two agents are allowed to update their variables simultaneously

if they are not neighbors. This is different from a standard BCD algorithm where only one block of the entire variable is updated in one iteration.

To achieve simultaneous update for non-adjacent agents, we decompose the set of agents $\mathcal{V}$ into s independent clusters (s has an upper bound depending on the graph), i.e., $\mathcal{V} = \cup_{j=1}^{s} \mathcal{V}_j$, and

$$\mathcal{V}_{j_1} \cap \mathcal{V}_{j_2} = \varnothing, \quad \forall \text{ distinct } j_1, j_2 \in \{1, ..., s\} \tag{20}$$

such that the agents in the same cluster are not adjacent in the interaction graph, i.e., $(i_1, i_2) \notin \mathcal{E}$ for any $i_1, i_2 \in \mathcal{V}_j$, $j \in \{1, ..., s\}$. Note that no matter graph $\mathcal{G}$ is directed or undirected, any two agents have to lie in different clusters if there is a link from one to the other. Without loss of generality, we assume $\mathcal{G}$ is undirected, and let $\mathcal{N}_i$ be the neighbor set of agent i. In the case when $\mathcal{G}$ is directed, we define $\mathcal{N}_i$ as the set of agents j such that $(i, j) \in \mathcal{E}$ or $(j, i) \in \mathcal{E}$. A simple algorithm for achieving such a clustering is shown in Algorithm 1. Note that the number of clusters obtained by implementing Algorithm 1 may be different each time. In practical applications, Algorithm 1 can be modified to solve for a clustering with maximum or minimum number of clusters.

Algorithm 1 Non-Adjacent Agents Clustering

Input: $\mathcal{V}$, $\mathcal{N}_i$ for $i = 1, ..., N$.
Output: s, $\mathcal{V}_j$, $j = 1, ..., s$.
1. Set $s = 0$, $\mathcal{C} = \varnothing$.
2. **while** $\mathcal{C} \neq \mathcal{V}$
3. Set $s \leftarrow s + 1$, $\mathcal{D} = \mathcal{C}$, **while** $\mathcal{D} \neq \mathcal{V}$
4. Randomly select i from $\mathcal{V} \setminus \mathcal{D}$, set $\mathcal{V}_s \leftarrow \mathcal{V}_s \cup \{i\}$, $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{N}_i$.
5. **end**
6. $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{V}_s$.
7. **end**

Algorithm 1 requires the global graph information as an input, thus is centralized. Nonetheless, Algorithm 1 is only implemented in one shot, and there is no real-time global information required during its implementation. Therefore, Algorithm 1 can be viewed as an off-line centralized deployment before implementing the distributed policy seeking algorithm. Based on the clustering obtained by Algorithm 1, we propose Algorithm 2 as the asynchronous distributed zeroth-order algorithm. Algorithm 2 can be viewed as a distributed RL algorithm where different clusters take actions asynchronously, different agents in one cluster take actions simultaneously and independently. In Algorithm 2, step 5 can be viewed as *policy evaluation* for agent i, while step 6 corresponds to *policy iteration*. Moreover, the local observation $h_i(\hat{x}_i^k + r_i u_i^k, x_{-i}^k, \xi^k)$ can be viewed as the reward returned by the environment to agent i.

Algorithm 2 Distributed Zeroth-Order Algorithm with Asynchronous Samplings

---

Input: Step-size η, smoothing radius r_i and variable dimension q_i, $i = 1, ..., N$, clusters $\mathcal{V}_j$, $j = 1, ..., s$, iteration number T, update order z_k (the index of the cluster to be updated at step k) and extrapolation weight w_i^k, $k = 0, ..., T - 1$, initial point $x^0 \in \mathcal{X}$.
Output: $x(T)$.
1. **for** $k = 0, 1, ..., T - 1$ **do**
2. Sample $\xi^k \sim \mathcal{D}$.
3. **for** all $i \in \mathcal{V}$ **do**
4. **if** $i \in \mathcal{V}_{z_k}$ **do** (Simultaneous Implementation)
5. Agent i computes $\hat{x}_i^k$ by (13), samples u_i^k randomly from $\mathbb{S}_{q_i-1}$ and observes $h_i(\hat{x}_i^k + r_i u_i^k, x_{-i}^k, \xi^k)$.
6. Agent i computes the estimated local gradient $g_i(\hat{x}_i^k, x_{-i}^k, u_i^k, \xi^k)$ according to (19). Then updates its policy:

$$x_i^{k+1} = \hat{x}_i^k - \eta g_i(\hat{x}_i^k, x_{-i}^k, u_i^k, \xi^k). \tag{21}$$

7. **else**
8.

$$x_i^{k+1} = x_i^k. \tag{22}$$

9. **end**
10. **end**
11. **end**

In the literature, BCD algorithms have been studied with different update orders such as deterministically cyclic [21], [23] and randomly shuffled [22], [26]. In this paper, we adopt an "essentially cluster cyclic update" scheme, which includes the standard cyclic update as a special case, and is a variant of the essentially cyclic update scheme in [23], [24], see the following assumption.

Assumption 3: (Essentially Cluster Cyclic Update) Given integer $T_0 \geq s$, for any cluster $j \in \{1, ..., s\}$ and any two steps k_1 and k_2 such that $k_2 - k_1 = T_0 - 1$, there exists $k_0 \in [k_1, k_2]$ such that $z_{k_0} = j$.

Assumption 3 implies that each cluster of agents update their states at least once during every consecutive T_0 steps. When $|\mathcal{V}_i| = 1$ for $i = 1, ..., s$, and $s = N$, Assumption 3 implies an essentially cyclic update in [23], [24].
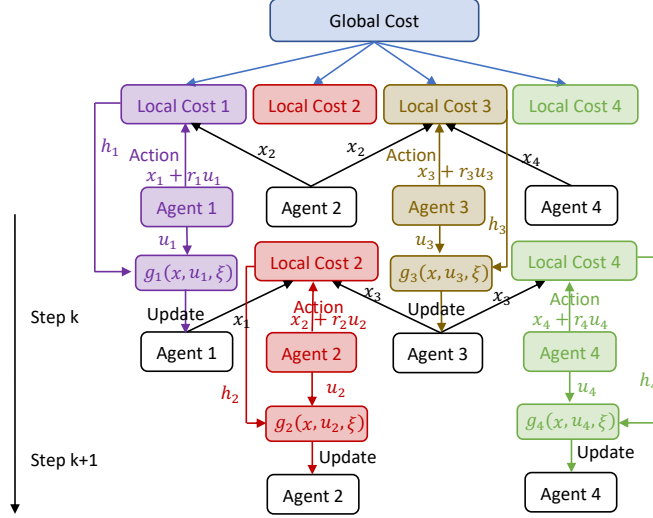
Fig. 1. The architecture of distributed RL via asynchronous actions during two consecutive iterations.

To better understand Algorithm 2, let us look at a multi-agent coordination example.

Example 1: Suppose that the global function to be minimized is (11) with $\mathcal{E} = \{(1,2),(2,3),(3,4)\}$ being the edge set. Then the local cost function for each agent is:

$$
\begin{aligned}
h_1 &= F(x_1, x_2, \xi_{12}), \quad h_2 = F(x_1, x_2, \xi_{12}) + F(x_2, x_3, \xi_{23}), \\
h_3 &= F(x_2, x_3, \xi_{23}) + F(x_3, x_4, \xi_{34}), \quad h_4 = F(x_3, x_4, \xi_{34}),
\end{aligned}
\tag{23}
$$

where ξ_{ij} is a bounded zero-mean noise. By implementing Algorithm 1, the two clusters obtained are $\mathcal{V}_1 = \{1,3\}$ and $\mathcal{V}_2 = \{2,4\}$. By implementing Algorithm 2 with two clusters updating alternatively, the diagram for two consecutive iterations is shown in Fig. 1, where agents in $\mathcal{V}_1$ and $\mathcal{V}_2$ take their actions successively. In multi-agent coordination, $F(x_i, x_j, \xi_{ij})$ can be set as $(\|x_i - x_j + \xi_{ij}\|^2 - d_{ij}^2)^2$ for distance-based formation control (d_{ij} is the desired distance between agents i and j), and $\|x_i - x_j + \xi_{ij} - h_{ij}\|^2$ for displacement-based formation control (h_{ij} is the desired displacement between agents i and j).

D. Convergence Result

We study the convergence of Algorithm 2 by focusing on $x \in \mathbb{X}$, where $\mathbb{X}$ is defined as

$$
\mathbb{X} = \{x \in \mathbb{R}^{nN} : f(x) \leq \alpha f(x^0), f_i(x) \leq \alpha_i f_i(x^0), \forall \, i \in \mathcal{V}\} \subseteq \mathcal{X},
\tag{24}
$$

where $\alpha, \alpha_i > 1$, $i \in \mathcal{V}$, $x^0 \in \mathbb{X}$ is the given initial condition. Since $f(x)$ is continuous and coercive, the set $\mathbb{X}$ is compact. Then we are able to find uniform parameters feasible for $f(x)$ over $\mathbb{X}$:

$$
\phi_0 = \sup_{x \in \mathbb{X}} \phi_x, \quad \lambda_0 = \sup_{x \in \mathbb{X}} \lambda_x, \quad \rho_0 = \inf_{x \in \mathbb{X}} \{\beta_x, \zeta_x\}.
\tag{25}
$$

The following theorem shows an approximate convergence result based on establishing the probability of the event $\{x^k \in \mathbb{X}\}$ for $k = 0, ..., T-1$. Let N_i denote the number of agents in the cluster containing agent i. For notation simplicity, we denote $N_0 = \max_{i \in \mathcal{V}} N_i = \max_{j \in \{1,...,s\}} |\mathcal{V}_j|$, $q_+ = \max_{i \in \mathcal{V}} q_i$, $r_- = \min_{i \in \mathcal{V}} r_i$, $f_0(x^0) = \max_{i \in \mathcal{V}} \alpha_i f_i(x^0)$.

Theorem 1: Under Assumptions 1-2, given positive scalars ϵ, ν, γ, and $\alpha \geq 2 + \gamma + \frac{1}{\nu} + \nu\gamma$, $x^0 \in \mathcal{X}$. Let $\{x^k\}_{k=0}^{T-1}$ be a sequence of states obtained by implementing Algorithm 2 for $k = 0, ..., T-1$. Suppose that

$$
\begin{aligned}
T &= \lceil \frac{2\alpha\nu f(x^0)}{\eta\epsilon} \rceil, \quad \eta \leq \min\{\frac{\rho_0}{2\delta\sqrt{N_0}}, \frac{2\alpha f(x^0)}{\gamma\epsilon}, \frac{\gamma\epsilon}{2\alpha N_0(\phi_0\delta^2 + 4\phi_0^2 + \phi_0 + 4)}\}, \\
w_i^k &\leq \frac{1}{\|x_i^k - x_i^{k_{prev}}\|} \min\{\eta^{3/2}, \frac{\rho_0}{2\sqrt{N_i}}\}, \quad r_i \leq \min\{\frac{\rho_0}{2}, \frac{1}{2\phi_0}\sqrt{\frac{\gamma\epsilon}{\alpha N_0}}\}, i \in \mathcal{V},
\end{aligned}
\tag{26}
$$

where $\delta = \frac{q_+}{r_-} c \left[f_0(x^0) + \lambda_0\rho_0\right]$ is the uniform bound on the estimated gradient (as shown in Lemma 6). The following statements hold.

(i). The following inequality holds with a probability at least $1 - \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma)$:

$$
\frac{1}{T}\sum_{k=0}^{T-1}\sum_{i \in \mathcal{V}_{z_k}} \|\nabla_{x_i} f(x^k)\|^2 < \epsilon.
\tag{27}
$$

(ii). Under Assumption 3, suppose that there exists some $\bar{\epsilon} \in (0, \rho_0]$ such that

$$w_i^k \leq \frac{\bar{\epsilon}}{2(T_0-1)N_i\|x_i^k - x_i^{k_{prev}}\|}, \quad \eta \leq \frac{\bar{\epsilon}}{2\delta(T_0-1)N_0}, \tag{28}$$

for any step $k \in \{0, ..., T-1\}$, and conditions in (26) are satisfied, then the following holds with a probability at least $1 - \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma)$:

$$\frac{1}{T}\sum_{k=0}^{T-1}\|\nabla_x f(x^k)\|^2 < \hat{\epsilon}. \tag{29}$$

where $\hat{\epsilon} = 2T_0(\epsilon + \phi_0^2\bar{\epsilon}^2)$.

Approximate Convergence. Given positive scalars ϵ α, γ, and ν, Theorem 1 (i) implies that Algorithm 2 converges to a sequence $\{x_k\}$, where the partial derivative of x_k w.r.t. one cluster is close to 0 at each step with high probability. When $s = 1$, Algorithm 2 reduces to an accelerated zeroth-order gradient descent algorithm, and Theorem (i) implies convergence to a ϵ-accurate stationary point with high probability. Theorem 1 (ii) implies that under an "essentially cluster cyclic update" scheme, Algorithm 2 converges to a $\hat{\epsilon}$-accurate stationary point if the extrapolation weight w_i^k and the step-size η are both sufficiently small.

Convergence Probability. The convergence probability can be controlled by adjusting parameters α, γ and ν. Statements in a similar nature have appeared in literature (see e.g., [18, Theorem 1]). However, instead of giving an explicit probability, we can use ν and γ to adjust the probability. For example, set $\alpha = 20$, $\gamma = 1$, $\nu = 1$, then the probability is at least $1 - 1/4 = 3/4$. To achieve a high probability of convergence, it is desirable to have large α and ν, and small γ, which implies that the performance of the algorithm can be enhanced at the price of adopting a large number of samples. As shown in (26), for a given ϵ and a sufficiently large α, γ is used to control the step-size η and the sampling radius r_i while ν is used to control the total number of iterations.

Sample Complexity. To achieve high accuracy of the convergent result, we consider ϵ as a small positive scalar such that $\epsilon \ll 1/\epsilon$. As a result, the sample complexity for convergence of Algorithm 2 is $T \sim \mathcal{O}(q_+^2 N_0^2/\hat{\epsilon}^3)$. This implies that the required iteration number mainly depends on the highest dimension of the variable for one agent, the largest size of one cluster, and the number of clusters (because $T_0 \geq s$). Note that even when the number of agents increases, q_+ may remain the same, implying high scalability of our algorithm to large-scale networks. Moreover, N_0 may increase as the number of clusters s decreases. Hence, there is a trade-off between the benefits of minimizing the largest cluster size against minimizing the number of clusters. When the network is of large scale, T_0 dominates the sample complexity, which makes minimizing the number of clusters the optimal clustering strategy. Note that the sample complexity is directly associated with the convergence accuracy. Analysis with diminishing step-sizes is of interest and may help achieve asymptotic convergence of the gradient.

Theorem 1 provides sufficient conditions for Algorithm 2 to converge. The objective of the analysis is to identify the range of applicability of the proposed algorithm and establish the qualitative behaviors of the algorithms, rather than providing optimal choices of algorithmic parameters. Some of the parameters, such as the Lipschitz constants, have analytical expressions in the model-based setting (see e.g., [33]). However, they are indeed difficult to obtain in the model-free setting. Therefore, in our experiments, we choose small step-sizes empirically to ensure that the cost function decreases over time.

E. Variance Analysis

In this subsection, we analyze the variance of our gradient estimation strategy and make comparisons with the gradient estimation via global cost evaluation. Without loss of generality, we analyze the estimation variance for the i-th agent. Let $u_i \sim \text{Uni}(\mathbb{S}_{q_i-1})$, $z \sim \text{Uni}(\mathbb{S}_{q-1})$, and $z_i \in \mathbb{R}^{q_i}$ be a component of z corresponding to agent i. Under the same smoothing radius $r > 0$, one time gradient estimates based on the local cost h_i and the global cost h are

$$g_l = \frac{q_i}{r}h_i(x_i + ru_i, x_{-i}, \xi)u_i, \tag{30}$$

and

$$g_g = \frac{q}{r}h(x + rz, \xi)z_i, \tag{31}$$

respectively.

Lemma 3: The covariance matrices for the gradient estimates (30) and (31) are

$$\text{Cov}(g_l) = \frac{q_i}{r^2}\left[\mathbb{E}[h_i^2(x,\xi)]I_{q_i} - \mathbb{E}[\nabla_{x_i}h_i(x,\xi)]\mathbb{E}^\top[\nabla_{x_i}h_i(x,\xi)] + \mathcal{O}(r^2)\right], \tag{32}$$

and

$$\text{Cov}(g_g) = \frac{q}{r^2}\left[\mathbb{E}[h^2(x,\xi)]I_{q_i} - \mathbb{E}[\nabla_{x_i}h(x,\xi)]\mathbb{E}^\top[\nabla_{x_i}h(x,\xi)] + \mathcal{O}(r^2)\right], \tag{33}$$

respectively.

Due to Assumption 2, the difference between two covariance matrices is

$$\text{Cov}(g_g) - \text{Cov}(g_l) = \frac{1}{r^2}\left[\sum_{j \in \mathcal{V}\backslash\{i\}} q_j \mathbb{E}[h^2(x,\xi)]I_{q_i} + q_i\mathbb{E}[h^2(x,\xi) - h_i^2(x,\xi)]I_{q_i} + \mathcal{O}(r^2)\right], \tag{34}$$

where the first term is positive definite as long as there are more than one agent in the network, the second term is usually positive semi-definite, and the third term is negligible if r is much smaller than $h(x,\xi)$.

Observe that the first two terms may have extremely large traces if the network is of large scale. This implies that the gradient estimate (30) leads to a high scalability of our algorithm to large-scale network systems.

IV. Application to Distributed RL of Model-Free Distributed Multi-Agent LQR

In this section, we will show how Algorithm 2 can be applied to distributively learning a sub-optimal distributed controller for a linear multi-agent system with unknown dynamics.

A. Multi-Agent LQR

Consider the following MAS with decoupled agent dynamics[4]:

$$x_i(t+1) = A_i x_i(t) + B_i u_i(t), \quad i = 1, ..., N \tag{35}$$

where $x_i \in \mathbb{R}^n$, $u_i \in \mathbb{R}^m$, A_i and B_i are assumed to be unknown. The entire system dynamics becomes

$$x(t+1) = \mathcal{A}x(t) + \mathcal{B}u(t). \tag{36}$$

By considering random agents' initial states, we study the following LQR problem:

$$\min_K \quad J(K) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t x^\top(t)(Q + K^\top RK)x(t)\right] \tag{37}$$
$$\text{s.t.} \quad x(t+1) = (\mathcal{A} - \mathcal{B}K)x(t), \quad x(0) \sim \mathcal{D},$$

here $0 < \gamma \leq 1$, $\mathcal{D}$ is a distribution such that $x(0)$ is bounded and has a positive definite second moment $\Sigma_x = \mathbb{E}[x(0)x^\top(0)]$, $Q = G \odot \tilde{Q}$ with $G \in \mathbb{R}^{N \times N} \succeq 0$, $\tilde{Q} \in \mathbb{R}^{nN \times nN} \succeq 0$, and the symbol $\odot$ denoting the Khatri-Rao product, and $R = \text{diag}\{R_1, ..., R_N\} \succ 0$. Note that while we assume that the agents' dynamics are unknown, the matrices Q and R are considered known. This is reasonable because in reality Q and R are usually artificially designed. It has been shown in [6] that when $0 < \gamma < 1$, the formulation (37) is equivalent to the LQR problem with fixed agents' initial states and additive noises in dynamics, where the noise follows the distribution $\mathcal{D}$. Hence, our results are extendable to LQR with noisy dynamics.

When $\gamma \in (0, 1)$, let $y(0) = x(0)$, and $y(t+1) = \sqrt{\gamma}(\mathcal{A} - \mathcal{B}K)y(t)$, then we have $y(t) = \gamma^{t/2}x(t)$. It follows that $J(K) = \mathbb{E}[\sum_{t=0}^{\infty} y^\top(t)(Q + K^\top RK)y(t)]$. This implies that $J(K)$ remains the same as that for $\gamma = 1$ by replacing $\mathcal{A}$ and $\mathcal{B}$ in system dynamics (36) with $\sqrt{\gamma}\mathcal{A}$ and $\sqrt{\gamma}\mathcal{B}$. Hence, we define the following set:

$$\mathbb{K}_s = \{K \in \mathbb{R}^{mN \times nN} : \sqrt{\gamma}(\mathcal{A} - \mathcal{B}K) \text{ is Schur Stable}\}. \tag{38}$$

Note that any $K \in \mathbb{K}_s$ always renders $J(K)$ finite because a Schur stabilizing gain always renders $J(K)$ with $\gamma = 1$ finite.

Based on [33, Lemma 3.7] and [6, Lemmas 4, 5], we give some properties of $J(K)$ in the following lemma.

Lemma 4: The cost function $J(K)$ in (37) has the following properties:

(i) $J(K)$ is coercive in the sense that $J(K) \to \infty$ if $K \to \partial\mathbb{K}_s$;

(ii) For any $K \in \mathbb{K}_s$, there exist continuous positive parameters λ_K, ζ_K, ϕ_K and β_K such that the cost function $J(K)$ in (37) is (λ_K, ζ_K) locally Lipschitz continuous and has a (ϕ_K, β_K) locally Lipschitz continuous gradient.

Based on the matrix G, we define the *cost graph* interpreting inter-agent coupling relationships in the cost function.

Definition 1: (**Cost Graph**) The cost graph $\mathcal{G}_C = (\mathcal{V}, \mathcal{E}_C)$ is an undirected graph such that $G_{ij} \neq 0$ if and only if $(i, j) \in \mathcal{E}_C$. The neighbor set of agent i in the cost graph is defined as $\mathcal{N}_C^i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}_C\}$.

Distributed control implies that each agent only needs to sense state information of its local neighbors. Next we define *sensing graph* interpreting required inter-agent sensing relationships for distributed control.

Definition 2: (**Sensing Graph**) The sensing graph $\mathcal{G}_S = (\mathcal{V}, \mathcal{E}_S)$ is a directed graph with each agent having a self-loop. The neighbor set for each agent i in graph $\mathcal{G}_S$ is defined as $\mathcal{N}_S^i = \{j \in \mathcal{V} : (j, i) \in \mathcal{E}_S\}$, where $(j, i) \in \mathcal{E}_S$ implies that agent i has access to x_j.

Notes for the cost graph and the sensing graph.

---

[4]Decoupled agent dynamics is common for multi-agent systems, e.g., a group of robots. Our results are extendable to the case of coupled dynamics. In that case, the coupling relationship in agents' dynamics has to be taken into account in the design of the local cost function and the learning graph in next subsection. The details are explained in Remark 4.

- The cost graph $\mathcal{G}_C$ is determined by the prespecified cost function, and is always undirected becuase Q is positive semi-definite.
- We assume $\mathcal{G}_C$ is connected. Note that if $\mathcal{G}_C$ is disconnected, then the performance index in (37) can be naturally decomposed according to those connected components, and the LQR problem can be transformed to smaller sized LQR problems.
- In real applications, the sensing graph is designed based on the sensing capability of each agent. It is even not necessarily weakly connected.
- Here the cost graph $\mathcal{G}_C$ and the sensing graph $\mathcal{G}_S$ are defined independently. In specific applications, they can be either related to or independent of each other.

Let $X(i,j) \in \mathbb{R}^{m \times n}$ be a submatrix of $X \in \mathbb{R}^{mN \times nN}$ consisting of elements of X on $(i-1)m+1$-th to im-th rows and $(j-1)n+1$-th to jn-th columns. The space of distributed controllers is then defined as

$$\mathbb{K}_d = \{X \in \mathbb{R}^{mN \times nN} : X(i,j) = \mathbf{0}_{m \times n} \text{ if } j \notin \mathcal{N}_S^i, \ i,j \in \mathcal{V}\}. \tag{39}$$

We make the following assumption to guarantee that the distributed LQR problem is solvable.

Assumption 4: $\mathbb{K}_d \cap \mathbb{K}_s \neq \varnothing$.

We aim to design a distributed RL algorithm for agents to learn a sub-optimal distributed controller $K^* \in \mathbb{K}_d$ such that during the learning process, each agent only requires information from partial agents (according to the sensing graph), and takes actions based on the obtained information.

B. Local Cost Function and Learning Graph Design

We have verified Assumption 1 in Lemma 4. To apply Algorithm 2, it suffices to find local cost functions such that Assumption 2 holds. In this subsection, we propose an approach to design of such local cost functions.

Note that the cost function can be written as a function of K:

$$\begin{aligned}
J(K) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t x^\top(t)(Q + K^\top RK)x(t)\right] \\
&= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t x^\top(0)(\mathcal{A} - \mathcal{B}K)^{t\top}(Q + K^\top RK)(\mathcal{A} - \mathcal{B}K)^t x(0)\right].
\end{aligned} \tag{40}$$

Let $K = [K_1^\top, ..., K_N^\top]^\top \in \mathbb{R}^{mN \times nN}$. Then $K_i \in \mathbb{R}^{m \times nN}$ is the local gain matrix to be designed for agent i. Based on the definition of $\mathbb{K}_d$ in (39), the distributed controller for each agent i has the form:

$$u_i = -K_i x = -\tilde{K}_i x_{\mathcal{N}_S^i}, \tag{41}$$

where $\tilde{K}_i \in \mathbb{R}^{m \times n_i}$ with $n_i = |\mathcal{N}_S^i|n$. We now view the control gain K_i for each agent i as the optimization variable. According to Assumption 2, we need to find a local cost $J_i(K)$ for each agent i such that its gradient is the same as the gradient of the global cost w.r.t. K_i. That is, $\nabla_{K_i} J_i(K) = \nabla_{K_i} J(K)$.

To design the local cost J_i for each agent i, we define the following set including all the agents whose control inputs and states will be affected by agent i during the implementation of the distributed controller:

$$\mathcal{V}_S^i = \{j \in \mathcal{V} : \text{A path from } i \text{ to } j \text{ exists in } \mathcal{G}_S\}. \tag{42}$$

Since different agents are coupled in the cost function, when extracting the local cost function involving an agent $j \in \mathcal{V}_S^i$ from the entire cost function, all of its neighbors in the cost graph (i.e., $\mathcal{N}_C^j$) should be taken into account. Based on the set $\mathcal{V}_S^i$ for each agent i, we formulate the following feasibility problem for each agent $i \in \mathcal{V}$:

$$\begin{aligned}
\text{find} \quad & M_i \in \mathbb{R}^{N \times N} \\
\text{s.t.} \quad & M_i[j,k] = G_{jk}, \text{ for all } k \in \mathcal{N}_C^j, j \in \mathcal{V}_S^i, \\
& M_i[j,k] = 0, \quad k \in \mathcal{V} \setminus \cup_{j \in \mathcal{V}_S^i} \mathcal{N}_C^j,
\end{aligned} \tag{43}$$

where $M_i[j,k]$ is the element of matrix M_i on the j-th row and k-th column.

The solution M_i to (43) must satisfy

$$\frac{\partial(x^\top(M_i \odot \tilde{Q})x)}{\partial x_j} = \frac{\partial(x^\top(G \odot \tilde{Q})x)}{\partial x_j} \text{ for all } j \in \mathcal{V}_S^i. \tag{44}$$

Moreover, we observe that the solution $M_i \in \mathbb{R}^{N \times N}$ is actually the matrix with the same principal submatrix associated with $\cup_{j \in \mathcal{V}_S^i} \mathcal{N}_C^j$ as G, and all the other elements of M_i are zeros. Then $M_i \succeq 0$ because all the principal minors of M_i are nonnegative.

$$\mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^{\top}x^{\top}(t)(Q+K^{\top}RK)x(t)\right]$$

Desired Structure in K

Cost Graph

Sensing Graph

Learning Graph
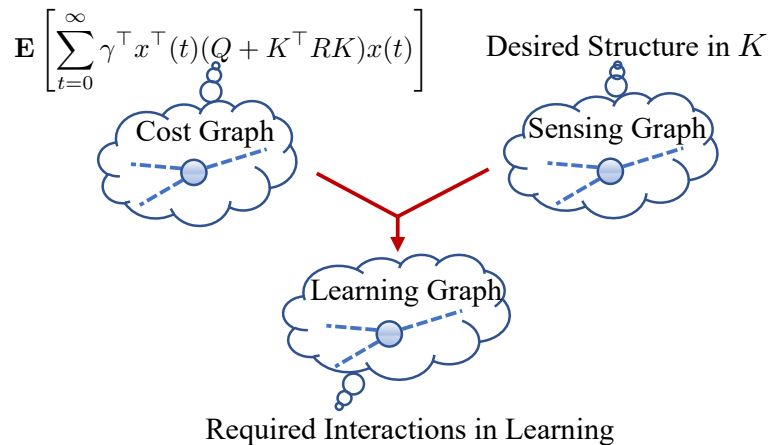
Required Interactions in Learning

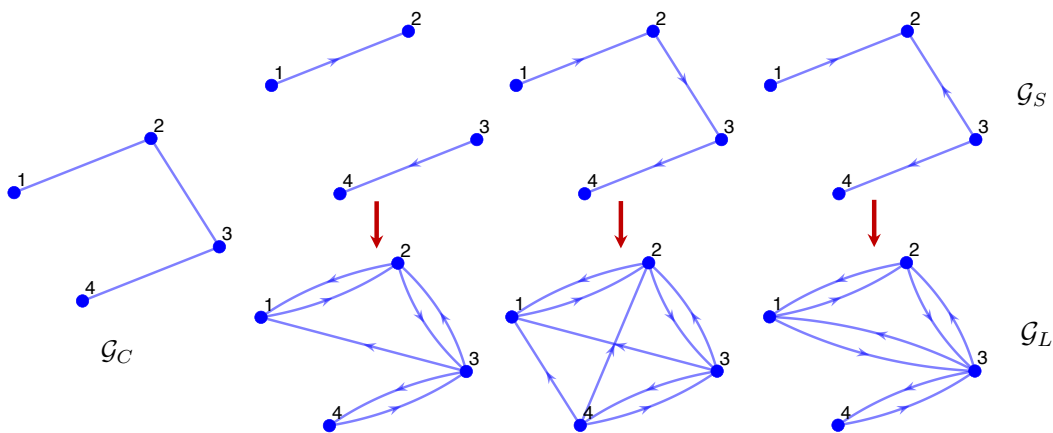Fig. 2. Summary of definitions for the cost, sensing, and learning graphs.

Fig. 3. With the same cost graph, three different sensing graphs result in three different learning graphs. Each node in each graph has a self-loop, which is omitted in this figure.

Now, based on the cost graph $\mathcal{G}_C$ and the sensing graph $\mathcal{G}_S$, we give the definition for the communication graph required in distributed learning.

Definition 3: (**Learning Graph**) The learning-required communication graph $\mathcal{G}_L = (\mathcal{V}, \mathcal{E}_L)$ is a directed graph with the edge set $\mathcal{E}_L$ defined as

$$\mathcal{E}_L = \{(k,i) \in \mathcal{V} \times \mathcal{V} : k \in \cup_{j \in \mathcal{V}_S^i} \mathcal{N}_C^j, i \in \mathcal{V}\}. \tag{45}$$

The neighbor set for each agent i in graph $\mathcal{G}_L$ is defined as $\mathcal{N}_L^i = \{k \in \mathcal{V} : (k,i) \in \mathcal{E}_L\}$, where $(k,i) \in \mathcal{E}_L$ implies that there is an oriented link from k to i.

To better understand the three different graphs, we summarize their definitions in Fig. 2, and show three examples demonstrating the relationships between $\mathcal{G}_S$, $\mathcal{G}_C$ and $\mathcal{G}_L$ in Fig. 3.

Remark 3: There are two points we would like to note for the learning graph. First, since we consider that all the agents have self-loops, by Definition 3, each edge of graph $\mathcal{G}_C$ must be an edge of $\mathcal{G}_L$. Second, when $\mathcal{G}_S$ is strongly connected, $\mathcal{G}_L$ is a complete graph because in this case $\mathcal{V}_S^i = \mathcal{V}$ for all $i \in \mathcal{G}$. As a result, if we regard each node in a sensing graph in Fig. 3 as a strongly connected component, then the resulting learning graph $\mathcal{G}_L$ still has the same structure as it is shown in Fig. 3, where each node denotes a fully connected component, and the edge from node a to another node b denotes edges from all agents in component a to all agents in component b.

Remark 4: When the multi-agent system has coupled dynamics, there will be another graph $\mathcal{G}_D = (\mathcal{V}, \mathcal{E}_D)$ describing the inter-agent coupling in dynamics. We assume that $\mathcal{G}_D$ is known, which is reasonable in many model-free scenarios because we only need to have the coupling relationship between different agents. In this scenario, to construct the learning graph, the sensing graph used in defining (45) should be replaced by a new graph $\mathcal{G}_{SD} = (\mathcal{V}, \mathcal{E}_{SD})$ with $\mathcal{E}_{SD} = \mathcal{E}_D \cup \mathcal{E}_S$, where $\mathcal{E}_S$ is the edge set of the graph $\mathcal{G}_S$ describing the desired structure of the distributed control gain. If $\mathcal{E}_D \subseteq \mathcal{E}_S$, then the learning graph is the same for the multi-agent networks with coupled and decoupled dynamics. The learning graph describes the required information flow between the agents. However, the learning process may be distributed by consensus-based estimation to avoid

having a dense communication graph. Another way to reduce communication is to redesign the local cost of each agent by ignoring agents that are far away from it in $\mathcal{G}_D$. In [45], it has been shown that ignoring those agents beyond the κ-hop neighborhood of each agent would lead to biases on the order of $\gamma^{\kappa+1}$ in convergence.

Let $\{M_i\}_{i=1}^N$ be solutions to (43) for $i = 1, ..., N$. Also let $\hat{Q}_i = M_i \odot \tilde{Q}$ and $\hat{R}_i = \sum_{j \in \mathcal{N}_L^i}(e_j^\top \otimes I_m)R_j(e_j \otimes I_m)$. By collecting the parts of the entire cost function involving agents in $\mathcal{N}_L^i$, we define the local cost J_i as

$$
\begin{aligned}
J_i(K) &= \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t (x^\top \hat{Q}_i x + x^\top K^\top \hat{R}_i Kx)\right] \\
&= \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t (x_{\mathcal{N}_L^i}^\top \bar{Q}_i x_{\mathcal{N}_L^i} + u_{\mathcal{N}_L^i}^\top \bar{R}_i u_{\mathcal{N}_L^i})\right],
\end{aligned}
\tag{46}
$$

where $\bar{Q}_i$ is the maximum nonzero principal submatrix of $M_i \odot \tilde{Q}$ and $\bar{R}_i = \text{diag}\{R_j\}_{j \in \mathcal{N}_L^i}$.

Remark 5: Each agent i computes its local cost $J_i(x_{\mathcal{N}_L^i}, u_{\mathcal{N}_L^i})$ based on x_j and $u_j^\top R_j u_j$, $j \in \mathcal{N}_L^i$. In practical application, agent i may compute a finite horizon cost value to approximate the infinite horizon cost. The state information x_j is sensed from its neighbors in the learning graph, and $u_j^\top R_j u_j$ is obtained from its neighbors via communications. Note that $u_{\mathcal{N}_L^i}$ may involve state information of agents in $\mathcal{V} \setminus \mathcal{N}_L^i$. However, it is not necessary for agent i to have such state information. Instead, it obtains $u_j^\top R_j u_j$ by communicating with agent $j \in \mathcal{N}_L^i \setminus \{i\}$.

Next we verify the validity of Assumption 2. Let $H_i(K, x(0)) = \sum_{t=0}^\infty \gamma^t x^\top(t)(\hat{Q}_i + K^\top \hat{R}K)x(t)$. The next proposition shows that the local cost functions we constructed satisfy Assumption 2.

Proposition 1: The local cost $J_i(K)$ constructed in (46) has the following properties.

(i). There exists a scalar $c_{lqr} > 0$ such that $H_i(K, x(0)) \le c_{lqr} J_i(K)$ for any $K \in \mathbb{K}_s$ and $x(0) \sim \mathcal{D}$.

(ii). $\nabla_{K_i} J_i(K) = \nabla_{K_i} J(K)$ for any $K \in \mathbb{K}_s$.

C. Distributed RL Algorithms for Multi-Agent LQR

Due to Proposition 1, we are able to apply Algorithms 2 to the distributed multi-agent LQR problem. Let $\mathbf{K}_i = \text{vec}(\tilde{K}_i) \in \mathbb{R}^{q_i}$, where $\tilde{K}_i$ is defined in (41), $q_i = mn_i = mn|\mathcal{N}_S^i|$, $i = 1, ..., N$. From the degree sum formula, we know

$$
\sum_{i=1}^N q_i = mn\sum_{i=1}^N |\mathcal{N}_S^i| = 2mn|\mathcal{E}_S| \triangleq q.
$$

Define $\mathbf{K} = (\mathbf{K}_1^\top, ..., \mathbf{K}_N^\top)^\top \in \mathbb{R}^q$. There must uniquely exist a control gain matrix $K \in \mathbb{R}^{mN \times nN}$ corresponding to $\mathbf{K}$. Let $\mathcal{M}_K(\cdot) : \mathbb{R}^q \to \mathbb{K}_d$ be the mapping transforming $\mathbf{K}$ to a distributed stabilizing gain.

To apply Algorithm 2 to the multi-agent LQR problem, we need to implement Algorithm 1 based on the learning graph $\mathcal{G}_L$ to achieve a clustering $\{\mathcal{V}_j, j = 1, ..., s\}$. Then the asynchronous RL algorithm for problem (37) is given in Algorithm 3. Note that the algorithm requires a stabilizing distributed control gain as the initial policy. One way to achieve this is making each agent learn a policy stabilizing itself, which has been studied in [46].

For the convenience of the readers to understand Algorithm 3, we present Table I to show the exact correspondence between the stochastic optimization (SO) problem in Section III and the LQR problem in this section. In Table I, $H(K, x(0)) = \sum_{t=0}^\infty \gamma^t x^\top(t)(Q + K^\top RK)x(t)$. From Table I and the distributed learning diagram in Fig. 1, we observe that the distributed learning nature of Algorithm 3 is reflected by the learning graph $\mathcal{G}_L$. Specifically, each agent learns its policy based on its own estimated cost value and information from its neighbors in graph $\mathcal{G}_L$.

TABLE I
THE CORRESPONDENCE BETWEEN THE SO PROBLEM AND THE LQR PROBLEM.

Problem	Variables					
SO	x	ξ	$h(x, \xi)$	$f(x)$	T	$\mathcal{G}$
LQR	$\mathbf{K}$	$x(0)$	$H(K, x(0))$	$J(K)$	T_K	$\mathcal{G}_L$

D. Convergence Analysis

In this subsection, we show the convergence result of Algorithm 3. Throughout this subsection, we adopt $J_i(K)$ in (46) as the local cost function for agent i. Let $Q_i^K = \hat{Q}_i + K^\top \hat{R}_i K$, $J_i^{T_J}(K) = \mathbb{E}[\sum_{t=0}^{T_J-1} x^\top(t)Q_i^K x(t)]$, $\mathbf{G} = (\mathbf{G}_1^\top, ..., \mathbf{G}_N^\top)^\top$, $\mathbf{G}_i(k) = \frac{q_i}{r_i}J_i(k)\mathbf{D}_i(k)$, $\mathbf{G}^{T_J} = ((\mathbf{G}_1^{T_J})^\top, ..., (\mathbf{G}_N^{T_J})^\top)^\top$, where $\mathbf{G}_i$ and $\mathbf{G}_i^{T_J}$ are the ideal and the actual estimates, respectively, of the gradient of the local cost function for agent i.

The essential difference between Algorithm 2 and Algorithm 3 is that the computation of the cost function in an LQR problem is inaccurate because (47) only provides a finite horizon cost, whereas the cost in (46) without expectation is with infinite horizon. This means that we need to take into account the estimation error of each local cost.

Algorithm 3 Asynchronous Distributed Learning for Multi-Agent LQR

Input: Step-size η, smoothing radius r_i and variable dimension q_i, $i = 1, ..., N$, clusters $\mathcal{V}_j$, $j = 1, ..., s$, iteration numbers T_K and T_J for controller variable and the cost, respectively, update sequence z_k and extrapolation weight w_i^k, $k = 0, ..., T_K - 1$, $\mathbf{K}^0$ such that $\mathcal{M}_K(\mathbf{K}^0) \in \mathbb{K}_s$.

Output: $\mathbf{K}^*$.

1. **for** $k = 0, 1, ..., T_K - 1$ **do**
2.   Sample $x_0 \sim \mathcal{D}$. Set $x(0) = x_0$.
3.   **for** $i \in \mathcal{V}$ **do**
4.   **if** $i \in \mathcal{V}_{z_k}$ **do** (Simultaneous Implementation)
5.   Agent i samples $\mathbf{D}_i(k) \in \mathbb{R}^{q_i}$ randomly from $\mathbb{S}_{q_i - 1}$, and computes $\hat{\mathbf{K}}_i^k = \mathbf{K}_i^k + w_i^k(\mathbf{K}_i^k - \mathbf{K}_i^{k_{prev}})$.
6.   Agent i implements its controller

$$u_i(t, k) = -\text{vec}^{-1}(\hat{\mathbf{K}}_i^k + r_i\mathbf{D}_i(k))x_{\mathcal{N}_S^i}(t),$$

             while each agent $j \in \mathcal{V} \setminus \mathcal{V}_{z_k}$ implements $u_j(t, k) = -\text{vec}^{-1}(\mathbf{K}_j^k)x_{\mathcal{N}_S^j}(t)$ for $t = 0, ..., T_J - 1$, and observes

$$H_{i,T_J}(K^{i,k}, x(0)) = \sum_{t=0}^{T_J - 1} \gamma^t \left[x_{\mathcal{N}_L^i}^\top(t)\bar{Q}_i x_{\mathcal{N}_L^i}(t) + u_{\mathcal{N}_L^i}^\top(t, k)\bar{R}_i u_{\mathcal{N}_L^i}(t, k) \right]. \tag{47}$$

7.   Agent i computes the estimated gradient:

$$\mathbf{G}_i^{T_J}(k) = \frac{q_i}{r_i}H_{i,T_J}(K^{i,k}, x(0))\mathbf{D}_i(k),$$

             then updates its policy:

$$\mathbf{K}_i^{k+1} = \hat{\mathbf{K}}_i^k - \eta\mathbf{G}_i^{T_J}(k). \tag{48}$$

8.   **else**

$$\mathbf{K}_i^{k+1} = \mathbf{K}_i^k. \tag{49}$$

9.   **end**
10.   **end**
11. **end**
12. Set $\mathbf{K}^* = \mathbf{K}^{k+1}$.

Let $H_i(K, x(0)) = \sum_{t=0}^\infty \gamma^t(x_{\mathcal{N}_L^i}^\top \bar{Q}_i x_{\mathcal{N}_L^i} + u_{\mathcal{N}_L^i}^\top \bar{R}_i u_{\mathcal{N}_L^i})$, whose expectation is $J_i(K)$ in (46). Then H_i corresponds to h_i in Algorithm 2. Therefore, the ideal and actual gradient estimates for agent i at step k are $\mathbf{G}_i(k) = \frac{q_i}{r_i}H_i(K^{i,k}, x(0))\mathbf{D}_i(k)$, and $\mathbf{G}_i^{T_J}(k) = \frac{q_i}{r_i}H_{i,T_J}(K^{i,k}, x(0))\mathbf{D}_i(k)$, respectively, where $K^{i,k} = \mathcal{M}_K(\mathbf{K}^{i,k})$, $\mathbf{K}^{i,k} = (\mathbf{K}_1^{k\top}, ..., (\hat{\mathbf{K}}_i^k + r_i\mathbf{D}_i(k))^\top, ..., \mathbf{K}_N^{k\top})^\top$. Note that for any $\mathbf{K} \in \mathbb{R}^q$, it holds that $\|\mathbf{K}\| = \|\mathcal{M}_K(\mathbf{K})\|_F$.

We study the convergence by focusing on $K \in \mathbb{K}_\alpha$, where $\mathbb{K}_\alpha$ is defined as

$$\mathbb{K}_\alpha = \{K \in \mathbb{K}_d : J(K) \leq \alpha J(K^0), J_i(K) \leq \alpha_i J_i(K^0)\} \subseteq \mathbb{K}_s, \tag{50}$$

where $\alpha, \alpha_i > 1$, $K^0 \in \mathbb{K}_d \cap \mathbb{K}_s$ is the given initial stabilizing control gain. Then $\mathbb{K}_\alpha$ is compact due to coerciveness of $J(K)$ (as shown in Lemma 4).

Due to the continuity of ϕ_K, λ_K, β_K and ζ_K, and the compactness of $\mathbb{K}_\alpha$, we define the following parameters for $K \in \mathbb{K}_\alpha$:

$$\phi_0 = \sup_{K \in \mathbb{K}_\alpha} \phi_K, \quad \lambda_0 = \sup_{K \in \mathbb{K}_\alpha} \lambda_K, \quad \rho_0 = \inf_{K \in \mathbb{K}_\alpha}\{\beta_K, \zeta_K\}, \quad \kappa_0 = \sup_{K \in \mathbb{K}_\alpha} \|\sqrt{\gamma}(\mathcal{A} - \mathcal{B}K^{i,k})\|, \tag{51}$$

where $\kappa_0 \in (0, 1)$.

The following lemma evaluates the gradient estimation in the LQR problem.

Lemma 5: Given $\epsilon' > 0$ and $K^k \in \mathbb{K}_\alpha$, if $r_i \leq \frac{\rho_0}{2}$, $w_i^k \leq \frac{\rho_0}{2\|K_i^k - K_i^{k_{prev}}\|_F}$ and

$$T_J \geq \max_{i \in \mathcal{V}} \frac{1}{2(1 - \kappa_0)} \log\left(\frac{\alpha_i J_i(K^0)\lambda_{\max}(x(0)x^\top(0))}{\lambda_{\min}(\Sigma_x)\epsilon'}\right), \tag{52}$$

then

$$H_i(K^{i,k}, x(0)) - H_{i,T_J}(K^{i,k}, x(0)) \leq \epsilon', \quad i \in \mathcal{V}, \tag{53}$$

$$\|\mathbb{E}[\mathbf{G}_i^{T_J}(k)] - \nabla_{\mathbf{K}_i}J(K^{i,k})\| \leq \frac{q_i\epsilon'}{r_i} + \phi_0 r_i, \tag{54}$$

$$\|G_i^{T_J}(k)\| \leq \frac{q_i}{r_i}\left[c_{lqr}(\alpha_i J_i(K^0) + \lambda_0\rho_0) + \epsilon'\right], \tag{55}$$

where J_{i,T_J} is in (47).

Let $J_0(K^0) = \max_{i \in \mathcal{V}} \alpha_i J_i(K^0)$. Based on Lemma 5 and Theorem 1, we have the following convergence result for Algorithm 3.

Theorem 2: Under Assumption 4, given positive scalars ϵ, ν, γ, and $\alpha \geq 2 + \gamma + \frac{1}{\nu} + \nu\gamma$, $K^0 \in \mathbb{K}_s \cap \mathbb{K}_d$. Let $\{\mathbf{K}^k\}_{k=0}^{T_K - 1}$ be the sequence of states obtained by implementing Algorithm 3 for $k = 0, ..., T_K - 1$. Suppose that T_J satisfies (52) for each k (T_J can be different for different steps), and

$$
T_K = \lceil \frac{2\alpha\nu J(\mathbf{K}^0)}{\eta\epsilon} \rceil, \quad \eta \leq \min\{\frac{\rho_0}{2\delta\sqrt{N_0}}, \frac{2\alpha J(\mathbf{K}^0)}{\gamma\epsilon}, \frac{\gamma\epsilon}{2\alpha N_0(\phi_0\delta_2^2 + 4\phi_0^2 + \phi_0 + 4)}\}, \quad \epsilon' \leq \frac{r_-}{4q_+}\sqrt{\frac{\gamma\epsilon}{\alpha N_0}},
$$
$$
w_i^k \leq \frac{1}{\|\mathbf{K}_i^k - \mathbf{K}_i^{k_{prev}}\|} \min\{\eta^{3/2}, \frac{\rho_0}{2\sqrt{N_i}}\}, \quad r_i \leq \min\{\frac{\rho_0}{2}, \frac{1}{4\phi_0}\sqrt{\frac{\gamma\epsilon}{\alpha N_0}}\}, i \in \mathcal{V},
$$
(56)

where $\delta = \frac{q_+}{r_-}\left[c_{lqr}(\alpha_i J_0(K^0) + \lambda_0\rho_0) + \epsilon'\right]$.

(i). The following holds with a probability at least $1 - \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma)$:

$$
\frac{1}{T_K} \sum_{k=0}^{T_K - 1} \sum_{i \in \mathcal{V}_{z_k}} \|\nabla_{\mathbf{K}_i} J(\mathbf{K}^k)\|^2 < \epsilon.
$$
(57)

(ii). Under Assumption 3, suppose that there exists some $\bar{\epsilon} \in (0, \rho_0]$ such that

$$
w_i^k \leq \frac{\bar{\epsilon}}{2(T_0 - 1)N_i\|\mathbf{K}_i^k - \mathbf{K}_i^{k_{prev}}\|}, \quad \eta \leq \frac{\bar{\epsilon}}{2\delta(T_0 - 1)N_0},
$$
(58)

for any step $k \in \{0, ..., T_K - 1\}$, and conditions in (56) are satisfied, then the following holds with a probability at least $1 - \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma)$:

$$
\frac{1}{T_K} \sum_{k=0}^{T_K - 1} \|\nabla_{\mathbf{K}} J(\mathbf{K}^k)\|^2 < \hat{\epsilon}.
$$
(59)

where $\hat{\epsilon} = 2T_0(\epsilon + \phi_0^2\bar{\epsilon}^2)$.

From Theorem 2 (ii), the sample complexity for convergence of Algorithm 3 is $T_K T_J = \mathcal{O}(q_+^2 N_0^2 \log(q_+/\hat{\epsilon})/\hat{\epsilon}^3)$, which is higher than that of Algorithm 2 because of the error on the local cost function evaluation in the LQR problem. The sample complexity here has a lower order on convergence accuracy than that in [18]. Compared with [18], our algorithm has two more advantages: (i) the sample complexity in [18] is affected by the convergence rate of the consensus algorithm, the number of agents, and the dimension of the entire state variable, while the sample complexity of our algorithm depends on the local optimization problem for each agent (the number of agents in one cluster and the dimension of the variable for one agent), therefore rendering high scalability of our algorithm to large-scale networks; (ii) the algorithm in [18] requires each agent to estimate the global cost during each iteration, while our algorithm is based on local cost evaluation, which benefits for variance reduction and privacy preservation.

V. SIMULATION EXPERIMENTS

A. Optimal Tracking of Multi-Robot Formation

In this section, we apply Algorithm 3 to a multi-agent formation control problem. Consider a group of $N = 10$ robots modeled by the following double integrator dynamics:

$$
\begin{aligned}
r_i(t+1) &= r_i(t) + v_i(t), \\
v_i(t+1) &= v_i(t) + C_i u_i(t), \quad i = 1, ..., 10,
\end{aligned}
$$
(60)

where $r_i, v_i, u_i \in \mathbb{R}^2$ are position, velocity, and control input of agent i, respectively, $C_i \in \mathbb{R}^{2 \times 2}$ is a coupling matrix in the dynamics of agent i. Let $x_i = (r_i^\top, v_i^\top)^\top$, $A_i = \begin{pmatrix} I_2 & I_2 \\ \mathbf{0}_{2 \times 2} & I_2 \end{pmatrix}$, $B_i = (\mathbf{0}_{2 \times 2}, C_i^\top)^\top$, the dynamics (60) can be rewritten as

$$
x_i(t+1) = A_i x_i(t) + B_i u_i(t).
$$
(61)

The control objective is to make the robots learn their own optimal controllers for the whole group to form a circular formation, track a moving target, maintain the formation as close as possible to the circular formation during the tracking process, and cost the minimum control energy. The target has the following dynamics:

$$
r_0(t+1) = r_0(t) + v_0,
$$
(62)

where the velocity $v_0 \in \mathbb{R}^2$ is fixed. Let $x_0 = (r_0^\top, v_0^\top)^\top$ be the state vector of the target, and $d_i(t) = x_0(t) + (\cos\theta_i, \sin\theta_i, 0, 0)^\top$ with $\theta_i = \frac{2\pi i}{N}$ be the desired time-varying state of robot i. Suppose that the initial state x_i of each robot i is a random variable
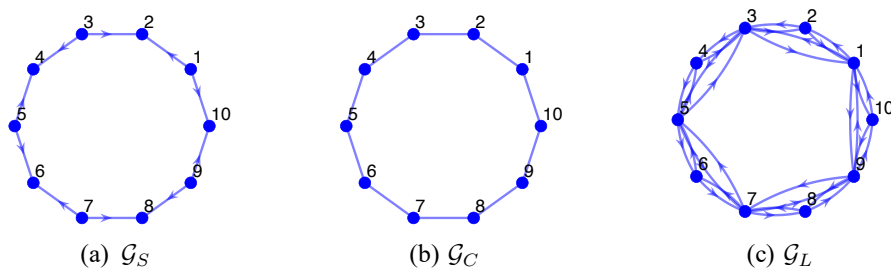
Fig. 4. The sensing graph $\mathcal{G}_S$, cost graph $\mathcal{G}_C$ and the resulting learning graph $\mathcal{G}_L$ for the formation control problem.

with mean d_i, which implies that each agent is randomly perturbed from its desired state. Then the objective function to be minimized can be written as

$$J = \mathbb{E}_{(x(0)-d)\sim\mathcal{D}} \left[\sum_{t=0}^{\infty} \left(\sum_{(i,j)\in\mathcal{E}_C} \|x_i(t) - x_j(t) - (d_i - d_j)\|^2 + \sum_{i\in\mathcal{V}_r} \|x_i - d_i\|^2 + \sum_{i=1}^{N} \|u_i(t)\|^2 \right) \right], \quad (63)$$

where $\mathcal{V}_r$ is the leader set consisting of robots that do not sense information from others but only chase for its target trajectory. In the literature of formation control, usually the sensing graph via which each robot senses its neighbors' state information is required to have a spanning tree. Here we assume that the sensing graph has a spanning forest from the leader set $\mathcal{V}_L$. Different from a pure formation control problem, the main goal of the LQR formulation is to optimize the transient relative positions between different robots and minimize the required control effort of each agent. Fig. 4 shows the sensing graph, the cost graph and the resulting learning graph for the formation control problem. The leader set is $\{1, 3, 5, 7, 9\}$.

Define the new state variable $y_i = x_i - d_i$ for each robot i. Using the property $A_i d_i(t) = d_i(t+1)$, we obtain dynamics of y_i as $y_i(t+1) = A_i y_i(t) + B_i u_i(t)$. The objective becomes

$$\min_u J = \mathbb{E}_{y(0)\sim\mathcal{D}} \left[\sum_{t=0}^{\infty} \gamma^t (y^\top(t)(L + \Lambda)y(t) + u^\top(t)u(t)) \right], \quad (64)$$

where $y = (y_1^\top, ..., y_N^\top)^\top$, L is the Laplacian matrix corresponding to the cost graph, Λ is a diagonal matrix with $\Lambda_{ii} = 1$ if $i \in \mathcal{V}_r$ and $\Lambda_{ii} = 0$ otherwise.

Let $C_i = \frac{i}{i+1}I_2$, the initial stabilizing gain is given by $\mathbf{K}(0) = \mathcal{M}_K^{-1}(K_0)$, where $K_0 = I_N \otimes \tilde{K}$ and $\tilde{K} = (I_2, 1.5I_2)$, which is stabilizing for each robot. By implementing Algorithm 1 based on the learning graph in Fig. 4 (c), the following clustering is obtained :

$$\mathcal{V}_1 = \{1, 4, 6, 8\}, \mathcal{V}_2 = \{2, 5, 9\}, \mathcal{V}_3 = \{3, 7, 10\}, \quad (65)$$

which means that there are three clusters update asynchronously, while agents in each cluster update their variable states via independent local cost evaluations at one step. It will be shown that compared with the traditional BCD algorithm where only one block coordinate is updated at one step, our algorithm is more efficient.

B. Simulation Results

We compare our algorithm with the centralized one-point ZOO algorithm in [5], where $\mathbf{K}$ is updated by taking an average of multiple repeated global cost evaluations. Fig. 5 shows the simulation results, where the performance is evaluated with given initial states of all the robots. When implementing the algorithms, each component of the initial states are randomly generated from a truncated normal distribution $\mathcal{N}(0, 1)$. Moreover, we set $\eta = 10^{-6}$, $r = 1$, $T_K = 1000$, $T_J = 50$ for both Algorithm 3 and the centralized algorithm. In Fig. 5, we show the performance trajectory of the controller generated by different algorithms. When $s = 10$, Algorithm 3 is a BCD algorithm without clustering, i.e., each cluster only contains one robot, while $s = 3$ corresponds to the clustering strategy in (65). The extrapolation weight w_i^k for each agent i at each step k is uniformly set. So $w_i^k = 0$ implies that Algorithm 3 is not accelerated. To compare the variance of the controller performance for different algorithms, we conduct gradient estimation for 50 times in each iteration, and plot the range of performance induced by all the estimated gradients, as shown in the shaded areas of Fig. 5 (a). Each solid trajectory in Fig. 5 corresponds to a controller updated by using an average of the estimated gradients for the algorithm. Fig. 5 (b) shows the trajectories of agents by implementing the convergent controller generated by Algorithm 3 with $s = 3$, and $w_i^k = 0.5$, $i \in \mathcal{V}$, $k = 0, ..., T_K - 1$. From the simulation results, we have the following observations:

- Algorithm 3 always converges faster and has a lower performance variance than the centralized zeroth-order algorithm.
- Algorithm 3 with the cluster-wise update scheme converges faster than it with the agent-wise update scheme. This is because the number of clusters dominates the sample complexity, as we analyzed below Theorem 2.
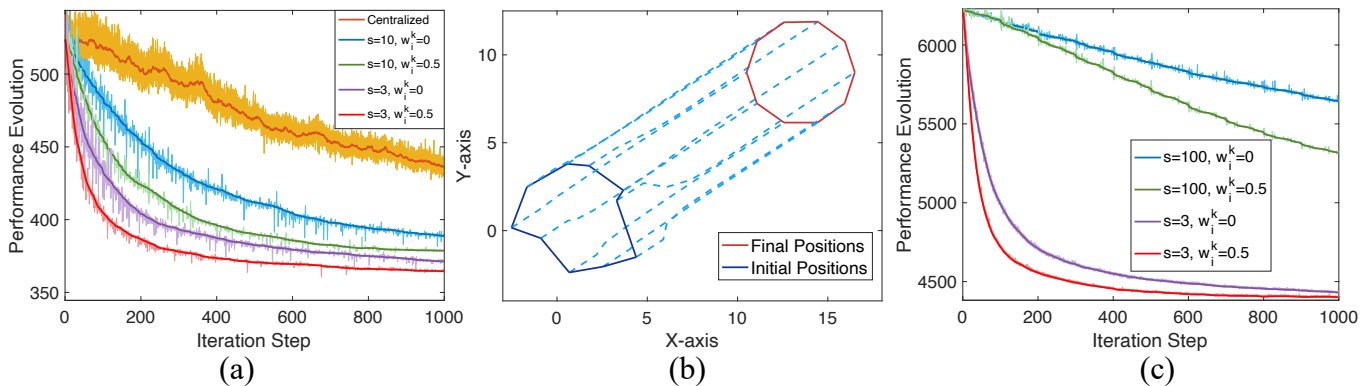
Fig. 5. (a) The group performance evolution of a 10-agent formation under the centralized ZOO algorithm, Algorithm 3 without clustering and acceleration, without clustering but with acceleration, with clustering but without acceleration, and with both clustering and acceleration. The shaded areas denote the performance corresponding to the controllers obtained by perturbing the current control gain with 50 random samplings. (b) The trajectories of robots under the controller learned by Algorithm 3 with $s = 3$, $w_i^k = 0.5$. (c) The performance evolution of a 100-agent formation under Algorithm 3 with different parameters.

- Appropriate extrapolation weights not only accelerate Algorithm 3, but also decrease the performance variance.

Note that the performance of the optimal centralized controller is 345.271. The reason why Algorithm 3 converges to a controller with a cost value larger than the optimal one in Fig. 5 (a) is that the distributed LQR problem has a structure constraint for the control gain, which results in a feasible set $\mathbb{K}_s \cap \mathbb{K}_d$, whereas the feasible set of the centralized LQR problem is $\mathbb{K}_s$.

Scalability to Large-Scale Networks. Even if we further increase the number of robots, the local cost of each robot always involves only 5 robots as long as the structures of the sensing and cost graphs are maintained. That is, the magnitude of each local cost does not change as the network scale grows. On the contrary, for global cost evaluation, the problem size severely influences the estimation variance. The difference of variances between these two methods has been analyzed in Subsection III-E. To show the advantage of our algorithm, we further deal with a case with 100 robots, where the cost, sensing and learning graphs have the same structure as the 10-robot case, and implementing Algorithm 1 still results in 3 clusters. Simulation results show that the centralized algorithm failed to solve the problem. The performance trajectories of Algorithm 3 with different settings are shown in Fig. 5 (c), from which we observe that Algorithm 3 has an excellent performance, and both clustering and extrapolation are efficient in improving the convergence rate. Note that the performance of the centralized optimal control gain for the 100-agent case is 4119.87.

VI. CONCLUSION

We have proposed a novel distributed RL (zeroth-order accelerated BCD) algorithm with asynchronous sample and update schemes. The algorithm was applied to distributed learning of the model-free distributed LQR by designing the specific local cost functions and the interaction graph required by learning. A simulation example of formation control has shown that our algorithm has significantly lower variance and faster convergence rate compared with the centralized ZOO algorithm. In the future, we will look into how to reduce the sample complexity of the proposed distributed ZOO algorithm.

VII. APPENDIX

A. Analysis for the Asynchronous RL Algorithm

Proof of Lemma 1: It suffices to show that

$$\mathbb{E}_{u_i^l \in \mathbb{S}_{n-1}}[f_i(x_i + r_i u_i^l, x_{-i})u_i^l] = \frac{r_i}{q_i}\nabla_{x_i}\hat{f}_i(x), \tag{66}$$

for $i = 1, ..., N$. This has been proved in [3, Lemma 1]. ∎

Proof of Lemma 2: Denote $v^i = (0, ..., 0, v_i^\top, 0, ..., 0)^\top \in \mathbb{R}^q$ with $v_i \in \mathbb{R}^{q_i}$. The following holds:

$$\begin{aligned}
&\|\nabla_{x_i}\hat{f}_i(x) - \nabla_{x_i}f(x)\| \\
&= \|\nabla_{x_i}\mathbb{E}_{v_i \in \mathbb{B}_{q_i}}[f_i(x_i + r_i v_i, x_{-i})] - \nabla_{x_i}f(x)\| \\
&= \|\mathbb{E}_{v_i \in \mathbb{B}_{q_i}}[\nabla_{x_i}f_i(x_i + r_i v_i, x_{-i}) - \nabla_{x_i}f(x)]\| \\
&\leq \mathbb{E}_{v_i \in \mathbb{B}_{q_i}}\|\nabla_{x_i}f_i(x_i + r_i v_i, x_{-i}) - \nabla_{x_i}f(x)\| \\
&= \mathbb{E}_{v^i \in \mathbb{B}_q}\|\nabla_{x_i}f(x + r_i v^i) - \nabla_{x_i}f(x)\| \\
&\leq \mathbb{E}_{v^i \in \mathbb{B}_q}\|\nabla_x f(x + r_i v^i) - \nabla_x f(x)\| \\
&\leq \phi_x r_i,
\end{aligned} \tag{67}$$

where the second equality follows from the smoothness of $f_i(x)$, and the first inequality used Jensen's inequality. ∎

Before proving Theorem 1, we show that once x^k is restricted in $\mathbb{X}$, we are able to give a uniform bound on the estimated gradient $g_i(\hat{x}_i, x^k_{-i}, u, \xi)$ at each step, see the following lemma.

Lemma 6: If $r_i \leq \rho_0/2$ and $w^k_i \leq \frac{\rho_0}{2\|x^k_i - x^{prev}_i\|}$, then for any $x^k \in \mathbb{X}$, the estimated gradient satisfies

$$\|g_i(\hat{x}^k_i, x^k_{-i}, u_i, \xi)\| \leq \frac{q_i}{r_i} c \left[\alpha_i f_i(x^0) + \lambda_0 \rho_0\right]. \tag{68}$$

Proof: Using the definition of $g_i(x, u_i, \xi)$ in (16), we have

$$
\begin{aligned}
\|g_i(\hat{x}^k_i, x^k_{-i}, u_i, \xi)\| &= \frac{q_i}{r_i} h_i(\hat{x}^k_i + r_i u_i, x^k_{-i}, \xi) \\
&\leq \frac{q_i}{r_i} c f_i(\hat{x}^k_i + r_i u_i, x^k_{-i}) \\
&\leq \frac{q_i}{r_i} c \left[f_i(x^k) + \lambda_0 \rho_0\right] \\
&\leq \frac{q_i}{r_i} c \left[\alpha_i f_i(x^0) + \lambda_0 \rho_0\right],
\end{aligned}
\tag{69}
$$

where the first inequality used Assumption 2, the second inequality used the local Lipschitz continuity of $f(x)$ over $\mathbb{X}$ because

$$\|\hat{x}^k_i + r_i u_i - x^k_i\| \leq \|\hat{x}^k_i - x^k_i\| + r_i = w^k_i\|x^k_i - x^{k_{prev}}_i\| + r_i \leq \rho_0, \tag{70}$$

and the last inequality holds because $x^k \in \mathbb{X}$. ∎

Proof of Theorem 1: To facilitate the proof, we give some notations first. Let $\mathcal{K}^k_i = \{j : i \in \mathcal{V}_{z_j}, j \leq k\}$ be the set of iterations that x_i is updated before step k, $d^k_i = |\mathcal{K}^k_i|$ be the number of times that x_i has been updated until step k. Let $\tilde{x}^j_i$ be the value of x_i after it is updated j times, then $x^k_i = \tilde{x}^{d^k_i}_i$. We use $g_i(\hat{x}^k_i, x^k_{-i})$ as the shorthand of $g_i(\hat{x}^k_i, x^k_{-i}, u, \xi)$. Let $\mathcal{F}_k$ denote the σ-field containing all the randomness in the first k iterations. Moreover, we use the shorthand $\mathbb{E}^k[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_k]$.

(i). Suppose $x_k \in \mathbb{X}$. Since $f(x)$ has a (ϕ_0, ρ_0) Lipschitz continuous gradient at x_k, and

$$
\begin{aligned}
\|x^{k+1} - x^k\|^2 &= \sum_{i \in \mathcal{V}_{z_k}} \|x^{k+1}_i - x^k_i\|^2 \\
&\leq \sum_{i \in \mathcal{V}_{z_k}} \left[w^k_i(x^k_i - x^{k_{prev}}_i) + \eta\|g_i(\hat{x}^k_i, x^k_{-i})\|\right]^2 \\
&\leq N_i\left(\frac{\rho_0}{2\sqrt{N_i}} + \frac{\rho_0}{2\sqrt{N_0}}\right)^2 \leq \rho_0^2,
\end{aligned}
\tag{71}
$$

it holds that

$$
\begin{aligned}
f(x^{k+1}) - f(x^k) &\leq \sum_{i \in \mathcal{V}_{z_k}} \left[\langle \nabla_{x_i} f(x^k), x^{k+1}_i - x^k_i\rangle + \frac{\phi_0}{2}\|x^{k+1}_i - x^k_i\|^2\right] \\
&\leq \sum_{i \in \mathcal{V}_{z_k}} \left[\langle \nabla_{x_i} f_i(x^k), w^k_i(x^k_i - x^{k_{prev}}_i) - \eta g_i(\hat{x}^k_i)\rangle + \frac{\phi_0}{2}\eta^2\|g_i(\hat{x}^k_i, x^k_{-i})\|^2 + \frac{\phi_0}{2}(w^k_i)^2\|x^k_i - x^{k_{prev}}_i\|^2\right] \\
&= \sum_{i \in \mathcal{V}_{z_k}} \left[-\eta\|\nabla_{x_i} f_i(x^k)\|^2 + \frac{\phi_0}{2}\eta^2\|g_i(\hat{x}^k_i, x^k_{-i})\|^2 + \eta\Delta^k_i + \Theta^k_i\right],
\end{aligned}
\tag{72}
$$

where $\Delta^k_i = \|\nabla_{x_i} f_i(x^k)\|^2 - \langle \nabla_{x_i} f_i(x^k), g_i(\hat{x}^k_i, x^k_{-i})\rangle$, $\Theta^k_i = \langle \nabla_{x_i} f(x^k), w^k_i(x^k_i - x^{k_{prev}}_i)\rangle + \frac{\phi_0}{2}(w^k_i)^2\|x^k_i - x^{k_{prev}}_i\|^2$.

Define the first iteration step at which x escapes from $\mathbb{X}$ below:

$$\tau = \min\{k : x(k) \notin \mathbb{X}\}. \tag{73}$$

Next we analyze $\mathbb{E}^k\left[(f(x^k) - f(x^{k+1}))1_{\tau > k}\right]$. Under the condition $\tau > k$, both $\|\nabla_{x_i} f_i(\hat{x}^k_i, x^k_{-i}) - \mathbb{E}^k[g_i(\hat{x}^k_i, x^k_{-i})]\|$ and $\|g_i(\hat{x}^k_i, x^k_{-i})\|$ are uniformly bounded. For notation simplicity in the rest of the proof, according to Lemma 2 and Lemma 6, we adopt $\theta = \phi_0 \max_{i \in \mathcal{V}} r_i$ as the uniform upper bound of $\|\nabla_{x_i} f_i(\hat{x}^k_i, x^k_{-i}) - \mathbb{E}^k[g_i(\hat{x}^k_i, x^k_{-i})]\|$, and $\delta = \frac{q_+}{r_-} c \left[\alpha f_0(x^0) + \lambda_0 \rho_0\right]$ as the uniform upper bound of $\|g_i(\hat{x}^k_i, x^k_{-i})\|$.

According to Assumption 2, $\nabla_{x_i} f_i(x)$ is (ϕ_0, ρ_0) locally Lipschitz continuous. Then we are able to bound $\mathbb{E}^k[\Delta_i^k]$ if $\tau > k$:

$$
\begin{aligned}
\mathbb{E}^k[\Delta_i^k] &= \langle \nabla_{x_i} f_i(x^k), \nabla_{x_i} f_i(x^k) - \nabla_{x_i} f_i(\hat{x}_i^k, x_{-i}^k) \rangle + \langle \nabla_{x_i} f_i(x^k), \nabla_{x_i} f_i(\hat{x}_i^k, x_{-i}^k) - \mathbb{E}^k[g_i(\hat{x}_i^k, x_{-i}^k)] \rangle \\
&\leq \|\nabla_{x_i} f_i(x^k)\| \phi_0 \|x_i^k - \hat{x}_i^k\| + \theta \|\nabla_{x_i} f_i(x^k)\| \\
&= w_i^k \phi_0 \|\nabla_{x_i} f_i(x^k)\| \|\tilde{x}_i^{d_i^k} - \tilde{x}_i^{d_i^k - 1}\| + \theta \|\nabla_{x_i} f_i(x^k)\| \\
&\leq \frac{1}{8} \|\nabla_{x_i} f_i(x^k)\|^2 + 2(w_i^k)^2 \phi_0^2 \|\tilde{x}_i^{d_i^k} - \tilde{x}_i^{d_i^k - 1}\|^2 + \theta \|\nabla_{x_i} f_i(x^k)\| \\
&\leq \frac{3}{8} \|\nabla_{x_i} f_i(x^k)\|^2 + 2\phi_0^2 \eta^3 + \theta^2,
\end{aligned}
\tag{74}
$$

where the first inequality used $\|x_i^k - \hat{x}_i^k\| = \|w_i^k(x_i^k - x_i^{k_{prev}})\| \leq \rho_0$, and the Lipschitz continuity of $\nabla_{x_i} f(x_i^k, x_{-i}^k)$, the second equality used (13), the second and the last inequalities used Young's inequality, and $w_i^k \leq \eta^{3/2} / \|x_i^k - x_i^{k_{prev}}\|$.

Similarly, we bound $\mathbb{E}^k[\Theta_i^k]$ for $\tau > k$ as follows.

$$
\mathbb{E}^k[\Theta_i^k] \leq \frac{1}{8} \eta \|\nabla_{x_i} f_i(x^k)\|^2 + 2\eta^2 + \frac{\phi_0}{2} \eta^3,
\tag{75}
$$

where we used $w_i^k \leq \frac{\eta^{3/2}}{\|x_i^k - x_i^{k_{prev}}\|}$.

Combining the inequalities (72), (74) and (75), we have

$$
\mathbb{E}^k[(f(x^{k+1}) - f(x^k))1_{\tau > k}] \leq -\frac{1}{2} \sum_{i \in \mathcal{V}_{z_k}} \eta \mathbb{E}^k \left[\|\nabla_{x_i} f(x^k)\|^2 1_{\tau > k} \right] + \frac{\eta Z}{2},
\tag{76}
$$

where $Z = N_0(\phi_0 \delta^2 \eta + 4\phi_0^2 \eta^3 + 2\theta^2 + 4\eta + \phi_0 \eta^2)$.

Note that

$$
\mathbb{E}^k \left[f(x^{k+1}) 1_{\tau > k+1} \right] \leq \mathbb{E}^k \left[f(x^{k+1}) 1_{\tau > k} \right] = \mathbb{E}^k \left[f(x^{k+1}) \right] 1_{\tau > k},
\tag{77}
$$

where the equality holds because all the randomness before the $(k+1)$th iteration has been considered in $\mathbb{E}^k[\cdot]$. Then $1_{\tau > k}$ is determined. It follows that

$$
\mathbb{E}^k \left[(f(x^k) - f(x^{k+1})) 1_{\tau > k} \right] \leq \mathbb{E}^k \left[f(x^k) 1_{\tau > k} - f(x^{k+1}) 1_{\tau > k+1} \right].
\tag{78}
$$

Summing (76) over k from 0 to $T - 1$ and utilizing (78) yields

$$
\mathbb{E}^k[f(x^\top) 1_{\tau > T}] - f(x^0) \leq -\frac{1}{2} \sum_{k=0}^{T-1} \sum_{i \in \mathcal{V}_{z_k}} \eta \mathbb{E}^k \left[\|\nabla_{x_i} f(x^k)\|^2 \right] + T\eta Z/2.
\tag{79}
$$

It follows that

$$
\frac{1}{T} \sum_{k=0}^{T-1} \sum_{i \in \mathcal{V}_{z_k}} \mathbb{E}^k[\|\nabla_{x_i} f(x^k)\|^2] \leq \frac{2}{\eta T} f(x^0) + Z.
\tag{80}
$$

Now we analyze the probability $\mathbf{P}(\tau < T)$. Define the process

$$
Y(k) = f(x^{\min\{k, \tau\}}) + \frac{\eta}{2} (T - k) Z, \quad k = 0, ..., T - 1,
\tag{81}
$$

which is non-negative and almost surely bounded under the given conditions. Next we show $Y(k)$ is a supermartingale by discussing the following two cases:

Case 1, $\tau > k$.

$$
\begin{aligned}
\mathbb{E}^k[Y(k+1)] &= \mathbb{E}^k[f(x^{k+1})] + \frac{\eta}{2}(T - k - 1)Z \\
&= f(x^k) + \mathbb{E}^k[f(x^{k+1}) - f(x^k)] + \frac{\eta}{2}(T - k - 1)Z \\
&\leq f(x^k) + \frac{\eta}{2}(T - k)Z = Y(k),
\end{aligned}
\tag{82}
$$

where the inequality used (76).

Case 2, $\tau \leq k$.

$$
\mathbb{E}^k[Y(k+1)] = f(x^\tau) + \frac{\eta}{2}(T - k - 1)Z \leq f(x^\tau) + \frac{\eta}{2}(T - k)Z = Y(k).
\tag{83}
$$

Therefore, $Y(k)$ is a super-martingale. Invoking Doob's maximal inequality for super-martingales yields

$$
\begin{aligned}
\mathbf{P}(\tau < T) &\leq \mathbf{P}(\max_{k=0,\ldots,T-1} f(x^k) > \alpha f(x^0)) \\
&\leq \mathbf{P}(\max_{k=0,\ldots,T-1} Y(k) > \alpha f(x^0)) \\
&\leq \frac{\mathbb{E}^k[Y(0)]}{\alpha f(x^0)} = \frac{f(x^0) + \eta T Z/2}{\alpha f(x^0)} = \frac{2 + \nu\gamma}{\alpha}.
\end{aligned}
\tag{84}
$$

where the last equality used $\eta \leq \frac{2\alpha f(x^0)}{\gamma\epsilon}$, $T \leq \frac{2\alpha f(x^0)}{\epsilon\eta}\nu + 1$, and $Z \leq \gamma\epsilon/\alpha$. The upper bound for Z is obtained by noting that our conditions on r_i and η cause that $2\theta^2 \leq \frac{\gamma\epsilon}{2\alpha N_0}$ and $\phi_0\delta^2\eta + 4\phi_0^2\eta^3 + 4\eta + \phi_0\eta^2 \leq (\phi_0\delta_2^2 + 4\phi_0^2 + 4 + \phi_0)\eta \leq \frac{\gamma\epsilon}{2\alpha N_0}$ (we consider $\eta \leq 1$ by default).

It follows that

$$
\begin{aligned}
\mathbf{P}(\frac{1}{T}\sum_{k=0}^{T-1}\sum_{i\in\mathcal{V}_{z_k}}\|\nabla_{x_i}f(x^k)\|^2 \geq \epsilon) &\leq \mathbf{P}(\frac{1}{T}\sum_{k=0}^{T-1}\sum_{i\in\mathcal{V}_{z_k}}\|\nabla_{x_i}f(x^k)\|^2 1_{\tau\geq T} \geq \epsilon) + \mathbf{P}(\tau < T) \\
&\leq \frac{1}{\epsilon}\mathbb{E}^k\left[\frac{1}{T}\sum_{k=0}^{T-1}\sum_{i\in\mathcal{V}_{z_k}}\|\nabla_{x_i}f(x^k)\|^2 1_{\tau\geq T}\right] + \mathbf{P}(\tau < T) \\
&\leq \frac{1}{\alpha}(\gamma + \frac{1}{\nu}) + \frac{2+\nu\gamma}{\alpha} = \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma),
\end{aligned}
\tag{85}
$$

where the last inequality used (80), $T \geq \frac{2\alpha\nu f(x^0)}{\eta\epsilon}$ and $Z \leq \gamma\epsilon/\alpha$.

(ii). Without loss of generality, suppose that T is divisible by T_0 (if not, the convergence accuracy ϵ can be re-selected from its small neighborhood such that $T = \lceil\frac{2\alpha\nu f(x^0)}{\eta\epsilon}\rceil$ is divisible by T_0.). Let $M = \frac{T}{T_0}$. For any $K \in \{0,\ldots,M-1\}$, distinct steps $k, k' \in [KT_0, KT_0 + T_0 - 1]$ and $x^k, x^{k'} \in \mathbb{X}$, we have

$$
\begin{aligned}
\|x^k - x^{k'}\| &\leq \sum_{l=KT_0}^{KT_0+T_0-2}\|x^l - x^{l+1}\| \\
&\leq \sum_{l=KT_0}^{KT_0+T_0-2}\sum_{i\in\mathcal{V}_{z_l}}\left(w_i^l\|x_i^l - x_i^{l_{prev}}\| + \eta\|g_i(\hat{x}_i^l, x_{-i}^l)\|\right) \\
&\leq \sum_{l=KT_0}^{KT_0+T_0-2}\sum_{i\in\mathcal{V}_{z_l}}\left(\frac{\bar{\epsilon}}{2(T_0-1)N_i} + \frac{\bar{\epsilon}}{2(T_0-1)N_0}\right) \leq \bar{\epsilon} \leq \rho_0,
\end{aligned}
\tag{86}
$$

where the third inequality utilized (28), and $|\mathcal{V}_{z_l}| = N_i \leq N_0$. Then the smoothness of $f(x)$ at x^k implies that

$$
\|\nabla_x f(x^k) - \nabla_x f(x^{k'})\| \leq \phi_0\bar{\epsilon}.
\tag{87}
$$

It follows that

$$
\begin{aligned}
\frac{1}{T_0}\|\nabla_x f(x^{k'})\|^2 &= \frac{1}{T_0}\sum_{j=1}^{s}\sum_{i\in\mathcal{V}_j}\|\nabla_{x_i}f(x^{k'})\|^2 \\
&\leq \frac{1}{T_0}\sum_{k=KT_0}^{(K+1)T_0-1}\sum_{i\in\mathcal{V}_{z_k}}\|\nabla_{x_i}f(x^{k'})\|^2 \\
&\leq \frac{1}{T_0}\sum_{k=KT_0}^{(K+1)T_0-1}\sum_{i\in\mathcal{V}_{z_k}}\left(2\|\nabla_{x_i}f(x^k)\|^2 + 2\|\nabla_{x_i}f(x^k) - \nabla_{x_i}f(x^{k'})\|^2\right) \\
&\leq \frac{1}{T_0}\sum_{k=KT_0}^{(K+1)T_0-1}\sum_{i\in\mathcal{V}_{z_k}}2\|\nabla_{x_i}f(x^k)\|^2 + 2\phi_0^2\bar{\epsilon}^2.
\end{aligned}
\tag{88}
$$

Then we have

$$\frac{1}{T}\sum_{k=0}^{T-1}\|\nabla_x f(x^k)\|^2 = \frac{T_0}{T}\sum_{K=0}^{M-1}\sum_{k'=KT_0}^{(K+1)T_0-1}\frac{1}{T_0}\|\nabla_x f(x^{k'})\|^2$$

$$\leq \frac{T_0}{T}\sum_{K=0}^{M-1}\left[\sum_{k=KT_0}^{(K+1)T_0-1}\sum_{i\in\mathcal{V}_{z_k}}2\|\nabla_{x_i} f(x^k)\|^2 + 2T_0\phi_0^2\bar{\epsilon}^2\right] \quad (89)$$

$$= \frac{T_0}{T}\left[\sum_{k=0}^{MT_0-1}\sum_{i\in\mathcal{V}_{z_k}}2\|\nabla_{x_i} f(x^k)\|^2 + 2MT_0\phi_0^2\bar{\epsilon}^2\right],$$

where the second inequality used (88).

Reusing (85), the following holds with probability $1 - \frac{1}{\alpha}(2 + \gamma + \frac{1}{\nu} + \nu\gamma)$:

$$\frac{1}{T}\sum_{k=0}^{T-1}\|\nabla_x f(x^k)\|^2 < 2T_0(\epsilon + \phi_0^2\bar{\epsilon}^2). \quad (90)$$

This completes the proof. ∎

Proof of Lemma 3: For notation simplicity, when the expectation is taken over all the random variables in a formula, we use $\mathbb{E}$ as the shorthand. Moreover, we use ∇_i to represent ∇_{x_i}.

During the derivation, we will use the first-order approximation (assuming a sufficiently small r), i.e.,

$$h_i(x_i + ru_i, x_{-i}, \xi) = h_i(x_i, x_{-i}, \xi) + \nabla_i h_i(x_i, x_{-i}, \xi)ru_i + \mathcal{O}(r^2). \quad (91)$$

For g_l, it holds that

$$\mathbb{E}[g_l g_l^\top] = \mathbb{E}[\frac{q_i}{r}h_i(x_i + ru_i, x_{-i}, \xi)u_i u_i^\top \frac{q_i}{r}h_i(x_i + ru_i, x_{-i}, \xi)]$$

$$= \frac{q_i^2}{r^2}\mathbb{E}[h_i^2(x_i + ru_i, x_{-i}, \xi)u_i u_i^\top]$$

$$= \frac{q_i^2}{r^2}\mathbb{E}\left[(h_i(x, \xi) + ru_i^\top \nabla_i h_i(x, \xi) + \mathcal{O}(r^2))^2 u_i u_i^\top\right] \quad (92)$$

$$= \frac{q_i^2}{r^2}\mathbb{E}_\xi[h_i^2(x, \xi) + \mathcal{O}(r^2)]\mathbb{E}_{u_i}[u_i u_i^\top]$$

$$= \frac{q_i}{r^2}\mathbb{E}[h_i^2(x, \xi) + \mathcal{O}(r^2)]I_{q_i},$$

where the fourth inequality is obtained by $\mathbb{E}(u_i u_i^\top u_i) = 0$, and the last equality used $\mathbb{E}[u_i u_i^\top] = \frac{1}{q_i}I_{q_i}$ since $u_i \sim \text{Uni}(\mathbb{S}_{q_i-1})$.

On the other hand,

$$\mathbb{E}[\frac{q_i}{r}h_i(x_i + ru_i, x_{-i}, \xi)u_i] = \frac{q_i}{r}\mathbb{E}\left[(h_i(x, \xi)u_i + ru^\top \nabla_{x_i} h_i(x, \xi))u_i + \mathcal{O}(r^2)\right] = \mathbb{E}[\nabla_{x_i} h_i(x, \xi)] + \mathcal{O}(r). \quad (93)$$

As a result, $\text{Cov}(g_l)$ is in (32).

Similarly,

$$\mathbb{E}[g_g] = \mathbb{E}[\frac{q}{r}h(x + rz, \xi)z_i]$$

$$= \frac{q}{r}\mathbb{E}[h(x, \xi)z_i + rz^\top \nabla_{x_i} h(x, \xi)z_i + \mathcal{O}(r^2)]$$

$$= q\mathbb{E}[z_i z^\top]\mathbb{E}[\nabla_x h(x, \xi)] + \mathcal{O}(r) \quad (94)$$

$$= [\mathbf{0}_{q_i \times q_1}, \cdots, I_{q_i}, \cdots, \mathbf{0}_{q_i \times q_N}]\mathbb{E}[\nabla_x h(x, \xi)] + \mathcal{O}(r)$$

$$= \mathbb{E}[\nabla_{x_i} h(x, \xi)] + \mathcal{O}(r),$$

where the last equality used $\mathbb{E}[zz^\top] = \frac{1}{q}I_q$. Also

$$\mathbb{E}[g_g g_g^\top] = \frac{q^2}{r^2}\mathbb{E}\left[h(x + rz, \xi)z_i z_i^\top h(x + rz, \xi)\right]$$

$$= \frac{q^2}{r^2}\mathbb{E}\left[(h(x, \xi) + rz^\top \nabla_x h(x, \xi) + \mathcal{O}(r^2))^2 z_i z_i^\top\right] \quad (95)$$

$$= \frac{q}{r^2}\mathbb{E}[h^2(x, \xi) + \mathcal{O}(r^2)]I_{q_i}.$$

Thus, $\text{Cov}(g_g)$ is in (33). ∎

B. Analysis for Application to Multi-Agent LQR

Proof of Proposition 1: (i). Due to the boundedness of $x(0)$, there must exist $c_{lqr} > 0$ such that for any $x(0) \sim \mathcal{D}$, it holds that $x(0)x^\top(0) \preceq c_{lqr}\mathbb{E}[x(0)x^\top(0)] = c_{lqr}\Sigma_x$. Let $Q_{i,K}^\infty = \sum_{t=0}^\infty \gamma^t (\mathcal{A} - \mathcal{B}K)^{t\top}(\hat{Q}_i + K^\top \hat{R}_i K)(\mathcal{A} - \mathcal{B}K)^t$. It follows that

$$H_i(K, x(0)) = \langle Q_{i,K}^\infty, x(0)x^\top(0) \rangle \leq c_{lqr}\langle Q_{i,K}^\infty, \Sigma_x \rangle = c_{lqr}\mathbb{E}_{x(0)\sim\mathcal{D}}[H_i(K, x(0))] = c_{lqr}J_i(K). \tag{96}$$

(ii). Given two control gains

$$K = (K_1^\top, ..., K_i^\top, ..., K_N^\top)^\top,$$
$$K' = (K_1^\top, ..., K_i'^\top, ..., K_N^\top)^\top.$$

Given the initial state $x(0)$, let $x(t)$ and $x'(t)$ be the resulting entire system state trajectories by implementing controllers $u = -Kx$ and $u = -K'x$, respectively. It holds for any $j \in \mathcal{V}$ that

$$x_j(t+1) = A_j x_j(t) + B_j K_j x(t) = A_j x_j(t) + B_j \tilde{K}_j x_{\mathcal{N}_S^j}(t). \tag{97}$$

Note that for all $j \in \mathcal{V} \setminus \mathcal{N}_L^i$, it must hold that $\mathcal{N}_C^j \cap \mathcal{V}_S^i = \varnothing$. From the definition of $\mathcal{V}_S^i$, we have $x_j(t) = x_j'(t)$ and $x_{\mathcal{N}_S^j}(t) = x'_{\mathcal{N}_S^j}(t)$ for all $j \in \mathcal{V} \setminus \mathcal{N}_L^i$. It follows that $x^\top(Q - \hat{Q}_i)x = x'^\top(Q - \hat{Q}_i)x'$, and for any $j \in \mathcal{V} \setminus \mathcal{N}_L^i$, we have

$$x^\top K_j^\top R_j K_j x = x_{\mathcal{N}_S^j}^\top \tilde{K}_j^\top R_j \tilde{K}_j x_{\mathcal{N}_S^j} = x'^\top_{\mathcal{N}_S^j} \tilde{K}_j^\top R_j \tilde{K}_j x'_{\mathcal{N}_S^j}.$$

Therefore,

$$\begin{aligned}
& J(K) - J(K') \\
&= \mathbb{E}\left[\sum_{t=0}^\infty x^\top(Q + K^\top RK)x\right] - \mathbb{E}\left[\sum_{t=0}^\infty x'^\top(Q + K^\top RK)x'\right] \\
&= \mathbb{E}\left[\sum_{t=0}^\infty x^\top(\hat{Q}_i + \sum_{j \in \mathcal{N}_L^i} K_j^\top R_j K_j)x\right] - \mathbb{E}\left[\sum_{t=0}^\infty x'^T(\hat{Q}_i + \sum_{j \in \mathcal{N}_L^i} K'^\top_j R_j K'_j)x'\right] \\
&= J_i(K) - J_i(K').
\end{aligned} \tag{98}$$

This means that any perturbation on K_i results in the same difference on $J(K)$ and $J_i(K)$. The proof is completed. ∎

Proof of Lemma 5: Note that

$$\begin{aligned}
H_i(K^{i,k}, x(0)) - H_{i,T_J}(K^{i,k}, x(0)) &= \sum_{t=T_J}^\infty \gamma^t x^\top(t)(\hat{Q}_i + K^{i,k\top}\hat{R}_i K^{i,k})x(t) \\
&= \sum_{t=T_J}^\infty y^\top(t)(\hat{Q}_i + K^{i,k\top}\hat{R}_i K^{i,k})y(t) \\
&= x^\top(0)\left((\sqrt{\gamma}(\mathcal{A} - \mathcal{B}K^{i,k}))^{T_J}\right)^\top P_i \left(\sqrt{\gamma}(\mathcal{A} - \mathcal{B}K^{i,k})\right)^{T_J} x(0) \\
&\leq \lambda_{\max}(x(0)x^\top(0))\text{trace}(P_i)\|\sqrt{\gamma}(\mathcal{A} - \mathcal{B}K^{i,k})\|^{2T_J}
\end{aligned} \tag{99}$$

where $y(t) = \gamma^{t/2}x(t)$, the third equality comes from the fact $y(t+1) = \sqrt{\gamma}(\mathcal{A} - \mathcal{B}K)y(t)$, and P_i is the solution to

$$P_i = \hat{Q}_i + K^{i,k\top}\hat{R}_i K^{i,k} + \gamma(\mathcal{A} - \mathcal{B}K^{i,k})^\top P_i(\mathcal{A} - \mathcal{B}K^{i,k}). \tag{100}$$

Since $\mathbb{E}[x^\top(0)P_i x(0)] = J_i(K^{i,k})$, we have $\text{trace}(P_i) \leq \frac{J_i(K^{i,k})}{\lambda_{\min}(\Sigma_x)} \leq \frac{\alpha_i J_i(K^0)}{\lambda_{\min}(\Sigma_x)}$. Recalling that $\|\sqrt{\gamma}(\mathcal{A} - \mathcal{B}K^{i,k})\| \leq \kappa_0$, we have

$$H_i(K^{i,k}, x(0)) - H_{i,T_J}(K^{i,k}, x(0)) \leq \varpi \alpha_i J_i(K^0)\kappa_0^{2T_J}, \tag{101}$$

where $\varpi = \frac{\lambda_{\max}(x(0)x^\top(0))}{\lambda_{\min}(\Sigma_x)}$.

It follows from (52) that

$$2T_J \geq \frac{\log\left(\varpi \alpha_i J_i(K^0)/\epsilon'\right)}{\log(1/\kappa_0)} = \log_{\kappa_0}(\epsilon'/(\varpi \alpha_i J_i(K^0))), \tag{102}$$

where the inequality used the fact $1 - \kappa_0 \leq \log(1/\kappa_0)$. Substituting (102) into (101) yields $H_i(K^{i,k}, x(0)) - H_{i,T_J}(K^{i,k}, x(0)) \leq \epsilon'$.

Next we prove (54). For any $i \in \mathcal{V}$, we have

$$\begin{aligned}
\|\mathbb{E}[\mathbf{G}_i(k) - \mathbf{G}_i^{T_J}(k)]\| &\leq |\frac{q_i}{r_i}(H_i(K^{i,k}, x(0)) - H_{i,T_J}(K^{i,k}, x(0)))| \cdot \max_{\mathbf{D}_i \in \mathbb{S}_{q_i-1}} \|\mathbf{D}_i\| \\
&\leq \frac{q_i}{r_i}\epsilon'.
\end{aligned} \tag{103}$$

According to Lemma 2, when $r_i \leq \rho_0 \leq \beta_K$, we have

$$\|\mathbb{E}[\mathbf{G}_i(k)] - \nabla_{\mathbf{K}_i} J(K^{i,k})\| \leq \phi_0 r_i. \tag{104}$$

Using the triangular inequality yields (54).

Next we bound $\|\mathbf{G}_i^{T_J}(k)\|$:

$$
\begin{aligned}
\|\mathbf{G}_i^{T_J}(k)\| &\leq \frac{q_i}{r_i}(H_i(K^{i,k}, x(0)) + \epsilon') \\
&\leq \frac{q_i}{r_i}(c_{lqr} J_i(K^{i,k}) + \epsilon') \\
&\leq \frac{q_i}{r_i}\left[c_{lqr}(J_i(K^k) + \lambda_0 \rho_0) + \epsilon'\right] \\
&\leq \frac{q_i}{r_i}\left[c_{lqr}(\alpha_i J_i(K^0) + \lambda_0 \rho_0) + \epsilon'\right],
\end{aligned}
\tag{105}
$$

where the first inequality used the first statement in Proposition 1, the second inequality used

$$\|\mathbf{K}^{i,k} - \mathbf{K}^k\| \leq \|\hat{\mathbf{K}}_i^k - \mathbf{K}_i^k\| + r_i = w_i \|\mathbf{K}_i^k - \mathbf{K}_i^{k_{prev}}\| + r_i \leq \rho_0,$$

the third inequality used $J_i(K^k) \leq \alpha_i J_i(K^0)$. ∎

REFERENCES

[1] J. C. Spall, *Introduction to stochastic search and optimization: estimation, simulation, and control*. John Wiley & Sons, 2005, vol. 65.

[2] Y. Nesterov and V. Spokoiny, "Random gradient-free minimization of convex functions," *Foundations of Computational Mathematics*, vol. 17, no. 2, pp. 527–566, 2017.

[3] A. D. Flaxman, A. T. Kalai, and H. B. McMahan, "Online convex optimization in the bandit setting: gradient descent without a gradient," *arXiv preprint cs/0408007*, 2004.

[4] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 15–26.

[5] M. Fazel, R. Ge, S. Kakade, and M. Mesbahi, "Global convergence of policy gradient methods for the linear quadratic regulator," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1467–1476.

[6] D. Malik, A. Pananjady, K. Bhatia, K. Khamaru, P. L. Bartlett, and M. J. Wainwright, "Derivative-free methods for policy optimization: Guarantees for linear quadratic systems," *Journal of Machine Learning Research*, vol. 21, no. 21, pp. 1–51, 2020.

[7] L. Furieri, Y. Zheng, and M. Kamgarpour, "Learning the globally optimal distributed lq regulator," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 287–297.

[8] H. Mohammadi, M. Soltanolkotabi, and M. R. Jovanović, "On the linear convergence of random search for discrete-time lqr," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 989–994, 2020.

[9] A. Agarwal, O. Dekel, and L. Xiao, "Optimal algorithms for online convex optimization with multi-point bandit feedback." in *COLT*. Citeseer, 2010, pp. 28–40.

[10] O. Shamir, "An optimal algorithm for bandit and zero-order convex optimization with two-point feedback," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1703–1713, 2017.

[11] Y. Zhang, Y. Zhou, K. Ji, and M. M. Zavlanos, "A new one-point residual-feedback oracle for black-box learning and control," *Automatica*, vol. 136, p. 110006, 2022.

[12] P. K. Sharma, E. Zaroukian, R. Fernandez, A. Basak, and D. E. Asher, "Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*, vol. 11746. International Society for Optics and Photonics, 2021, p. 117462K.

[13] H. Sun and M. Hong, "Distributed non-convex first-order optimization and information processing: Lower complexity bounds and rate optimal algorithms," *IEEE Transactions on Signal processing*, vol. 67, no. 22, pp. 5912–5928, 2019.

[14] D. Hajinezhad, M. Hong, and A. Garcia, "Zone: Zeroth-order nonconvex multiagent optimization over networks," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 3995–4010, 2019.

[15] C. Gratton, N. K. Venkategowda, R. Arablouei, and S. Werner, "Privacy-preserving distributed zeroth-order optimization," *arXiv preprint arXiv:2008.13468*, 2020.

[16] Y. Tang, J. Zhang, and N. Li, "Distributed zero-order algorithms for nonconvex multi-agent optimization," *IEEE Transactions on Control of Network Systems*, 2020.

[17] A. Akhavan, M. Pontil, and A. B. Tsybakov, "Distributed zero-order optimization under adversarial noise," *arXiv preprint arXiv:2102.01121*, 2021.

[18] Y. Li, Y. Tang, R. Zhang, and N. Li, "Distributed reinforcement learning for decentralized linear quadratic control: A derivative-free policy optimization approach," *arXiv preprint arXiv:1912.09135*, 2019.

[19] Y. Nesterov, "Efficiency of coordinate descent methods on huge-scale optimization problems," *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 341–362, 2012.

[20] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, "Coordinate-friendly structures, algorithms and applications," *Annals of Mathematical Sciences and Applications*, vol. 1, no. 1, pp. 57–119, 2016.

[21] A. A. Canutescu and R. L. Dunbrack Jr, "Cyclic coordinate descent: A robotics algorithm for protein loop closure," *Protein science*, vol. 12, no. 5, pp. 963–972, 2003.

[22] P. Richtárik and M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Mathematical Programming*, vol. 144, no. 1, pp. 1–38, 2014.

[23] S. J. Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3–34, 2015.

[24] Y. Xu and W. Yin, "A globally convergent algorithm for nonconvex optimization based on block coordinate update," *Journal of Scientific Computing*, vol. 72, no. 2, pp. 700–734, 2017.

[25] H. Cai, Y. Lou, D. McKenzie, and W. Yin, "A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization," *arXiv preprint arXiv:2102.10707*, 2021.

[26] Z. Yu and D. W. Ho, "Zeroth-order stochastic block coordinate type methods for nonconvex optimization," *arXiv preprint arXiv:1906.05527*, 2019.

[27] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[28] X. Fang, X. Li, and L. Xie, "Angle-displacement rigidity theory with application to distributed network localization," *IEEE Transactions on Automatic Control*, 2020.

[29] H. Ebel and P. Eberhard, "Distributed decision making and control for cooperative transportation using mobile robots," in *International Conference on Swarm Intelligence*. Springer, 2018, pp. 89–101.

[30] H. Feng and J. Lavaei, "On the exponential number of connected components for the feasible set of optimal decentralized control problems," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 1430–1437.

[31] J. Bu, A. Mesbahi, and M. Mesbahi, "On topological and metrical properties of stabilizing feedback gains: the mimo case," *arXiv preprint arXiv:1904.02737*, 2019.

[32] F. Borrelli and T. Keviczky, "Distributed lqr design for identical dynamically decoupled systems," *IEEE Transactions on Automatic Control*, vol. 53, no. 8, pp. 1901–1912, 2008.

[33] J. Bu, A. Mesbahi, M. Fazel, and M. Mesbahi, "Lqr through the lens of first order methods: Discrete-time case," *arXiv preprint arXiv:1907.08921*, 2019.

[34] G. Jing, H. Bai, J. George, A. Chakrabortty, and P. K. Sharma, "Learning distributed stabilizing controllers for multi-agent systems," *IEEE Control Systems Letters*, 2021.

[35] G. Jing, H. Bai, J. George, and A. Chakrabortty, "Model-free optimal control of linear multi-agent systems via decomposition and hierarchical approximation," *IEEE Transactions on Control of Network Systems*, 2021.

[36] S. Kar, J. M. Moura, and H. V. Poor, "$\mathcal{QD}$-learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1848–1862, 2013.

[37] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2681–2690.

[38] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv preprint arXiv:1706.02275*, 2017.

[39] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5872–5881.

[40] T. Chen, K. Zhang, G. B. Giannakis, and T. Basar, "Communication-efficient policy gradient methods for distributed reinforcement learning," *IEEE Transactions on Control of Network Systems*, 2021.

[41] M. Rudelson, R. Vershynin *et al.*, "Hanson-wright inequality and sub-gaussian concentration," *Electronic Communications in Probability*, vol. 18, 2013.

[42] A. G. Chkhartishvili, D. A. Gubanov, and D. A. Novikov, *Social Networks: Models of information influence, control and confrontation*. Springer, 2018, vol. 189.

[43] X. Chen, "Multi-agent systems with reciprocal interaction laws," Ph.D. dissertation, Harvard University, 2014.

[44] J. Qin, Q. Ma, Y. Shi, and L. Wang, "Recent advances in consensus of multi-agent systems: A brief survey," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 6, pp. 4972–4983, 2016.

[45] 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.

[46] A. Lamperski, "Computing stabilizing linear controllers via policy iteration," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 1902–1907.
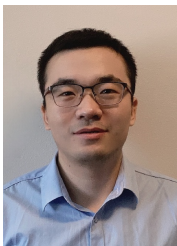
Gangshan Jing received the Ph.D. degree in Control Theory and Control Engineering from Xidian University, Xi'an, China, in 2018. He was a research assistant and a postdoctoral researcher at Hong Kong Polytechnic University, Hong Kong and Ohio State University, USA in 2016-2017 and 2018-2019, respectively. He was a postdoctoral researcher at North Carolina State University, USA, during 2019-2021. Since Dec. 2021, he has been a faculty member in School of Automation, Chongqing University. His research interests include control, optimization, and machine learning for network systems.

He Bai received his Ph.D. degree in Electrical Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2009. From 2009 to 2010, he was a postdoctoral researcher at Northwestern University, Evanston, IL. From 2010 to 2015, he was a Senior Research and Development Scientist at UtopiaCompression Corporation, Los Angeles, CA. In 2015, he joined the School of Mechanical and Aerospace Engineering at Oklahoma State University, Stillwater, OK, where he is currently an associate professor. His research interests include distributed estimation, control and learning, reinforcement learning, nonlinear control, and robotics.

Jemin George received his M.S. ('07), and Ph.D. ('10) in Aerospace Engineering from the State University of New York at Buffalo. Prior to joining ARL in 2010, he worked at the U.S. Air Force Research Laboratory's Space Vehicles Directorate and the National Aeronautics, and Space Administration's Langley Aerospace Research Center. From 2014-2017, he was a Visiting Scholar at the Northwestern University, Evanston, IL. His principal research interests include decentralized/distributed learning, stochastic systems, control theory, nonlinear estimation/filtering, networked sensing and information fusion.

Aranya Chakrabortty 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 received the NSF CAREER award in 2011.

Piyush Sharma received his M.S. and Ph.D. degrees in Applied Mathematics from the University of Puerto Rico and Delaware State University respectively. He has government and industry work experiences. Currently, he is with the U.S. Army as an AI Coordinator at ATEC, earlier a Computer Scientist at ARL. Prior to joining ARL, he worked for Infosys' Data Analytics Unit (DNA) where his role was a Senior Associate Data Scientist. He has a core data science experience and has been responsible for Thought Leadership and solving stakeholders' problems, communicating results and methodologies with clients. His principal research includes Multi-Agent Reinforcement Learning (MARL), Multi-Modality Learning in Multi-Agent Systems (MAS), decision-making, and Internet of Battlefield Things (IoBT): a growing area for gathering reliable and actionable information from various sources in Multi-Domain Operations (MDO). His recent research work appeared in ARL NEWS ¡ https://www.army.mil/article/258408 ¿