Ji Ma; Zheng Yang; Ziqin Chen

Distributed Nash equilibrium tracking via the alternating direction method of multipliers

Kybernetika, Vol. 59 (2023), No. 4, 612–632

Persistent URL: http://dml.cz/dmlcz/151854

KYBERNETIKA — VOLUME *59* (2023), NUMBER 4, PAGES 612–632

DISTRIBUTED NASH EQUILIBRIUM TRACKING VIA THE ALTERNATING DIRECTION METHOD OF MULTIPLIERS

Ji Ma, Zheng Yang and Ziqin Chen

Nash equilibrium is recognized as an important solution concept in non-cooperative game theory due to its broad applicability to economics, social sciences, computer science, and engineering. In view of its importance, substantial progress has been made to seek a static Nash equilibrium using distributed methods. However, these approaches are inapplicable in dynamic environments because, in this setting, the Nash equilibrium constantly changes over time. In this paper, we propose a dynamic algorithm that can track the time-varying Nash equilibrium in a non-cooperative game. Our approach enables each player to update its action using an alternating direction method of multipliers while ensuring this estimated action of each player always converges to a neighborhood of the Nash equilibrium at each sampling instant. We prove that the final tracking error is linearly proportional to the sampling interval, which implies that the tracking error can be sufficiently close to zero when the sampling interval is small enough. Finally, numerical simulations are conducted to verify the correctness of our theoretical results.

Keywords: game theory, time-varying Nash equilibrium tracking, alternating direction method of multipliers

Classification: 90C33,68W15

1. INTRODUCTION

In recent years, non-cooperative games have attracted extensive attention due to their diverse applications in various engineering fields, such as power systems [2, 3], communication networks [4, 7], and multi-cloud systems [1]. The fundamental problem in non-cooperative games is finding the Nash equilibrium (NE), which serves as the basis for understanding and predicting the outcomes of strategic interactions between multiplayers. Recently, the rapid development of multi-agent system and distributed optimization theory provide new perspectives and ideas for solving NE seeking problems in a distributed manner [5, 10, 15, 19, 20]. In these distributed NE seeking algorithms, players determine their actions individually and reach the NE after certain rounds of iterations by exchanging partial information with neighbors through a communication network.

Note that the above mentioned distributed NE seeking algorithms [5, 10, 15, 19, 20] mainly focused on static games with time-invariant cost functions. However, in many realistic situations, such as real-time traffic networks, online auctions, and dynamic

DOI: 10.14736/kyb-2023-4-0612

wireless environment [12], the cost functions within the game are time-varying. Traditional distributed NE seeking algorithms have difficulty tracking fast-changing NEs within short time intervals. To address this issue, researchers have developed dynamic approaches. A notable approach is the online distributed algorithm [11, 13]. This approach enables each player to decide their current actions based on the values and gradients of past cost functions available to them. By leveraging historical information, the online algorithm ensures that the tracking error (referred to as a regret function) is bounded by the product of a term that depends on the deviation of the varying NE sequence and a sublinear function of learning time.

In addition to online algorithms, another approach is the distributed tracking algorithm. Unlike online algorithms for minimizing regret functions, distributed tracking algorithms aim to directly track the time-varying optimal point, resulting in a minimal tracking error (the norm of difference between the tracking variables and the optimal point). This approach has been successfully applied to solving distributed dynamic optimization problem. For example, the authors in [9] presented a dynamic alternating direction method of multipliers (ADMM) and the authors in [14, 16] designed a distributed prediction-correction algorithm. Both of them can achieve a bounded tracking error. However, to the best of our knowledge, distributed tracking algorithms have not been applied to solving time-varying NE problems. Note that some distributed continuous-time NE tracking algorithms [6, 17, 18] have been proposed, while these methods rely on ODE solutions and may not be implemented using digital computation.

In this paper, we are devoted to solving the distributed NE tracking problem. Our basic idea is to treat the time-varying NE tracking problem as a sequence of static NE seeking problems and solve them one by one. However, for the real-time tracking requirement in some applications, traditional NE seeking algorithms cannot afford to spend the time required to precisely solve every static problem. To address this dilemma, we proposed a distributed dynamic algorithm, which approximately solves each static problem by using distributed ADMM, and then uses the current step's approximate solution as the initial point for the next step. We prove that the proposed algorithm can effectively track the time-varying NE with a bounded tracking error, whose size is determined by the variation of cost functions and the sampling interval. Our numerical experiments with a Nash–Cournot game problem confirm that the proposed algorithm can indeed track the time-varying NE.

The contributions of this paper are summarized as follows.

- We propose a distributed dynamic algorithm for tracking time-varying NE in a non-cooperative game, in which the cost functions constantly change over time. To the best of our knowledge, it is the first to propose a distributed tracking approach to real-time tracking of the time-varying NE.

- We prove that the proposed algorithm ensures that the tracking error is approximately linear with the product of a term that depends on the variation of the cost function and the sampling interval. This result can include the linearly convergent result of distributed NE seeking in terms of time-invariant cost functions in [5, 10, 15, 19, 20].

- The recursive method is used to analyze the tracking accuracy of the proposed

algorithm. This is in sharp contrast to the analysis method for existing distributed continuous-time NE tracking algorithms [6,17,18] that relies on ODE solutions.

- We experimentally evaluate the proposed dynamic algorithm using a practical Nash–Cournot game. Results show that the tracking error of our algorithm decreases as the sampling interval decreases.

This paper is organized as follows. Section 2 introduces some related preliminaries on basic notations, graph theory and formulates the distributed time-varying NE tracking problem. Section 3 provides the proposed distributed time-varying ADMM and analyzes its convergence performance. Then, Section 4 gives a numerical example and Section 5 concludes the paper.

2. PRELIMINARIES AND PROBLEM FORMULATION

2.1. Basic notations and notions

We denote $\mathbb{R}^n$ as the set of real vectors with n-dimension, $\mathbb{R}^n_+$ as the set of vectors with nonnegative coordinates of n-dimension and $\mathbb{R}^{n\times m}$ as the set of real matrices with n-rows and m-columns. Let $\|\cdot\|$ and $\otimes$ be the standard Euclidean norm and the Kronecker product, respectively. Denote by $\mathbf{1}_N$ and $\mathbf{0}_N$ the column vectors of N dimension with all entries being 1 and 0, respectively. Let I_n be the compatible identity matrix with dimension n. For any vector $x, y \in \mathbb{R}^n$, let x^T be the transpose of x. Denote $[x_i]_{i\in\Omega}$ as the column vector by stacking up x_i associated with $i \in \Omega$. For any square matrix H, denote $\rho(H)$ as the spectral radius of H.

For a differentiable function $f(x) : \mathbb{R}^n \to \mathbb{R}$, its gradient is defined as $\nabla_x f(x) = \mathrm{col}(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}) \in \mathbb{R}^n$ and its component gradient is defined as $\nabla_i f(x) = \frac{\partial f}{\partial x_i}$.We say that a function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is m-strongly convex for $\mu > 0$ if for any $x, y \in \mathbb{R}^n$, we have $f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|x - y\|^2$ and a function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is radially unbounded on $\mathbb{R}^n$ if for any $x_n \in \mathbb{R}^n$ satisfying $\|x_n\| \to \infty$, we have $f(x_n) \to \infty$.

2.2. Graph theory

We assume that the information sharing between players is modeled as a connected and undirected graph $\mathcal{G} = (V, \mathcal{E})$. We denote $\mathcal{V}$ as the player set with cardinality $|\mathcal{V}| = N$ and $\mathcal{E}$ as the directed arcs set with cardinality $|\mathcal{E}| = m$. Let $\mathcal{N}_i$ be the set of neighbors of node i with cardinality $|\mathcal{N}_i| = d_i$. We define the block arc source matrices as $A_s \in \mathbb{R}^{mN\times N^2}$ and $A_d \in \mathbb{R}^{mN\times N^2}$. If an arc l goes from i to j, then the block $(A_s)_{l,i}$ and the block of $(A_d)_{l,j}$ are both $I_N \in \mathbb{R}^{N\times N}$ and are null, otherwise. Furthermore, we define the signed incidence matrix $E_o = A_s - A_d$ and the unsigned incidence matrix $E_u = A_s + A_d$. The extended signed Laplacian matrix is given by $L_o = (1/2)E_o^T E_o$ and the extended unsigned Laplacian matrix is given by $L_u = (1/2)E_u^T E_u$. The degree matrix $D = \mathrm{diag}(d_1, \ldots, d_N)$ satisfies $D = (1/2)(L_o + L_u)$. We denote λ_{Lu0} and λ_{Lo0} as the smallest singular values of L_u and L_o, respectively. Correspondingly, we denote λ_{LuN} and λ_{LoN} as the largest singular values of L_u and L_o, respectively.

We give the following assumption on the communication graph:

Assumption 1. The graph $\mathcal{G}$ is undirected and connected.

2.3. Problem formulations

We consider a non-cooperative game $G^t(V, f_i^t, x_i)$ with N players. $V = \{1, \ldots, N\}$ is the set of all players and $f_i^t(x_i, x_{-i}),\ t \geq 0$ is the time-varying cost function for player $i \in V$ associated with the player i's action $x_i \in \mathbb{R}$ and all players' actions $x_{-i} \in \mathbb{R}^{N-1}$ except player i's. At any time t, each player i aims to minimize his own cost function selfishly with respect to x_i. That is, each player $i,\ i \in V$ tracks an optimal solution trajectory of the following time-varying optimization problem:

$$\min_{x_i \in \mathbb{R}} f_i^t(x_i, x_{-i}),\ \forall i \in V. \tag{1}$$

Note that at any time t, each player's optimal action is dependent on the other players' actions. An NE lies at the intersection of the solutions set for Problem (1), such that no player can reduce his cost by unilaterally deviating from his action. We assume that there exists a NE trajectory for the problem (1), and introduce the following assumption to guarantee the uniqueness of this NE trajectory.

Assumption 2. For every player $i \in V$, the cost function $f_i^t(x_i, x_{-i})$ is μ-strongly convex in $x_i(t)$ for any fixed $x_{-i}(t)$, and uniformly in t.

Problem (1) can be interpreted as a sequence of static NE seeking problems. Specifically, the cost functions $f_i^t(x_i, x_{-i})$ are sampled at time instant t_k with $k = 1, \ldots$, and sampling interval $h = t_k - t_{k-1}$ can be chosen as a small number. Then, the problem (1) can be approximated by

$$\min_{x_i \in \mathbb{R}} f_i^{t_k}(x_i, x_{-i}),\ \forall i \in V. \tag{2}$$

In the distributed framework, the actions of all players cannot be accessed by each player. Hence, the augmented problem (2) is addressed by introducing local estimations of players' actions and using a consensus constraint to reach an agreement on these estimations. The notations below are used to represent players' estimations.

- $x_{-i}^i \in \mathbb{R}^{N-1}$: Player i's estimations of all others' actions except himself.
- $x_i^i \in \mathbb{R}$: Player i's estimation of his action, which is indeed its actual action, i. e., $x_i^i = x_i$ for $i \in V$.
- $x^i = (x_i^i, x_{-i}^i) \in \mathbb{R}^N$: Player i's estimations of all players' actions.
- $\boldsymbol{x} = [x^1; \ldots; x^N] \in \mathbb{R}^{N^2}$: The stacked vector of all estimations for all players.

Based on the actions' estimations $x^i,\ \forall i \in V$, Problem (2) can be reformulated as

$$\begin{aligned} \min_{x^i \in \mathbb{R}^N} \quad & f_i^{t_k}(x^i), \\ \text{subject to} \quad & x^i = x^j,\ \forall i \in V,\ \forall j \in \mathcal{N}_i. \end{aligned} \tag{3}$$

The consensus constraint in (3) ensures that all local estimations x^i are identical. Hence, (3) is equivalent to (2). By introducing a slack variable $z^{ij} \in \mathbb{R}^N,\ \forall (i,j) \in \mathcal{E}$ to separate the consensus constraint, we can rewrite (3) as

$$\begin{aligned} \min_{x^i \in \mathbb{R}^N} \quad & f_i^{t_k}(x^i), \\ \text{subject to} \quad & x^i = z^{ij},\ z^{ij} = x^j,\ \forall\ i \in V,\ \forall j \in \mathcal{N}_i. \end{aligned} \tag{4}$$

3. MAIN RESULT

In this section, we propose the distributed dynamic algorithm and establish its convergence analysis.

3.1. Algorithm Design

In this subsection, we develop a distributed time-varying ADMM to track the NE trajectory. Let $\alpha^{ij} \in \mathbb{R}^N$ and $\beta^{ij} \in \mathbb{R}^N$ denote Lagrange multipliers associated with two constraints in (4). Then, the augmented Lagrange function of (4) is designed as

$$\begin{aligned} L_i^{t_k}(x^i, z^{ij}, \alpha^{ij}, \beta^{ij}) = f_i^{t_k}(x^i) + \sum_{i \in V} \sum_{j \in \mathcal{N}_i} \left[(\alpha^{ij})^T (x^i - z^{ij}) + (\beta^{ij})^T (x^j - z^{ij}) \right] \\ + \frac{c}{2} \big(\|x^i - z^{ij}\|^2 + \|x^j - z^{ij}\|^2 \big), \end{aligned} \tag{5}$$

where c is a positive parameter. Note that the last term of the right hind of (5) is a correct quadratic penalty term for two constraints in (4).

Next, we propose our algorithm, which includes two steps: i) Each player estimates the actions of all other players based on exchanged information with his neighbors over the communication graph; ii) then, based on the estimations of other players' actions, each playerupdates his own action. The first step is characterized by

$$x^i_{-i}(t_{k+1}) = \frac{1}{2} \big(x^i_{-i}(t_k) + \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} x^j_{-i}(t_k) \big) - \frac{1}{2cd_i} \sum_{j \in \mathcal{N}_i} \big(\alpha^{ij}_{-i}(t_k) + \beta^{ji}_{-i}(t_k) \big). \tag{6}$$

Based on the estimation x^i_{-i} of other players' actions in (6) and the augmented Lagrange function in (5), each player updates its strategy using the following distributed ADMM:

$$\begin{aligned} x^i_i(t_{k+1}) = \arg\min_{x^i_i \in \mathbb{R}} \Big\{ f_i^{t_{k+1}}(x^i_i, x^i_{-i}(t_{k+1})) + \sum_{j \in \mathcal{N}_i} [\alpha^{ij}(t_k) + \beta^{ji}(t_k)]^T (x^i_i, x^i_{-i}(t_{k+1})) \\ + c \sum_{j \in \mathcal{N}_i} \|(x^i_i, x^i_{-i}(t_{k+1})) - z^{ij}(t_k)\|^2 \Big\}. \end{aligned} \tag{7}$$

By using (5), we obtain the update rule for the auxiliary variable z^{ij}:

$$\begin{aligned} z^{ij}(t_{k+1}) = \arg\min_{z^{ij} \in \mathbb{R}^N} \Big\{ -(\alpha^{ij}(t_k) + \beta^{ij}(t_k))^T z^{ij} + \frac{c}{2} \big(\|x^i(t_{k+1}) - z^{ij}\|^2 \\ + \|x^j(t_{k+1}) - z^{ij}\|^2 \big) \Big\}, \end{aligned}$$

which is equivalent to

$$z^{ij}(t_{k+1}) = \frac{1}{2c}(\alpha^{ij}(t_k) + \beta^{ij}(t_k)) + \frac{1}{2}(x^i(t_{k+1}) + x^j(t_{k+1})). \tag{8}$$

The updates of the dual Lagrange multipliers of each player i are described as

$$\alpha^{ij}(t_{k+1}) = \alpha^{ij}(t_k) + \frac{c}{2}\left(x^i(t_{k+1}) - x^j(t_{k+1})\right), \tag{9}$$

$$\beta^{ij}(t_{k+1}) = \beta^{ij}(t_k) + \frac{c}{2}\left(x^j(t_{k+1}) - x^i(t_{k+1})\right). \tag{10}$$

By setting $\alpha^{ij}(t_0) = \beta^{ij}(t_0) = \mathbf{0}_N$, the equations (9) and (10) imply

$$\alpha^{ij}(t_k) + \beta^{ij}(t_k) = \mathbf{0}_N,$$

which implies that the updated rule of z^{ij} in (8) can be simplified as

$$z^{ij}(t_{k+1}) = \frac{1}{2}(x^i(t_{k+1}) + x^j(t_{k+1})). \tag{11}$$

Substituting (11) into (7), the update of $x_i^i(t_{k+1})$ can be rewritten as

$$\nabla_i f_i^{t_{k+1}}\left(x_i^i, x_{-i}^i(t_{k+1})\right) + \sum_{j\in\mathcal{N}_i}(\alpha_i^{ij}(t_k) + \beta_i^{ji}(t_k)) + 2c\sum_{j\in\mathcal{N}_i}\left(x_i^i - \frac{1}{2}(x_i^i(t_k) + x_i^j(t_k))\right) = 0. \tag{12}$$

We summarize the proposed algorithm in Algorithm 3.1.

Algorithm 3.1 Distributed time-varying ADMM for each agent i

Initialization: Initialize $\alpha^{ij}(t_0)$ and $\beta^{ji}(t_0)$ to zero and set the positive constant $c > 0$.

1: **For** $k = 0, 1, 2\ldots$ **do**,

2: Each player i exchanges his estimation of other players' actions with his neighbors j, $\forall j \in \mathcal{N}_i$. Then, he updates his estimation x_{-i}^i by **c.f.** (6).

$$x_{-i}^i(t_{k+1}) = \frac{1}{2}\Big(x_{-i}^i(t_k) + \frac{1}{d_i}\sum_{j\in\mathcal{N}_i} x_{-i}^j(t_k)\Big) - \frac{1}{2cd_i}\sum_{j\in\mathcal{N}_i}\Big(\alpha_{-i}^{ij}(t_k) + \beta_{-i}^{ji}(t_k)\Big).$$

3: Obtaining the local cost function $f_i^{t_{k+1}}(x_i^i, x_{-i}^i(t_{k+1}))$, each player updates his real action x_i^i by **c.f** (12).

$$\nabla_i f_i^{t_{k+1}}\left(x_i^i, x_{-i}^i(t_{k+1})\right) + \sum_{j\in\mathcal{N}_i}(\alpha_i^{ij}(t_k) + \beta_i^{ji}(t_k)) + 2c\sum_{j\in\mathcal{N}_i}\left(x_i^i - \frac{1}{2}(x_i^i(t_k) + x_i^j(t_k))\right) = 0.$$

4: The updated rules of dual Lagrange multipliers α^{ij} and β^{ij} are described by **c.f.** (9) − (10).

$$\alpha^{ij}(t_{k+1}) = \alpha^{ij}(t_k) + \frac{c}{2}\left(x^i(t_{k+1}) - x^j(t_{k+1})\right),$$

$$\beta^{ij}(t_{k+1}) = \beta^{ij}(t_k) + \frac{c}{2}\left(x^j(t_{k+1}) - x^i(t_{k+1})\right).$$

5: **End**

We stack dual variables as $\boldsymbol{\lambda} = [\boldsymbol{\alpha}; \boldsymbol{\beta}] \in \mathbb{R}^{2mN}$, where $\boldsymbol{\alpha} = [\alpha^{ij}]_{i\in} \in \mathbb{R}^{mN}$ and $\boldsymbol{\beta} = [\beta^{ij}]_{i\in} \in \mathbb{R}^{mN}$, while stacking $\boldsymbol{z} = [z^{ij}]_{i\in} \in \mathbb{R}^{mN}$. Then, we obtain the compact form of the updates in Algorithm 3.1.

Based on the update of estimated variables in (6), we have

$$\sum_{j\in\mathcal{N}_i} \left(\alpha^{ij}_{-i}(t_k) + \beta^{ji}_{-i}(t_k)\right) + 2c \sum_{j\in\mathcal{N}_i} \left(x^i_{-i}(t_{k+1}) - \frac{1}{2}(x^i_{-i}(t_k) + x^j_{-i}(t_k))\right) = \mathbf{0}_{N-1}. \quad (13)$$

Combining with the updates of x^i_i and x^i_{-i} in (11) – (13), x^i is updated by

$$\nabla f_i^{t_{k+1}}(x^i)e_i + \sum_{j\in\mathcal{N}_i} (\alpha^{ij}(t_k) + \beta^{ij}(t_k)) + 2c \sum_{j\in\mathcal{N}_i} (x^i - z^{ij}(t_k)) = \mathbf{0}_N, \quad (14)$$

where the ith dimension element of $e_i \in \mathbb{R}^N$ is 1, and the rest elements are zero.

We further define three matrices:

$$\begin{aligned} R &= \mathrm{diag}(e_1, \ldots, e_N) \in \mathbb{R}^{N^2\times N}, \\ A &= [A_s; A_d] \in \mathbb{R}^{2mN\times N^2}, \\ B &= [-I_{mN}; -I_{mN}] \in \mathbb{R}^{2mN\times 2mN}. \end{aligned}$$

At time t_k, we define the extended pesudogradient mapping as $\mathbf{F}^{t_k}(\boldsymbol{x}) = [\nabla_i f_i^{t_k}(x^i)]_{i\in V} : \mathbb{R}^{N^2} \to \mathbb{R}^N$ with the stacked vector of all players' estimations $\boldsymbol{x}(t_k) = [x^i(t_k)]_{i\in V}$. Then, the update of x^i in (14) is rewritten as the following compact form:

$$R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}) + A^T\boldsymbol{\lambda}(t_k) + cA^T[A\boldsymbol{x} + B\boldsymbol{z}(t_k)] = \mathbf{0}_{N^2}, \quad (15)$$

with any initial strategy $\boldsymbol{x}(t_0) \in \mathbb{R}^{N^2}$. According to the update of z^{ij} in (8), the stacked auxiliary variable $\boldsymbol{z}(t_{k+1})$ is computed by

$$B^T\boldsymbol{\lambda}(t_k) + cB^T[A\boldsymbol{x}(t_{k+1}) + B\boldsymbol{z}] = \mathbf{0}_{2mN}, \quad (16)$$

with the initial auxiliary variable $\boldsymbol{z}(t_0) = 1/2E_u\boldsymbol{x}(t_0)$. Using the updates of α^{ij} and β^{ij} in (9) and (10), the stacked dual variable $\boldsymbol{\lambda}(t_{k+1})$ is given by

$$\boldsymbol{\lambda}(t_{k+1}) = \boldsymbol{\lambda}(t_k) + c[A\boldsymbol{x}(t_{k+1}) + B\boldsymbol{z}(t_{k+1})], \quad (17)$$

with the initial values of $\boldsymbol{\lambda}(t_0) = \mathbf{0}_{2mN}$.

Remark 3.1. Equations (15) – (16) is a combination of descent steps on $\boldsymbol{x}$ and $\boldsymbol{z}$, as well as an ascent step on $\boldsymbol{\lambda}$ for the Lagrangian $\boldsymbol{L}^{t_k}$, which is the compact form of (5) and given as

$$\boldsymbol{L}^{t_k}(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\lambda}) = e_i^T\boldsymbol{f}^{t_k}(\boldsymbol{x}) + \boldsymbol{\lambda}^T(A\boldsymbol{x} + B\boldsymbol{z}) + \frac{c}{2}\|A\boldsymbol{x} + B\boldsymbol{z}\|^2,$$

with $\boldsymbol{f}^{t_k}(\boldsymbol{x}) = [f_i^{t_k}(x^i)]_{i\in V}$.

3.2. Convergence analysis

In this subsection, we analyze the convergence of Algorithm 1 by bounding $\|\boldsymbol{x}(t_k) - \boldsymbol{x}^*(t_k)\|$, where $\boldsymbol{x}^*(t_k) = \mathbf{1}_N \otimes x^*(t_k)$ is the stacked NE at time instant t_k. To proceed, we establish the following auxiliary conclusion for the subsequent analysis.

Lemma 3.2. Considering the compact updates (15) – (17), the following three equations always hold.

$$\begin{aligned} &R(\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) \\ &\quad = cE_u^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})) - E_o^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})), \end{aligned} \tag{18}$$

$$\boldsymbol{z}(t_{k+1}) - \boldsymbol{z}^*(t_{k+1}) = \frac{1}{2}E_u(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})), \tag{19}$$

$$\frac{c}{2}E_o(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})) = \boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}(t_k). \tag{20}$$

P r o o f. Substituting (17) into (15), we can obtain

$$R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}) + A^T\boldsymbol{\lambda}(t_{k+1}) + cA^TB[\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})] = \mathbf{0}_{N^2}. \tag{21}$$

According to the definitions of matrices A, E_o and E_u, we have

$$A^T\boldsymbol{\lambda} = A_s^T\boldsymbol{\alpha} - A_d^T\boldsymbol{\beta} = E_o^T\boldsymbol{\alpha}, \tag{22}$$

$$A^TB = -A_s^T - A_d^T = -E_u^T. \tag{23}$$

Substituting (22) and (23) into (21) yields

$$R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}) + E_o^T\boldsymbol{\alpha}(t_{k+1}) - cE_u^T[\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})] = \mathbf{0}_{N^2}. \tag{24}$$

Similarly, substituting the multiplier update in (17) into $\boldsymbol{z}(t_{k+1})$ in (16) yields

$$B^T\boldsymbol{\lambda}(t_{k+1}) = \mathbf{0}_{2mN}, \tag{25}$$

which further derives $\boldsymbol{\alpha}(t_{k+1}) = -\boldsymbol{\beta}(t_{k+1})$ by using the initial hypothesis $\boldsymbol{\lambda}(t_0) = \mathbf{0}_{2mN}$. Then, the multiplier update (17) can be split into

$$\boldsymbol{\alpha}(t_{k+1}) = \boldsymbol{\alpha}(t_k) + c(A_s\boldsymbol{x}(t_{k+1}) - \boldsymbol{z}(t_{k+1})), \tag{26}$$

$$\boldsymbol{\beta}(t_{k+1}) = \boldsymbol{\beta}(t_k) + c(A_d\boldsymbol{x}(t_{k+1}) - \boldsymbol{z}(t_{k+1})). \tag{27}$$

Summing up the above equations, we arrive at

$$\boldsymbol{z}(t_{k+1}) = \frac{1}{2}(A_s + A_d)\boldsymbol{x}(t_{k+1}) = \frac{1}{2}E_u\boldsymbol{x}(t_{k+1}). \tag{28}$$

Substituting $\boldsymbol{z}(t_{k+1}) = \frac{1}{2}(A_s + A_d)\boldsymbol{x}(t_{k+1})$ into (26) yields

$$\boldsymbol{\alpha}(t_{k+1}) = \boldsymbol{\alpha}(t_k) + \frac{c}{2}E_o\boldsymbol{x}(t_{k+1}). \tag{29}$$

Next, we prove Lemma 3.2 based on equations (24), (28) and (29).

The KKT conditions of problem (4) can be written as the following compact form:

$$R\mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k)) + A^T\boldsymbol{\lambda}^*(t_k) = \mathbf{0}_{N^2}, \tag{30}$$

$$B^T\boldsymbol{\lambda}^*(t_k) = \mathbf{0}_{2mN}, \tag{31}$$

$$A\boldsymbol{x}^*(t_k) + B\boldsymbol{z}^*(t_k) = \mathbf{0}_{2mN}. \tag{32}$$

According to (22), it yields $A^T\boldsymbol{\lambda} = E_o^T\boldsymbol{\alpha}$. Then, (30) can be rewritten as

$$R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) + E_o^T\boldsymbol{\alpha}^*(t_{k+1}) = \mathbf{0}_{N^2}. \tag{33}$$

Recalling the definitions of A and B and splitting (32) yield

$$\begin{aligned} A_s\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{z}^*(t_{k+1}) &= \mathbf{0}_{mN}, \\ A_d\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{z}^*(t_{k+1}) &= \mathbf{0}_{mN}. \end{aligned} \tag{34}$$

Summing up the two equalities in (34), in view of $E_u = A_s + A_d$, we transform (32) into

$$\boldsymbol{z}^*(t_{k+1}) = \frac{1}{2}E_u\boldsymbol{x}^*(t_{k+1}). \tag{35}$$

Based on $E_o = A_s - A_d$ and (34), we have

$$E_o\boldsymbol{x}^*(t_{k+1}) = \mathbf{0}_{mN}. \tag{36}$$

Up to now, subtracting (33) from (24) yields (18). Subtracting (35) from (28) yields (19). Multiplying (36) by $c/2$ and subtracting the result from (29) yields (20).

This completes the proof. □

Define the variable $\boldsymbol{u} = [\boldsymbol{z}; \boldsymbol{\alpha}] \in \mathbb{R}^{2mN}$ and the matrix

$$G = \mathrm{diag}(cI_{mN}, (1/c)I_{mN}) \in \mathbb{R}^{2mN\times 2mN}.$$

The following assumption on the extended pesudo-gradient mapping $\mathbf{F}^{t_k}(\boldsymbol{x})$ is needed.

Assumption 3. For any time instants $t_k \geq 0$, the extended pesudo-gradient mapping $\mathbf{F}^{t_k}(\boldsymbol{x})$ is cocoercive. That is, for any $\boldsymbol{x},\ \boldsymbol{x}^* \in \mathbb{R}^{N^2}$, it follows that

$$\left(\mathbf{F}^{t_k}(\boldsymbol{x}) - \mathbf{F}^{t_k}(\boldsymbol{x}^*)\right)^T (x - x^*) \geq \sigma_{\mathbf{F}} \left\|\mathbf{F}^{t_k}(\boldsymbol{x}) - \mathbf{F}^{t_k}(\boldsymbol{x}^*)\right\|^2, \tag{37}$$

where $\sigma_{\mathbf{F}} > 0$ is a cocoercive constant.

In the following lemma, we bound $\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G$.

Lemma 3.3. Under Assumptions 1 – 3, the following inequality always holds for Algorithm 3.1.

$$\|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G \leq \frac{\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G}{\sqrt{1+\delta}} + \frac{g(t_{k+1})}{\sqrt{1+\delta}}. \tag{38}$$

where

$$g(t_{k+1}) = \frac{\sqrt{cm}}{\sqrt{N}}\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}^*(t_k)\| + \frac{1}{\sqrt{2c\lambda_{Lo0}}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - \mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\|, \quad (39)$$

$$\delta = \min\left\{\frac{(\mu-1)\lambda_{Lo0}}{\varsigma\lambda_{LuN}\left(\mu + 2(\mu-1)\lambda_{Lo0}\lambda_{LuN}\right)}, \frac{\varsigma\mu + 2\varsigma c\lambda_{Lo0}}{2c\sigma_{\mathbf{F}}\lambda_{Lo0}}\right\}, \quad (40)$$

with $\varsigma = \max\left\{\frac{c}{(\frac{c^2\lambda_{Lu0}}{2} - 2c^2\lambda_{LuN})}, \frac{1}{1-c\lambda_{LoN}}\right\}$ and $c \in [1, 1/\lambda_{Lo0})$.

P r o o f. See Appendix. □

Lemma 3.3 implies that $\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G$ linearly converges to a bound $\|\frac{g(t_{k+1})}{\sqrt{1+\delta}}\|$. It should be noted that in the static NE seeking problem, $\|g(t_k)\|$ in (39) is zero. As a result, our result includes the convergent result of distributed NE seeking in terms of time-invariant cost functions in [5, 10, 15, 19, 20].

Now, we are ready to present our main result.

Theorem 3.4. Under Assumptions 1–3, the following inequality holds for Algorithm 3.1.

$$\|\boldsymbol{x}(t_k) - \boldsymbol{x}^*(t_k)\| \le \frac{2}{c\lambda_{Lu0}(\sqrt{1+\delta})^k}\|\boldsymbol{u}(t_0) - \boldsymbol{u}^*(t_0)\| + \frac{2 - (\sqrt{1+\delta})^{-k}}{c\lambda_{Lu0}(\sqrt{1+\delta}-1)}\left(\frac{\sqrt{cm}}{\mu} + \frac{\sqrt{N}}{\sqrt{2c\lambda_{Lo0}}}\right)c_0 h, \quad (41)$$

which implies $\limsup_{k\to\infty}\|\boldsymbol{x}(t_k) - \boldsymbol{x}^*(t_k)\| = O(h)$.

P r o o f. Based on the results in (19) and Lemma 3.3, we derive

$$\frac{\lambda_{Lu0}}{2}\|\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})\|^2 \le \|\boldsymbol{z}(t_{k+1}) - \boldsymbol{z}^*(t_{k+1})\|^2 \le \frac{1}{c}\|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G^2 \le \frac{\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G}{c\sqrt{1+\delta}} + \frac{g(t_{k+1})}{c\sqrt{1+\delta}}. \quad (42)$$

By using (38), we obtain

$$\begin{aligned} &\|\boldsymbol{u}(t_0) - \boldsymbol{u}^*(t_0)\|_G + g(t_1) \\ &\ge \sqrt{1+\delta}\|\boldsymbol{u}(t_1) - \boldsymbol{u}^*(t_1)\|_G, \\ \cdots \qquad &\qquad \cdots \\ \|\boldsymbol{u}(t_{k-2}) - \boldsymbol{u}^*(t_{k-2})\|_G + g(t_{k-1}) &\ge \sqrt{1+\delta}\|\boldsymbol{u}(t_{k-1}) - \boldsymbol{u}^*(t_{k-1})\|_G. \end{aligned}$$

Applying the induction argument yields

$$\|\boldsymbol{u}(t_0) - \boldsymbol{u}^*(t_0)\|_G + \sum_{s=1}^{k-1}(\sqrt{1+\delta})^{s-1}g(t_s) \ge (\sqrt{1+\delta})^{k-1}\|\boldsymbol{u}(t_{k-1}) - \boldsymbol{u}^*(t_{k-1})\|. \quad (43)$$

Adding $(\sqrt{1+\delta})^{k-1}g(t_k)$ to both sides of (43), we have

$$\begin{aligned}&\|\boldsymbol{u}(t_0)-\boldsymbol{u}^*(t_0)\|_G+\sum_{s=1}^{k}(\sqrt{1+\delta})^{s-1}g(t_s)\\&\geq(\sqrt{1+\delta})^{k-1}\big[\|\boldsymbol{u}(t_{k-1})-\boldsymbol{u}^*(t_{k-1})\|+g(t_k)\big].\end{aligned}\tag{44}$$

By using (42), we have

$$\|\boldsymbol{u}(t_{k-1})-\boldsymbol{u}^*(t_{k-1})\|+g(t_k)\geq\frac{c\sqrt{1+\delta}\lambda_{Lu0}}{2}\|\boldsymbol{x}(t_k)-\boldsymbol{x}^*(t_k)\|,\tag{45}$$

which further combines with (44) resulting in

$$\begin{aligned}&\frac{1}{(\sqrt{1+\delta})^{k-1}}\|\boldsymbol{u}(t_0)-\boldsymbol{u}^*(t_0)\|_G+\sum_{s=1}^{k}\frac{1}{(\sqrt{1+\delta})^{k-s}}g(t_s)\\&\geq\frac{c\sqrt{1+\delta}\lambda_{Lu0}}{2}\|\boldsymbol{x}(t_k)-\boldsymbol{x}^*(t_k)\|.\end{aligned}\tag{46}$$

By using $g(t_s)\leq g_{\max}$ for any $t_s>0$ and the summation formula of geometric series, we obtain

$$\sum_{s=1}^{k}\frac{1}{\sqrt{1+\delta}^{(k-s)}}g(t_s)\leq\sum_{s=1}^{k}\frac{1}{\sqrt{1+\delta}^{(k-s)}}g_{\max}=\frac{1-\sqrt{1+\delta}^{(-k)}}{1-\sqrt{1+\delta}^{(-1)}}g_{\max}.\tag{47}$$

Then, we can rewrite (46) as

$$\frac{c\sqrt{1+\delta}\lambda_{Lu0}}{2}\|\boldsymbol{x}(t_k)-\boldsymbol{x}^*(t_k)\|\leq\frac{1}{(\sqrt{1+\delta})^{k-1}}\|\boldsymbol{u}(t_0)-\boldsymbol{u}^*(t_0)\|_G+\frac{1-\sqrt{1+\delta}^{(-k)}}{1-\sqrt{1+\delta}^{(-1)}}g_{\max}.\tag{48}$$

Next, we compute the upper bound of $g_{\max}$. Based on the definition of $g(t_k)$, we have

$$g(t_{k+1})=\frac{\sqrt{cm}}{\sqrt{N}}\|\boldsymbol{x}^*(t_{k+1})-\boldsymbol{x}^*(t_k)\|+\frac{1}{\sqrt{2c\lambda_{Lo0}}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))-\mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\|.\tag{49}$$

The first term is bounded by

$$\begin{aligned}\|\boldsymbol{x}^*(t_{k+1})-\boldsymbol{x}^*(t_k)\|\leq\sqrt{N}\|x^*(t_{k+1})-x^*(t_k)\|&\leq\frac{\sqrt{N}}{\mu}\|\nabla_{tx_i}f_i^{t_s}(x(t_s))\|(t_{k+1}-t_k)\\&\leq\frac{\sqrt{N}c_0h}{\mu}.\end{aligned}\tag{50}$$

By using

$$\boldsymbol{x}^*=1_N\otimes x^*\text{ and }\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))=\mathbf{F}^{t_{k+1}}(1_N\otimes x^*(t_{k+1}))=F^{t_{k+1}}(x^*(t_{k+1})),$$

then the second term of (49) is bounded by

$$\begin{aligned}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - \mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\| &\le \|F^{t_{k+1}}(x^*(t_{k+1})) - F^{t_k}(x^*(t_k))\| \\ &\le \sqrt{N}\|\nabla_{x_i} f_i^{t_{k+1}}(x^*(t_{k+1})) - \nabla_{x_i} f_i^{t_k}(x^*(t_k))\| \\ &\le \sqrt{N}\|\nabla_{tx_i} f_i^{t_s}(x(t_s))\|(t_{k+1} - t_k),\ s \in (k, k+1). \\ &\le \sqrt{N} c_0 h. \end{aligned} \tag{51}$$

Summing up (50) and (51), we have

$$g_{\max} = \frac{\sqrt{cm}c_0 h}{\mu} + \frac{\sqrt{N}c_0 h}{\sqrt{2c\lambda_{Lo0}}}. \tag{52}$$

Substituting (52) into (48) arrives at

$$\begin{aligned}\|\boldsymbol{x}(t_k) - \boldsymbol{x}^*(t_k)\| \le{}& \frac{2}{c\lambda_{Lu0}(\sqrt{1+\delta})^k}\|\boldsymbol{u}(t_0) - \boldsymbol{u}^*(t_0)\| \\ &+ \frac{2 - (\sqrt{1+\delta})^{-k}}{c\lambda_{Lu0}(\sqrt{1+\delta} - 1)}\left(\frac{\sqrt{cm}}{\mu} + \frac{\sqrt{N}}{\sqrt{2c\lambda_{Lo0}}}\right) c_0 h. \end{aligned} \tag{53}$$

When $k \to \infty$, the first term in the right-hand side of (53) tends to zero. The second term reaches a limit value $\frac{2}{c\lambda_{Lu0}(\sqrt{1+\delta}-1)} g_{\max}$, where $g_{\max}$ is related to the sampling interval h.

This completes the proof. □

Remark 3.5. We observe that the parameters λ_{Lo0}, μ, δ, m, N and c_0 can be determined when the problem and the underlying network under study are given. With these fixed parameters, Theorem 3.4 demonstrates that the asymptotic tracking error is approximately linear with the sampling interval h. In this case, it can be said that the tracking accuracy is in the order of $O(h)$. For practical applications, especially for large-scale or complex NE systems, the bound provided in Theorem 3.4 can help users adjust the sampling interval to improve tracking accuracy and obtain a better estimate of the NE trajectory.

4. SIMULATION

To evaluate the performance of the proposed algorithm, we consider the Nash–Cournot game presented in [11]. Assume that there are four players, whose communication graph can be shown as in Figure 1. The cost function of each player is described as follows.

$$f_i^t(x_i, x_{-i}) = p_i^t(x_i) - x_i d_i^t(x_i, x_{-i}),$$

where $p_i^t(x_i)$ and d_i^t are called as the production cost and the demand price respectively, and are assumed to satisfy

$$p_i^t = \alpha(t)x_i,\ d_i^t(x_i, x_{-i}) = \beta_i(t) - \sum_{j=1}^{4} x_j.$$

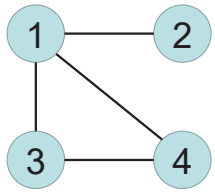

Fig. 1. Communication graph.

In this example, we set $\alpha(t) = \sin(\frac{t}{2})$ and $\beta_i(t) = 45 + 5i - 0.5i\sin(\frac{t}{2})$. The time intervals are chosen as $h = 0.1s$, $h = 0.2s$ and $h = 0.5s$. The parameter $c = 0.1$. The initial states $x^i(0),\ i = 1, 2, 3, 4$, are determined as $x^1(0) = [1, 3, -1, -1]^T$, $x^2(0) = [-1, 1, 0, 2]^T$, $x^3(0) = [0, -2, -1, 3]^T$ and $x^4(0) = [0.5, 3, -1, 1]^T$. The initial dual Lagrange multipliers are set $\alpha^{ij}(0) = \beta^{ij}(0) = 0,\ (i, j) \in \mathcal{E}$.

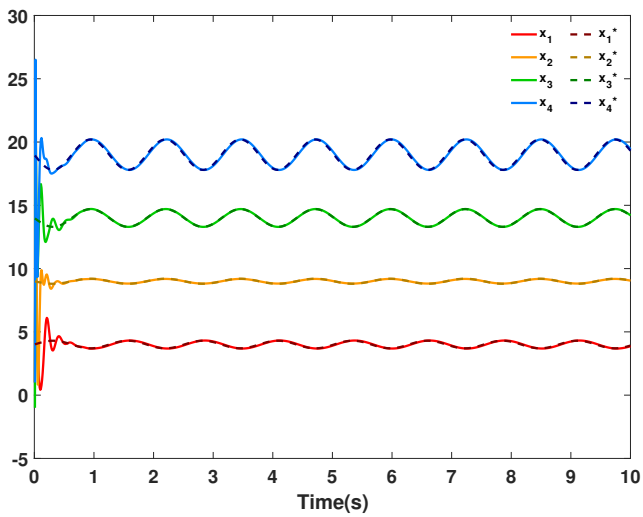

Fig. 2. Trajectories of the players' actions $x_i(t_k),\ i = 1, 2, 3, 4$ and the time varying NE $x^*(t_k) = [x_1^*(t_k), x_2^*(t_k), x_3^*(t_k), x_4^*(t_k)]^T$ with the sampling interval $h = 0.1s$.

The trajectories of the players' actions $x_i(t_k),\ i = 1, 2, 3, 4$ and the NE $x^* = [x_1^*, x_2^*, x_3^*, x_4^*]^T$ with $h = 0.1s$, $h = 0.2s$ and $h = 0.5s$ are given in Figure 2–4, respectively. Denote that tracking error as $e(t_k) = [x_1(t_k) - x_s 1^*(t_k), x_2(t_k) - x_2^*(t_k), x_3(t_k) - x_3^*(t_k), x_4(t_k) - x_4^*(t_k)]^T$. The trajectories of $\|e(t_k)\|$ are shown as in Figure 5. The results show that the tracking errors decrease as the sampling interval decreases. The convergence trajectories of the cost function $f_i^t(x_i, x_{-i}^*),\ i = 1, 2, 3, 4$ with the sampling interval $h = 0.1s$ are presented in Figure 6. The results indicate that $f_i^t(x_i, x_{-i}^*)$ converges rapidly to $f_i^t(x_i^*, x_{-i}^*)$.

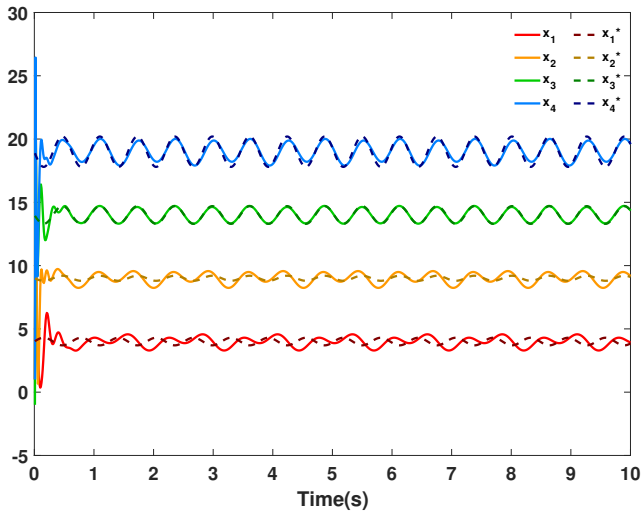

Fig. 3. Trajectories of the players' actions $x_i(t_k),\ i = 1, 2, 3, 4$ and the time varying NE $x^*(t_k) = [x_1^*(t_k), x_2^*(t_k), x_3^*(t_k), x_4^*(t_k)]^T$ with the sampling interval $h = 0.2s$.

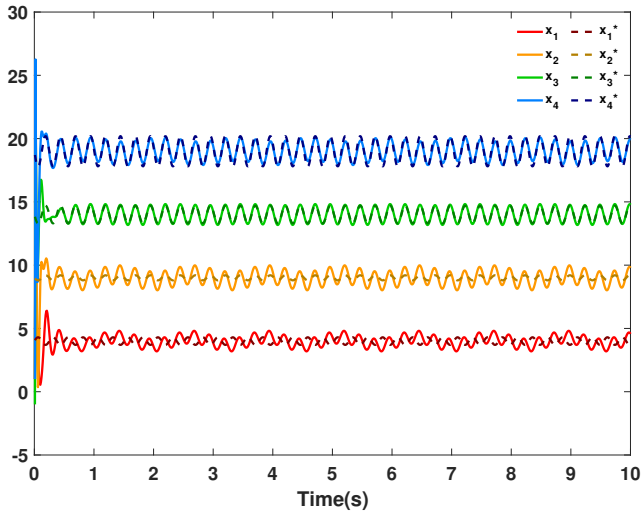

Fig. 4. Trajectories of the players' actions $x_i(t_k),\ i = 1, 2, 3, 4$ and the time varying NE $x^*(t_k) = [x_1^*(t_k), x_2^*(t_k), x_3^*(t_k), x_4^*(t_k)]^T$ with the sampling interval $h = 0.5s$.

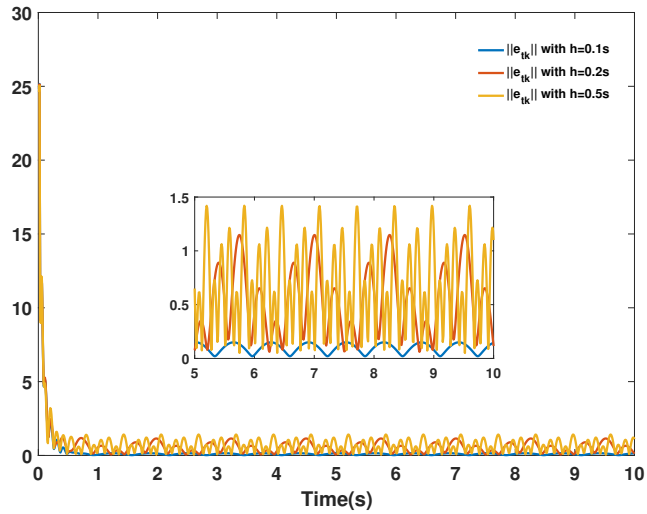

Fig. 5. Trajectories of the tracking error $\|e_{t_k}\| = \|x(t_k) - x^*(t_k)\|$ with the sampling interval $h = 0.1s$, $h = 0.2s$ and $h = 0.5s$.

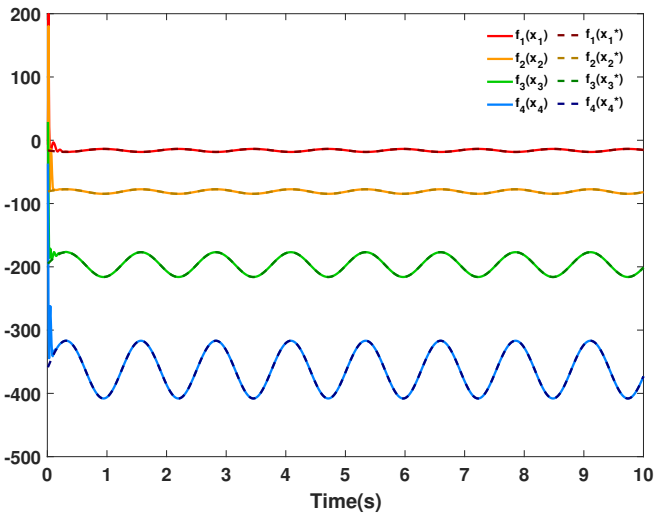

Fig. 6. The convergence trajectories of the cost function $f_i^t(x_i, x_{-i}^*),\ i = 1, 2, 3, 4$ with the sampling interval $h = 0.1s$.

5. CONCLUSION

In this paper, we proposed a distributed dynamic algorithm to solve time-varying NE tracking problem, in which each player minimizes his time-varying cost function by using a distributed ADMM at each sampling instant. We proved that our algorithm ensures the asymptotic tracking error to be linearly proportional to the sampling interval h. It implies that when the sampling interval is sufficiently small, the tracking error approaches zero. Future research will explore extensions to distributed NE tracking with equality and inequality constraints as well as distributed prediction-correction NE tracking algorithms to further reduce the tracking error.

6. APPENDIX

We prove Lemma 3.3 by using the following two steps.

Step 1: We prove the following inequality.

$$\|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G \leq \frac{\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G}{\sqrt{1+\delta}}. \tag{54}$$

By using Assumption 3, we first bound the term $\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2$ by

$$\begin{aligned}\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \leq \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G^2 - \|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G^2 \\ - \|\boldsymbol{u}(t_k) - \boldsymbol{u}(t_{k+1})\|_G^2.\end{aligned} \tag{55}$$

Multiplying $\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})$ on both sides of (18) yields

$$\begin{aligned}&(R^T\boldsymbol{x}(t_{k+1}) - R^T\boldsymbol{x}^*(t_{k+1}))^T(\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})),\\ &= c(E_u(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})))^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1}))\\ &\quad - (E_o(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})))^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})).\end{aligned} \tag{56}$$

Based on Assumption 3 and $R^T\boldsymbol{x} = x$, we have

$$\begin{aligned}\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \leq c(E_u(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})))^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})),\\ - (E_o(\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}^*(t_{k+1})))^T\\ (\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})).\end{aligned} \tag{57}$$

Substituting (19) and (20) into (57), in view of the definition of G, it derives

$$\begin{aligned}&\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2\\ &\leq 2c(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1}))^T(\boldsymbol{z}(t_{k+1}) - \boldsymbol{z}(t_{k+1}))\\ &\quad + \frac{2}{c}(\boldsymbol{\alpha}(t_k) - \boldsymbol{\alpha}(t_{k+1}))^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1}))\\ &\leq 2(\boldsymbol{u}(t_k) - \boldsymbol{u}(t_{k+1}))^T G(\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})).\end{aligned} \tag{58}$$

For any $a, b, c \in \mathbb{R}^n$ and $n \times n$ matrix $A \succeq 0$, there is

$$(a-b)^T A(a-c) = 1/2\|a-c\|_A^2 + 1/2\|a-b\|_A^2 - 1/2\|b-c\|_A^2. \tag{59}$$

By using (58),(55) and (59), we have

$$\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \leq \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G^2 - \|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G^2 - \|\boldsymbol{u}(t_k) - \boldsymbol{u}(t_{k+1})\|_G^2.$$

Then, we will prove that the term $\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2$ satisfies

$$\begin{aligned}&\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \\ &\geq \delta\|\boldsymbol{u}(t_{k+1}) - \boldsymbol{u}^*(t_{k+1})\|_G^2 - \|\boldsymbol{u}(t_k) - \boldsymbol{u}(t_{k+1})\|_G^2.\end{aligned} \tag{60}$$

We observe $(\mu-1)\|a-b\|^2 \geq (1-1/\mu)\|b\|^2 - \|a\|^2$ valid for any $\mu > 1$, then we use (18) to obtain

$$(\mu-1)\|R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \geq \frac{\mu-1}{\mu}\|E_o^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1}))\|^2 - \|cE_u^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})\|^2. \tag{61}$$

The first term in (61) is bounded by

$$\begin{aligned}\|E_o^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1}))\|^2 &= (\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1}))^T E_o E_o^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})), \\ &\geq 2\lambda_{Lo0}\|\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned} \tag{62}$$

The second term in (61) is bounded by

$$\begin{aligned}\|cE_u^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1}))\|^2 &= c^2(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1}))^T E_u E_u^T(\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})), \\ &\leq 2c^2\lambda_{LuN}\|\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})\|^2.\end{aligned} \tag{63}$$

Substituting (62) and (63) into (61) and multiplying $\mu/(2c\lambda_{Lo0}(\mu-1))$ on both side of (61) yield

$$\begin{aligned}&\frac{\mu c\lambda_{LuN}}{(\mu-1)\lambda_{Lo0}}\|\boldsymbol{z}(t_k) - \boldsymbol{z}(t_{k+1})\|^2 + \frac{\mu}{2c\lambda_{Lo0}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - \mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2 \\ &\geq \frac{1}{c}\|\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned} \tag{64}$$

From (18), we have

$$\begin{aligned}&R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1})) - R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - cE_u^T(\boldsymbol{z}(t_k) - \boldsymbol{z}^*(t_{k+1})) \\ &= -cE_u^T(\boldsymbol{z}(t_{k+1}) - \boldsymbol{z}^*(t_{k+1})) - E_o^T(\boldsymbol{\alpha}(t_{k+1}) - \boldsymbol{\alpha}^*(t_{k+1})).\end{aligned}$$

Using the fact that for any a, b, c and $d \in \mathbb{R}$, there is

$$a + b = c + d \Rightarrow a^2 + b^2 \geq \frac{1}{4}c^2 - \frac{1}{2}d^2,$$

we can obtain

$$\begin{aligned}&\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2+2c^2\lambda_{LuN}\|\boldsymbol{z}(t_k)-\boldsymbol{z}^*(t_{k+1})\|^2\\&\geq\frac{1}{2}c^2\lambda_{Lu0}\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2-\lambda_{LoN}\|\boldsymbol{\alpha}(t_{k+1})-\boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned}\tag{65}$$

Using $\|\boldsymbol{z}(t_k)-\boldsymbol{z}(t_{k+1})\|^2+\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2\geq\|\boldsymbol{z}(t_k)-\boldsymbol{z}^*(t_{k+1})\|^2$, we have

$$\begin{aligned}&\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2+2c^2\lambda_{LuN}\|\boldsymbol{z}(t_k)-\boldsymbol{z}(t_{k+1})\|^2\\&\quad+2c^2\lambda_{LuN}\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2\\&\geq\frac{1}{2}c^2\lambda_{Lu0}\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2-\lambda_{LoN}\|\boldsymbol{\alpha}(t_{k+1})-\boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned}\tag{66}$$

Summing up (64) and (66), we obtain

$$\begin{aligned}&\Big(\frac{\mu c\lambda_{LuN}}{(\mu-1)\lambda_{Lo0}}+2c^2\lambda_{LuN}\Big)\|\boldsymbol{z}(t_k)-\boldsymbol{z}(t_{k+1})\|^2\\&\quad+\Big(\frac{\mu}{2c\lambda_{Lo0}}+1\Big)\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2\\&\geq\Big(\frac{c^2\lambda_{Lu0}}{2}-2c^2\lambda_{LuN}\Big)\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2+\Big(\frac{1}{c}-\lambda_{LoN}\Big)\|\boldsymbol{\alpha}(t_{k+1})-\boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned}\tag{67}$$

We set $c<\frac{1}{\lambda_{LoN}}$ and $\varsigma=\max\{\varsigma_1,\varsigma_2\}$, where

$$\varsigma_1=c/\left(\frac{c^2\lambda_{Lu0}}{2}-2c^2\lambda_{LuN}\right)\text{ and }\varsigma_2=\Big(\frac{1}{c}\Big)/\Big(\frac{1}{c}-\lambda_{LoN}\Big).$$

Multiplying both sides of (67) by ς yields

$$\begin{aligned}&\varsigma\Big(\frac{\mu c\lambda_{LuN}}{(\mu-1)\lambda_{Lo0}}+2c^2\lambda_{LuN}\Big)\|\boldsymbol{z}(t_k)-\boldsymbol{z}(t_{k+1})\|^2\\&\quad+\varsigma\Big(\frac{\mu}{2c\lambda_{Lo0}}+1\Big)\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2\\&\geq c\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2+\frac{1}{c}\|\boldsymbol{\alpha}(t_{k+1})-\boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned}$$

Based on the definition of δ in (40), we obtain

$$\begin{aligned}&\frac{c}{\delta}\|\boldsymbol{z}(t_k)-\boldsymbol{z}(t_{k+1})\|^2+\frac{\sigma_{\mathbf{F}}}{\delta}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})\|^2\\&\geq c\|\boldsymbol{z}(t_{k+1})-\boldsymbol{z}^*(t_{k+1})\|^2+\frac{1}{c}\|\boldsymbol{\alpha}(t_{k+1})-\boldsymbol{\alpha}^*(t_{k+1})\|^2.\end{aligned}$$

Recalling the following inequality:

$$\begin{aligned}&\sigma_{\mathbf{F}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}(t_{k+1}))-\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1}))\|^2\\&\geq\delta\|\boldsymbol{u}(t_{k+1})-\boldsymbol{u}^*(t_{k+1})\|^2-c\|\boldsymbol{u}(t_k)-\boldsymbol{u}(t_{k+1})\|^2,\end{aligned}$$

we can arrive at (60).

Summing up (55) and (60) yields (54).

Step 2: We prove the following inequality:

$$\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G \leq \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G + g(t_{k+1}). \tag{68}$$

By the triangle inequality, we obtain

$$\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G - \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G \leq \|\boldsymbol{u}^*(t_{k+1}) - \boldsymbol{u}^*(t_k)\|_G.$$

Combining with the definition of $\boldsymbol{u}(t_k)$, we obtain

$$\begin{aligned} &\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G - \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G \\ &\leq \sqrt{c}\|\boldsymbol{z}^*(t_{k+1}) - \boldsymbol{z}^*(t_k)\| + \frac{1}{\sqrt{c}}\|\boldsymbol{\alpha}^*(t_{k+1}) - \boldsymbol{\alpha}^*(t_k)\|. \end{aligned} \tag{69}$$

According to the consensus constraints in (4), we have $\boldsymbol{z}^*(t_k) = \mathbf{1}_m \otimes x^*(t_k)$. Similarly, using $\boldsymbol{x}^*(t_k) = \mathbf{1}_N \otimes x^*(t_k)$ yields

$$\|\boldsymbol{z}^*(t_{k+1}) - \boldsymbol{z}^*(t_k)\| = \frac{\sqrt{m}}{\sqrt{N}}\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}^*(t_k)\|. \tag{70}$$

By the KKT condition in (33), we have

$$\|R\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - R\mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\| = \|E_o^T(\boldsymbol{\alpha}^*(t_{k+1}) - \boldsymbol{\alpha}^*(t_k))\|,$$

which implies

$$\|\boldsymbol{\alpha}^*(t_{k+1}) - \boldsymbol{\alpha}^*(t_k)\| \leq \frac{1}{\sqrt{2\lambda_{Lo0}}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - \mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\|. \tag{71}$$

Substituting (70) and (71) into (69) yields (68) due to

$$\begin{aligned} &\|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_{k+1})\|_G - \|\boldsymbol{u}(t_k) - \boldsymbol{u}^*(t_k)\|_G \\ &\leq \frac{\sqrt{cm}}{\sqrt{N}}\|\boldsymbol{x}^*(t_{k+1}) - \boldsymbol{x}^*(t_k)\| + \frac{1}{\sqrt{2c\lambda_{Lo0}}}\|\mathbf{F}^{t_{k+1}}(\boldsymbol{x}^*(t_{k+1})) - \mathbf{F}^{t_k}(\boldsymbol{x}^*(t_k))\|. \end{aligned}$$

Using (54) and (68) in Step 1 and Step 2 yields (38).

This completes the proof. □

ACKNOWLEDGEMENT

This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0112600, in part by the National Natural Science Foundation of China under Grants 62103343.

(Received November 10, 2022)

REFERENCES

[1] D. Ardagna, B. Panicucci, and M. Passacantando: Generalized Nash equilibria for the service provisioning problem in cloud systems. IEEE Trans. Serv. Comput. *6* (2012), 429–442. DOI:10.1109/TSC.2012.14

[2] B. A. Bhatti and R. Broadwater: Distributed Nash equilibrium seeking for a dynamic micro-grid energy trading game with non-quadratic payoffs. Energy. *202* (2020), 117709. DOI:10.1016/j.energy.2020.117709

[3] H. Le Cadre, P. Jacquot, C. Wan and C. Alasseur: Peer-to-peer electricity market analysis: From variational to generalized Nash equilibrium. Eur. J. Oper. Res., *282* (2020), 753–771. DOI:10.1016/j.ejor.2019.09.035

[4] Z. Chen, J. Ma, S. Liang, and L. Li: Distributed Nash equilibrium seeking under quantization communication. Automatica *141* (2022), 110318. DOI:10.1016/j.automatica.2022.110318

[5] C. De Persis and S. Grammatico: Distributed averaging integral Nash equilibrium seeking on networks. Automatica *110* (2019), 1085448. DOI:10.1016/j.automatica.2019.108548

[6] B. Huang, C. Yang, Z. Meng, F. Chen, and W. Ren: Distributed nonlinear placement for multicluster systems: A time-varying Nash equilibrium-seeking approach. IEEE Trans. Cybernet. *52* (2022), 11614–11623. DOI:10.1109/TCYB.2021.3085583

[7] Z. Li, Z. Li, and Z. Ding: Distributed generalized Nash equilibrium seeking and its application to Femtocell networks. IEEE Trans. Cybern., *52* (2022), 2505–2517. DOI:10.1109/TCYB.2020.3004635

[8] X. Li, X. Li, Y. Hong, J. Chen, and L. Wang: A survey of decentralized online learning. arxiv preprint (2022). DOI:10.48550/arXiv.2205.00473

[9] Q. Ling and A. Ribeiro: Decentralized dynamic optimization through the alternating direction method of multipliers. IEEE Trans. Signal Process. *62* (2014), 1185–1197. DOI:10.1109/TSP.2013.2295055

[10] K. Lu, G. Jing, and L. Wang: Distributed algorithms for searching generalized Nash equilibrium of noncooperative games. IEEE Trans. Cybernet. *49* (2019), 2362–2371. DOI:10.1109/TCYB.2018.2828118

[11] K. Lu, H. Li, and L. Wang: Online distributed algorithms for seeking generalized Nash equilibria in dynamic environments. IEEE Trans. Autom. Control *66* (2020), 2289–2296. DOI:10.1109/TAC.2020.3002592

[12] M. Maskery, V. Krishnamurthy, and Q. Zhao: Decentralized dynamic spectrum access for cognitive radios: Cooperative design of a noncooperative game. IEEE Trans. Commun. *57* (2009), 459–469. DOI:10.1109/TCOMM.2009.02.070158

[13] M. Meng, X. Li, Y. Hong, J. Chen, and L. Wang: Decentralized online learning for noncooperative games in dynamic environments. arxiv preprint (2021). DOI:10.48550/arXiv.2105.06200

[14] A. M. Ospina, A. Simonetto, and E. Dall'Anese: Time-varying optimization of networked systems with human preferences. IEEE Trans. Control Netw. Syst. *10* (2023), 503–515. DOI:10.1109/TCNS.2022.3203467

[15] F. Salehisadaghiani and L. Pavel: Distributed Nash equilibrium seeking: A gossip-based algorithm. Automatica *72* (2016), 209–216. DOI:10.1016/j.automatica.2016.06.004

[16] A. Simonetto, A. Mokhtari, A. Koppel, G. Leus, and A. Ribeiro: A class of prediction-correction methods for time-varying convex optimization. IEEE Trans. Signal Process. *64* (2016), 4576–4591. DOI10.1109/TSP.2016.2568161

[17] Q. Tao, Y. Liu, C. Xian, and Y. Zhao: Prescribed-time distributed time-varying Nash equilibrium seeking for formation placement control. IEEE Trans. Circuits Syst., II, Exp. Briefs *69* (2022), 4423–4427. DOI:10.1109/TCSII.2022.3179576

[18] M. Ye and G. Hu: Distributed seeking of time-varying Nash equilibrium for non-cooperative games. IEEE Trans. Autom. Control *60* (2015), 3000–3005. DOI:10.1109/TAC.2015.2414817

[19] M. Ye and G. Hu: Distributed Nash equilibrium seeking by a consensus based approach. IEEE Trans. Autom. Control *62* (2017), 4811–4818. DOI:10.1109/TAC.2017.2688452

[20] X. Zeng, J. Chen, S. Liang, and Y. Hong: Generalized Nash equilibrium seeking strategy for distributed nonsmooth multi-cluster game. Automatica *103* (2019), 20–26. DOI:10.1016/j.automatica.2019.01.025

Ji Ma, The Department of Automation, Xiamen University, Xiamen, Fujian, 361000. P. R. China.
e-mail: maji08@xmu.edu.cn

Zheng Yang, The Department of Automation, Xiamen University, Xiamen, Fujian, 361000. P. R. China.
e-mail: yangzheng@stu.xmu.edu.cn

Ziqin Chen, Corresponding author, The Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634. U. S. A.
e-mail: chenziqin1993@gmail.com