Team-optimal distributed MMSE estimation in general and tree networks

Muhammed O. Sayin [a,*], Suleyman S. Kozat [b], Tamer Başar [a]

[a] *The Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Champaign, IL 61801, USA*
[b] *The Department of Electrical and Electronics Engineering, Bilkent University, Bilkent, Ankara 06800, Turkey*

A R T I C L E   I N F O

A B S T R A C T

We construct team-optimal estimation algorithms over distributed networks for state estimation in the finite-horizon mean-square error (MSE) sense. Here, we have a distributed collection of agents with processing and cooperation capabilities. These agents observe noisy samples of a desired state through a linear model and seek to learn this state by interacting with each other. Although this problem has attracted significant attention and been studied extensively in fields including machine learning and signal processing, all the well-known strategies do not achieve team-optimal learning performance in the finite-horizon MSE sense. To this end, we formulate the finite-horizon distributed minimum MSE (MMSE) when there is no restriction on the size of the disclosed information, i.e., oracle performance, over an arbitrary network topology. Subsequently, we show that exchange of local estimates is sufficient to achieve the oracle performance only over certain network topologies. By inspecting these network structures, we propose recursive algorithms achieving the oracle performance through the disclosure of local estimates. For practical implementations we also provide approaches to reduce the complexity of the algorithms through the time-windowing of the observations. Finally, in the numerical examples, we demonstrate the superior performance of the introduced algorithms in the finite-horizon MSE sense due to optimal estimation.

© 2017 Elsevier Inc. All rights reserved.

1. Introduction

Over a distributed network of agents with measurement, processing and communication capabilities, we can have enhanced processing performance, e.g., fast response time, relative to the centralized networks by distributing the processing power over the networks [1–4]. Mainly, distributed agents observe the true state of the system through noisy measurements from different viewpoints, process the observation data in order to estimate the state, and communicate with each other to alleviate the estimation process in a fully distributed manner. Notably, the agents can respond to streaming data in an online manner by disclosing information among each other at certain instances. This framework is conveniently used to model highly complex structures from defense applications to social and economical networks [5–8]. As an example, say that we have radar systems distributed over an area and seeking to locate hostile missiles, i.e., the location of the missile is the underlying state of the system. In that respect, distributed

processing approach has vital importance in terms of detecting the missiles and reacting as fast as possible. In particular, even if the viewpoints of a few radar systems are blocked due to environmental obstacles, through the communication among the radar systems, each system should still be able to locate the missiles. Additionally, since each radar system not only collects measurements but also process them to locate the missiles, the overall system can respond to the missiles faster than a centralized approach in which measurements of all the radar systems are collected at a centralized unit and processed together.

Although there is an extensive literature on this topic, e.g., [2, 6–11] and references therein, we still have significant and yet unexplored problems for disclosure and utilization of information among agents. Prior work has focused on the computationally simple algorithms that aim to minimize certain cost functions through the exchange of local estimates, e.g., diffusion or consensus based estimation algorithms [2,9,3,12–14], due to processing power related practical concerns. However, there is a trade-off in terms of computational complexity and estimation performance.

Formulating the optimal distributed estimation algorithms with respect to certain performance criteria is a significant and unexplored challenge. To this end, we consider here the distributed

---

* Corresponding author.
E-mail addresses: sayin2@illinois.edu (M.O. Sayin), kozat@ee.bilkent.edu.tr
(S.S. Kozat), basar1@illinois.edu (T. Başar).

estimation problem as a team problem for distributed agents in which agents take actions, e.g., which information to disclose and how to construct the local estimate. This differs from the existing approaches in which agents exchange their local estimates. Furthermore, we address the optimality of exchanging local estimates with respect to the team problem over arbitrary network structures.

We examine the optimal usage of the exchanged information based on its content rather than a blind approach in which exchanged information is handled irrespective of the content as in the diffusion or consensus based approaches. In such approaches, the agents utilize the exchanged information generally through certain static combination rules, e.g., the uniform rule [15], the Laplacian rule [16] or the Metropolis rule [17]. However, if the statistical profile of the measurement data varies over the network, i.e., each agent observes diverse signal-to-noise ratios, by ignoring the variation in noise, these rules yield severe degradation in the estimation performance [2]. In such cases the agents can perform better even without cooperation [2]. Therefore, the optimal usage of the exchanged information plays an essential role in performance improvement in the team problem.

Consider distributed networks of agents that observe noisy samples of an underlying state (possibly multi-dimensional) over a finite horizon. The agents can exchange information with only certain other agents at each time instant. In particular, agents cooperate with each other as a team according to a certain team cost depending on the agents' actions. To this end, each agent constructs a local estimate of the underlying state and constructs messages to disclose to the neighboring agents at each time instant. We particularly consider a quadratic cost function and that the underlying state and measurement noises are jointly Gaussian.

We note that restrictions on the sent messages, e.g., on the size of the disclosed information, have significant impact on the optimal team actions. We introduce the concept of the oracle performance, in which there is no restriction on the disclosed information. In that case, a time-stamped information disclosure can be team-optimal and we introduce the optimal distributed online learning (ODOL) algorithm using the time-stamped information disclosure. Through a counter example, we show that the oracle performance cannot be achieved through the exchange of local estimates in general. Then, we analytically show that over certain networks, e.g., tree networks, agents can achieve the oracle performance through the exchange of local estimates. We propose the optimal and efficient distributed online learning (OEDOL) algorithm, which is practical for real life applications and achieves the oracle performance over tree networks through the exchange of local estimates. Finally, we introduce the time windowing of the measurements in the team cost and propose a recursive algorithm, sub-optimal distributed online learning (SDOL) algorithm, combining the received messages linearly through time-invariant combination weights.

We can list our main contributions as follows: 1) We introduce a team-problem to minimize finite horizon mean square error cost function in a distributed manner. 2) We derive the ODOL algorithm achieving the oracle performance over arbitrary networks through time-stamped information exchange. 3) We address whether agents can achieve the oracle performance through the disclosure of local estimates. 4) We propose a recursive algorithm, the OEDOL algorithm, achieving the oracle performance over certain network topologies with tremendously reduced communication load. 5) We also formulate sub-optimal versions of the algorithms with reduced complexity. 6) We provide numerical examples demonstrating the significant gains due to the introduced algorithms.

The remainder of the paper is organized as follows. We introduce the team problem for distributed-MMSE estimation in Sec-
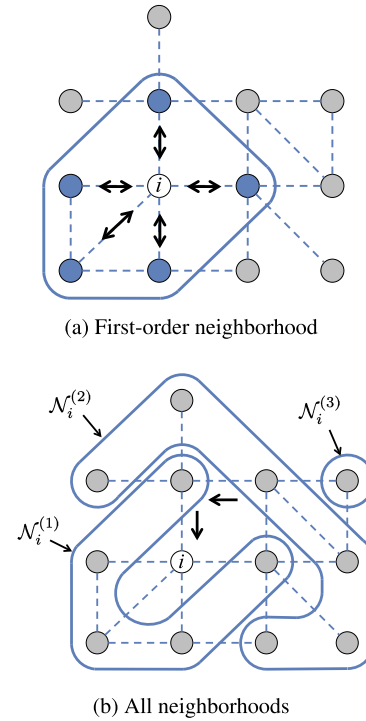

(a) First-order neighborhood

(b) All neighborhoods

Fig. 1. The neighborhoods of ith agent over the distributed network.

tion 2. We study the tree networks, exploit the network topology to formulate the OEDOL algorithm that reduces the communication load and introduce cell structures, which is relatively more connected than tree networks, in Section 3. We propose the suboptimal versions of the ODOL algorithm for practical implementations in Section 4. In Section 5, we provide numerical examples demonstrating significant gains due to the introduced algorithms. We conclude the paper in Section 6 with several remarks.

Notation: We work with real data for notational simplicity. $\mathbb{N}(0, .)$ denotes the multivariate Gaussian distribution with zero mean and designated covariance. For a vector a (or matrix A), a' (or A') is its ordinary transpose. We denote the vector whose terms are all 1s (or all 0s) by $\mathbf{1}$ (and $\mathbf{0}$). We denote random variables by bold lower case letters, e.g., $\mathbf{x}$. The operator col$\{\cdot\}$ produces a column vector or a matrix in which the arguments of col$\{\cdot\}$ are stacked one under the other. For a matrix A, diag$\{A\}$ operator constructs a diagonal matrix with the diagonal entries of A. For a given set $\mathcal{N}$, diag$\{\mathcal{N}\}$ creates a diagonal matrix whose diagonal block entries are elements of the set. The operator $\otimes$ denotes the Kronecker product.

2. Team problem for distributed-MMSE estimation

Consider a distributed network of m agents with processing and communication capabilities. In Fig. 1, we illustrate this network through an undirected graph, where the vertices and the edges correspond to the agents and the communication links across the network, respectively. For each agent i, we denote the set of agents whose information could be received at least after k hops, i.e., k-hop neighbors, by $\mathcal{N}_i^{(k)}$, and $\pi_i^{(k)} := \left| \mathcal{N}_i^{(k)} \right|$ is the cardinality of $\mathcal{N}_i^{(k)}$ (see Fig. 1b)[1]. We assume that $\mathcal{N}_i^{(0)} = \{i\}$ and $\mathcal{N}_i^{(k)} = \varnothing$ for $k < 0$. Note that the sequence of the sets $\{\mathcal{N}_i^{(0)}, \mathcal{N}_i^{(1)}, \ldots\}$ is a nondecreasing sequence such that $\mathcal{N}_i^{(k)} \subseteq \mathcal{N}_i^{(l)}$ if $k < l$.

---

[1] For notational simplicity, we define $\mathcal{N}_i := \mathcal{N}_i^{(1)}$ and $\pi_i := \pi_i^{(1)}$.

Here, at certain time instants, the agents observe a noisy version of a time-invariant and unknown state vector $x \in \mathbb{R}^p$ which is a realization of a Gaussian random variable $\boldsymbol{x}$ with mean $\bar{x}$ and auto-covariance matrix Σ_x. In particular, at time instant t, each agent i observes a noisy version of the state as follows:

$$y_{i,t} = H_i x + n_{i,t}, \ t = 1, \ldots, T,$$

where $H_i \in \mathbb{R}^{q \times p}$ is a matrix commonly known by all agents, and $n_{i,t} \in \mathbb{R}^q$ is a realization of a zero-mean white Gaussian vector process $\{\boldsymbol{n}_{i,t}\}$ with auto-covariance Σ_{n_i}. Correspondingly, the observation $y_{i,t} \in \mathbb{R}^q$ is a realization of the random process $\{\boldsymbol{y}_{i,t}\}$, where $\boldsymbol{y}_{i,t} = H_i \boldsymbol{x} + \boldsymbol{n}_{i,t}$ almost everywhere (a.e.). The noise $\boldsymbol{n}_{i,t}$ is also independent of the state $\boldsymbol{x}$ and the other noise parameters $\boldsymbol{n}_{j,\tau}$, $j \neq i$ and $\tau \leq t$. We assume that the statistical profiles of the noise processes are common knowledge of the agents since they can readily be estimated from the data [18].

The agents have communication capabilities and at certain time instants, i.e., after each measurement, they can exchange information with the neighboring agents as seen in Fig. 1a. Let $z_{i,j,t} \in \mathbb{R}^r$ denote the information disclosed by i to j at time t, and $r \geq 0$ is the size of the disclosed information. We assume that there exists a perfect channel between the agents such that the disclosed information can be transmitted with infinite precision. Therefore, we denote the information available to agent-i at time t by

$$\delta_{i,t} = \left\{ y_{i,t}, y_{i,\tau}, z_{j,i,\tau}, \text{ for } j \in \mathcal{N}_i, \tau = 1, \ldots, t-1 \right\}$$

and let $\sigma_{i,t}$ denote the sigma-algebra generated by the information set $\delta_{i,t}$. Furthermore, we define the set of all $\sigma_{i,t}$-measurable functions from $\mathbb{R}^{qt} \times \mathbb{R}^{r\pi_i(t-1)}$ to $\mathbb{R}^r$ by $\Gamma_{i,t}$. Importantly, here, which information to disclose is not determined a priori in the problem formulation. Let $\gamma_{i,j,t}$ be the decision strategy for $z_{i,j,t}$, then agent-i chooses $\gamma_{i,j,t}$, $j \in \mathcal{N}_i$, from the set $\Gamma_{i,t}$, i.e., $\gamma_{i,j,t} \in \Gamma_{i,t}$ and $\gamma_{i,j,t}(\delta_{i,t}) = z_{i,j,t}$, based on his/her objective.

In addition to the disclosed information $z_{i,j,t}$, $j \in \mathcal{N}_i$, agent-i takes action $u_{i,t} \in \mathbb{R}^q$, where the corresponding decision strategy $\eta_{i,t}$ is chosen from the set $\Omega_{i,t}$, which is the set of all $\sigma_{i,t}$-measurable functions from $\mathbb{R}^{qt} \times \mathbb{R}^{r\pi_i(t-1)}$ to $\mathbb{R}^p$, i.e., $\eta_{i,t} \in \Omega_{i,t}$ and $\eta_{i,t}(\delta_{i,t}) = u_{i,t}$. Here, we consider that the agents have a common cost function:

$$\sum_{t=1}^{T} \sum_{j=1}^{m} \|x - u_{j,t}\|^2,$$

where all actions $u_{i,t}$, $i = 1, \ldots, m$ and $t = 1, \ldots, T$ are costly, and agent-i should take actions $u_{i,t}$ and $z_{i,j,t}$, $j = 1, \ldots, \pi_i$ and $t = 1, \ldots, T$, accordingly. Therefore, this corresponds to a team-problem, in which agent-i faces the following minimization problem:

$$\min_{\substack{\gamma_{i,j,t} \in \Gamma_{i,t}, \eta_{i,t} \in \Omega_{i,t}, \\ j \in \mathcal{N}_i, t=1,\ldots,T}} \sum_{t=1}^{T} \sum_{j=1}^{m} E\|\boldsymbol{x} - \eta_{j,t}(\delta_{j,t})\|^2. \tag{1}$$

We point out that both $\Gamma_{i,t}$ and $\Omega_{i,t}$ are infinite dimensional, i.e., (1) is a functional optimization problem and the optimal strategies can be a nonlinear function of the available information. Furthermore, the agents should also construct the disclosed information accordingly since other agents' decisions $u_{j,t}$ directly depend on the disclosed information.

2.1. A lower bound

In order to compute the team optimal strategies, we first construct a lower bound on the performance of the agents by removing the limitation on the size of the disclosed information, i.e., $r \to \infty$. In that case, the following proposition provides an optimal information disclosure strategy.

$$\boxed{4\ 3\ 2\ 1\ 2\ 1} \equiv [y'_{A,4}, y'_{B,3}, y'_{C,2}, y'_{D,1}, y'_{E,2}, y'_{F,1}]$$
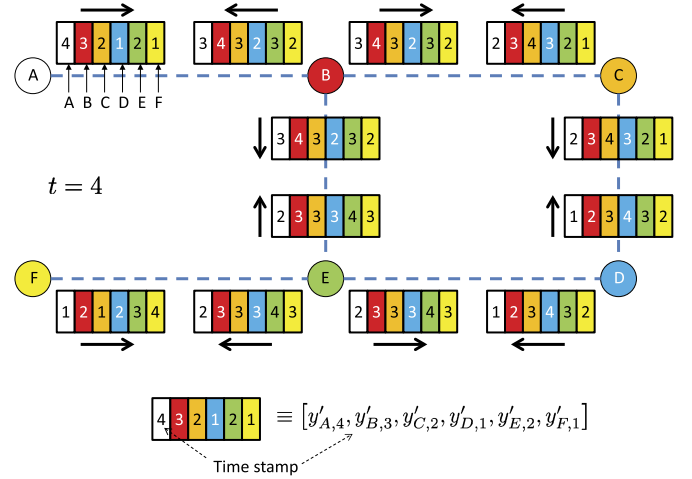
Time stamp

Fig. 2. Time stamped information disclosure over a network of 6 agents at time instant $t = 4$.

Proposition 2.1. *When $r \geq mq$, a time stamped information disclosure strategy, in which agents transmit the most current version of the available information (e.g., see Fig. 2), can lead to the team-optimal solution.*

Proof. Through the time stamped information disclosure, each agent can obtain the measurements of the other agents separately in a connected network. However, the measurements of the non-neighboring agents could only be received after certain hops due to the partially connected structure, i.e., certain agents are not directly connected. As an example, the disclosed information of $j \in \mathcal{N}_i^{(2)}$ reaches to i by passing through two communication links as seen in Fig. 1b. In particular, this case assumes that each agent has access to full information from the other agents, albeit with certain hops, and corresponds to the direct aggregation of all measurements across the network at each agent.

Correspondingly, at time t, all the information aggregated at ith agent is given by

$$\delta_{i,t}^o := \left\{ \{y_{i,\tau}\}^{\tau \leq t}, \{y_{j,\tau}\}_{j \in \mathcal{N}_i^{(1)}}^{\tau \leq t-1}, \ldots, \{y_{j,\tau}\}_{j \in \mathcal{N}_i^{(\kappa_i)}}^{\tau \leq t-\kappa_i} \right\}, \tag{2}$$

where the information from the furthest agent is received at least after κ_i hops. Therefore, a time-stamped information disclosure strategy can lead to the team-optimal solution. $\square$

Let $\sigma_{i,t}^o$ denote the sigma-algebra generated by the information set $\delta_{i,t}^o$ and $\Omega_{i,t}^o$ be the set of all $\sigma_{i,t}^o$-measurable functions from $\mathbb{R}^{qt} \times \mathbb{R}^{q\pi_i(t-1)} \times \ldots \times \mathbb{R}^{q\pi_i^{\kappa_i}|t-\kappa_i|_+}$ (where $|t - \kappa_i|_+ = 0$ if $t - \kappa_i < 0$) to $\mathbb{R}^p$. Then, agent-i faces the following minimization problem:

$$\min_{\substack{\eta_{i,t} \in \Omega_{i,t}^o, \\ t=1,\ldots,T}} \sum_{t=1}^{T} \sum_{j=1}^{m} E\|\boldsymbol{x} - \eta_{j,t}(\delta_{j,t}^o)\|^2, \tag{3}$$

which is equivalent to

$$\sum_{t=1}^{T} \min_{\eta_{i,t} \in \Omega_{i,t}^o} E\|\boldsymbol{x} - \eta_{i,t}(\delta_{i,t}^o)\|^2 \tag{4}$$

since $\gamma_{i,j,t}$, $j \in \mathcal{N}_i$, is set to the time-stamped strategy and $\eta_{i,t}$ has impact only on the term $E\|\boldsymbol{x} - \eta_{i,t}(\delta_{i,t}^o)\|^2$. Let $\delta_{i,t}^o$ be defined by

$$\delta_{i,t}^o := \left\{ \{\boldsymbol{y}_{i,\tau}\}^{\tau \leq t}, \{\boldsymbol{y}_{j,\tau}\}_{j \in \mathcal{N}_i^{(1)}}^{\tau \leq t-1}, \ldots, \{\boldsymbol{y}_{j,\tau}\}_{j \in \mathcal{N}_i^{(\kappa_i)}}^{\tau \leq t-\kappa_i} \right\}$$

Then, team optimal strategy in the lower bound, i.e., oracle strategy, $\eta_{i,t}^o$ and the corresponding action $u_{i,t}^o$ are given by

$$\eta_{i,t}^o(\delta_{i,t}^o) = u_{i,t}^o = E[\boldsymbol{x}|\delta_{i,t}^o = \delta_{i,t}^o] \qquad (5)$$

and we define $\boldsymbol{u}_{i,t}^o := E[\boldsymbol{x}|\delta_{i,t}^o]$.

2.2. ODOL algorithm

Since the state and the observation noise are jointly Gaussian random parameters, we can compute (5) through a Kalman-like recursion [19]. Therefore, we provide the following ODOL algorithm. We introduce a difference set $\Delta_{i,t} := \delta_{i,t}^o \backslash \delta_{i,t-1}^o = \left\{ y_{i,t}, \{y_{j,t-1}\}_{j \in \mathcal{N}_i^{(1)}}, \dots, \{y_{j,t-\kappa_i}\}_{j \in \mathcal{N}_i^{(\kappa_i)}} \right\}$ and a vector $w_{i,t} = \mathrm{col}\{\Delta_{i,t}\}$. Then, for $t \geq 1$ the iterations of the ODOL algorithm are given by

$$u_{i,t}^o = \left(I - K_{i,t}\bar{H}_i\right)u_{i,t-1}^o + K_{i,t}w_{i,t},$$

$$K_{i,t} = \hat{\Sigma}_{i,t-1}\bar{H}_i'\left(\bar{H}_i\hat{\Sigma}_{i,t-1}\bar{H}_i' + \bar{\Sigma}_{n_i}\right)^{-1},$$

$$\hat{\Sigma}_{i,t} = \left(I - K_{i,t}\bar{H}_i\right)\hat{\Sigma}_{i,t-1},$$

where[2] $u_{i,0}^o = \bar{x}$, $\hat{\Sigma}_{i,0} = \Sigma_x$, $\bar{H}_i := (P_i \otimes I_q)H$,

$$H := \mathrm{col}\{H_1, \dots, H_m\},$$

$\bar{\Sigma}_{n_i} := (P_i \otimes I_p)\Sigma_n(P_i \otimes I_p)'$, $\Sigma_n := \mathrm{diag}\{\Sigma_{n_1}, \dots, \Sigma_{n_m}\}$, and P_i is the corresponding permutation matrix.

We point out that this is a lower bound on the original cost function (1), i.e.,

$$\sum_{t=1}^T \sum_{i=1}^m E\|\boldsymbol{x} - u_{i,t}^o\|^2 \leq \min_{\substack{\gamma_{i,j,t} \in \Gamma_{i,t}, \eta_{i,t} \in \Omega_{i,t}, \\ i=1,\dots,m; \\ j \in \mathcal{N}_i; t=1,\dots,T}} \sum_{t=1}^T \sum_{j=1}^m E\|\boldsymbol{x} - \eta_{j,t}(\delta_{j,t})\|^2,$$

$$(6)$$

where we substitute team-optimal action (when $r \to \infty$) $u_{i,t}^o$ back into (4) and sum over $t = 1, \dots, T$ and $i = 1, \dots, m$. However, the lower bound is not necessarily tight depending on r. By Proposition 2.1, time-stamped information disclosure strategy, in which the size of the disclosed information is $q \times m$, yields the oracle solution. This implies that when $r \geq qm$, the lower bound is tight. Furthermore, team optimal solutions are linear in the available information and can be constructed through the recursive algorithm ODOL. However, qm is linear in the number of agents, m, and in large networks this can cause excessive communication load yet communication load is crucial for the applicability of the distributed learning algorithms [14,13]. Therefore, in the following section, we provide a sufficient condition on the size of the disclosed information, which depends on the network structure (rather than its size), in order to achieve the lower bound (6).

3. Distributed-MMSE estimation with disclosure of local estimate

In the conventional distributed estimation algorithms, e.g., consensus and diffusion approaches, agents disclose their local estimates, which have size p (note that this does not depend on the network size). The following example addresses whether the disclosure of local estimates can achieve the lower bound (6) or not.
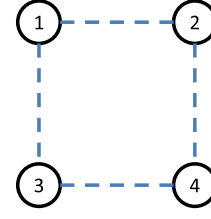

Fig. 3. A cycle network of 4 agents.

3.1. A counter example

Consider a cycle network of 4 agents as seen in Fig. 3, where $p = q = 1$, $H_i = 1$, $\Sigma_{n_i} = \sigma_n^2$, for $i = 1, \dots, 4$, and $\Sigma_x = \sigma_x^2$. We aim to show that agent-1's oracle action at time $t = 3$, i.e., $u_{1,3}^o$, cannot be constructed through the exchange of local estimates.

At time $t = 2$, agent-2 and agent-3 have the following oracle actions:

$$u_{2,2}^o = E[\boldsymbol{x}|\boldsymbol{y}_{2,2} = y_{2,2}, \boldsymbol{y}_{2,1} = y_{2,1}, \boldsymbol{y}_{1,1} = y_{1,1}, \boldsymbol{y}_{4,1} = y_{4,1}]$$

$$= \frac{\sigma_x^2}{4(\sigma_x^2 + \sigma_n^2)}(y_{2,2} + y_{2,1} + y_{1,1} + y_{4,1}), \qquad (7)$$

$$u_{3,2}^o = E[\boldsymbol{x}|\boldsymbol{y}_{3,2} = y_{3,2}, \boldsymbol{y}_{3,1} = y_{3,1}, \boldsymbol{y}_{1,1} = y_{1,1}, \boldsymbol{y}_{4,1} = y_{4,1}]$$

$$= \frac{\sigma_x^2}{4(\sigma_x^2 + \sigma_n^2)}(y_{3,2} + y_{3,1} + y_{1,1} + y_{4,1}). \qquad (8)$$

Note that since there are two hops between agents 2 and 3, at $t = 2$, agents do not have access to each other's any measurement yet. At time $t = 3$, agent-1's oracle action is given by

$$u_{1,3}^o = E[\boldsymbol{x}|\boldsymbol{y}_{1,3} = y_{1,3}, \boldsymbol{y}_{1,2} = y_{1,2}, \boldsymbol{y}_{1,1} = y_{1,1},$$

$$\boldsymbol{y}_{2,2} = y_{2,2}, \boldsymbol{y}_{2,1} = y_{2,1},$$

$$\boldsymbol{y}_{3,2} = y_{3,2}, \boldsymbol{y}_{3,1} = y_{3,1},$$

$$\boldsymbol{y}_{4,1} = y_{4,1}],$$

$$= \frac{\sigma_x^2}{8(\sigma_x^2 + \sigma_n^2)}(y_{1,3} + y_{1,2} + y_{1,1} + y_{2,2} + y_{2,1} + y_{3,2}$$

$$+ y_{3,1} + y_{4,1}).$$

Assume that $u_{1,3}^o$ can be obtained through the exchange of local estimates:

$$\hat{u}_{1,3} := E[\boldsymbol{x}|\boldsymbol{y}_{1,3} = y_{1,3}, \boldsymbol{y}_{1,2} = y_{1,2}, \boldsymbol{y}_{1,1} = y_{1,1},$$

$$\boldsymbol{u}_{2,2}^o = u_{2,2}^o, \boldsymbol{u}_{2,1}^o = u_{2,1}^o,$$

$$\boldsymbol{u}_{3,2}^o = u_{3,2}^o, \boldsymbol{u}_{3,1}^o = u_{3,1}^o]. \qquad (9)$$

Since all parameters are jointly Gaussian, the local estimates are also jointly Gaussian, $\hat{u}_{1,3}$, is linear in $u_{2,2}^o$ and $u_{3,2}^o$. Furthermore, the measurements $y_{2,2}$, $y_{3,2}$, and $y_{4,1}$ are only included in $u_{2,2}^o$ and $u_{3,2}^o$. Therefore, we obtain

$$\hat{u}_{1,3} = \dots + \alpha u_{2,2}^o + \beta u_{3,2}^o$$

$$= \dots + \alpha \frac{\sigma_x^2}{4(\sigma_x^2 + \sigma_n^2)}(y_{2,2} + y_{4,1} + \dots)$$

$$+ \beta \frac{\sigma_x^2}{4(\sigma_x^2 + \sigma_n^2)}(y_{3,2} + y_{4,1} + \dots),$$

where $\dots$ refers to the other terms. However, the equality of $\hat{u}_{1,3}$ and $u_{1,3}^o$ implies $\alpha = \beta = 1/2$ due to the combination weights of $y_{2,2}$ and $y_{3,2}$, respectively, and $\alpha + \beta = 1/2$ due to the combination weight of $y_{4,1}$, which leads to a contradiction. Hence, which information to disclose over arbitrary networks for team-optimal

---

[2] If the inverse fails to exist, a pseudo-inverse can replace the inverse [19].
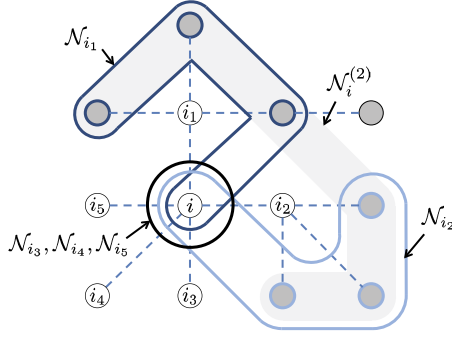
Fig. 4. An example tree network. Notice the eliminated links from Fig. 1 to avoid multi-path information diffusion.

solutions should be considered elaborately. In the following, we analytically show that lower bound could be achieved through the disclosure of local estimates over "tree networks".

3.2. Tree networks

A network has a "tree structure" if its corresponding graph is a *tree*, i.e., connected and undirected without any cycles [20]. As an example, the conventional star or line networks have tree structures. We remark that for an arbitrary network topology we can also construct the spanning tree of the network and eliminate the cycles. In the literature, there exists numerous distributed algorithms for minimum spanning tree construction [21–25].

Importantly, the following theorem shows that over tree networks we can achieve the performance of the oracle algorithm through the disclosure of the local estimates only.

Theorem 3.1. *Consider the team-problem over a tree network, in which* $r = p$. *Then, exchange of local estimates can lead to the team-optimal solution, i.e., agents can achieve the oracle performance.*

Proof. Initially, agent-i has access to $y_{i,1}$ only and the oracle action is $u_{i,1}^o = E[\mathbf{x}|\mathbf{y}_{i,1} = y_{i,1}]$. At time $t = 2$, the oracle action is given by

$$u_{i,2}^o = E\left[\mathbf{x} \middle| \left\{\mathbf{y}_{i,\tau} = y_{i,\tau}\right\}_{\tau=1,2}, \left\{\mathbf{y}_{j,1} = y_{j,1}\right\}_{j \in \mathcal{N}_i}\right], \quad (10)$$

which can be written as

$$u_{i,2}^o = E\left[\mathbf{x} \middle| \left\{\mathbf{y}_{i,\tau} = y_{i,\tau}\right\}_{\tau=1,2}, \left\{E[\mathbf{x}|\mathbf{y}_{j,1} = y_{j,1}]\right\}_{j \in \mathcal{N}_i}\right]$$

$$= E\left[\mathbf{x} \middle| \left\{\mathbf{y}_{i,\tau} = y_{i,\tau}\right\}_{\tau=1,2}, \left\{\mathbf{u}_{j,1}^o = u_{j,1}^o\right\}_{j \in \mathcal{N}_i}\right]. \quad (11)$$

This implies that for $t = 1$ and $t = 2$, the oracle performance can be achieved through the disclosure of local estimate. Therefore, we can consider the oracle action (10) even though agents disclose their local estimate instead of time-stamped information disclosure.

As seen in Fig. 4, over a tree network, for $k \in \{1, \ldots, \kappa_i\}$ we have

$$\mathcal{N}_i^{(k)} = \bigcup_{j \in \mathcal{N}_i} \left(\mathcal{N}_i^{(k)} \cap \mathcal{N}_j^{(k-1)}\right). \quad (12)$$

Note that the sets in (12) are disjoint as

$$\left(\mathcal{N}_i^{(k)} \cap \mathcal{N}_{j_1}^{(k-1)}\right) \cap \left(\mathcal{N}_i^{(k)} \cap \mathcal{N}_{j_2}^{(k-1)}\right) = \varnothing \quad (13)$$

for all $j_1, j_2 \in \mathcal{N}_i$ and $j_1 \neq j_2$. Notably, over a tree network, by (13), we can partition the collection set of the measurements received after at least k-hops as follows

$$\left\{y_{j,\tau}\right\}_{j \in \mathcal{N}_i^{(k)}} = \left\{ \left\{y_{j,\tau}\right\}_{j \in \mathcal{N}_i^{(k)} \cap \mathcal{N}_{j_1}^{(k-1)}}, \right.$$

$$\left. \ldots, \left\{y_{j,\tau}\right\}_{j \in \mathcal{N}_i^{(k)} \cap \mathcal{N}_{j_{\pi_i}}^{(k-1)}} \right\}. \quad (14)$$

In the time-stamped information disclosure, at time $t = 3$, agent-i has access to $\delta_{i,3}^o$, defined in (2). We denote the set of new measurements received by i over j at time $t = 2$ by

$$\Delta_{j,i,2}^o := \left\{ \overbrace{\left\{y_{k,2}\right\}_{k \in \mathcal{N}_i^{(1)} \cap \mathcal{N}_j^{(0)}}}^{= y_{j,2}}, \left\{y_{k,1}\right\}_{k \in \mathcal{N}_i^{(2)} \cap \mathcal{N}_j^{(1)}} \right\},$$

which can also be written as

$$\Delta_{j,i,2}^o = \delta_{j,2}^o \setminus \left\{ \overbrace{y_{j,1}}^{= \delta_{j,1}^o}, \underbrace{y_{i,1}}_{= \Delta_{i,j,1}^o = \delta_{i,1}^o} \right\}, \quad (15)$$

where we exclude the information sent by i to j at time $t = 1$, i.e., $y_{i,1}$. Then, we can write the accessed information as the union of new measurement $y_{i,3}$, new measurements received over the neighboring agents and the accessed information at time $t = 2$ as follows:

$$\delta_{i,3}^o = \left\{ y_{i,3}, \Delta_{j_1,i,2}^o, \ldots, \Delta_{j_{\pi_i},i,2}^o, \delta_{i,2}^o \right\}. \quad (16)$$

Note that the sets on the right hand side of (16) are disjoint due to tree structure. Furthermore, by (15) and (16), the sigma-algebra generated by $\delta_{i,3}^o$ is equivalent to the sigma-algebra generated by the set $\left\{\{y_{i,\tau}\}^{\tau \leq 3}, \{\delta_{j,\tau}^o\}_{j \in \mathcal{N}_i}^{\tau \leq 2}\right\}$. Since $u_{j,t}^o = E[\mathbf{x}|\delta_{j,t}^o = \delta_{j,t}^o]$, we obtain

$$u_{i,3}^o = E\left[\mathbf{x} \middle| \left\{\mathbf{y}_{i,\tau} = y_{i,\tau}\right\}^{\tau \leq 3}, \left\{\mathbf{u}_{j,\tau}^o = u_{j,\tau}^o\right\}_{j \in \mathcal{N}_i}^{\tau \leq 2}\right]. \quad (17)$$

By (15), we have

$$\Delta_{j,i,t}^o = \delta_{j,t}^o \setminus \left\{ \delta_{j,t-1}^o \cup \Delta_{i,j,t-1}^o \right\}, \quad (18)$$

which implies that for $t \geq 2$, $\Delta_{j,i,t}^o$ is constructible from $\delta_{i,\tau}^o$ and $\delta_{j,\tau}^o$ for $\tau \leq t$. Hence, by induction, we conclude that the lower bound can be achieved through the exchange of local estimates. □

Remark 3.1. When the expectation of the state is conditioned on infinite number of observations over even a constructed spanning tree, only a finite number of the observations is missing compared to the case over a fully connected network. Hence, even if we construct the spanning tree of that network, we would still achieve the lower bound over a fully connected (or centralized) network asymptotically. As an illustrative example, in Fig. 10, we observe that the MMSE performance over the fully connected, star and line networks are asymptotically the same. Similarly, in [26–28], the authors show that the performance of the diffusion based algorithms could approach the performance of a fully connected network under certain regularity conditions.

In the sequel, we propose the OEDOL algorithm that achieves the lower bound over tree networks iteratively.

3.3. OEDOL algorithm

By Theorem 3.1, over a tree network, oracle action can be constructed by

$$u_{i,t}^o = E\left[\boldsymbol{x}\middle| \{\boldsymbol{y}_{i,\tau} = y_{i,\tau}\}^{\tau \le t}, \left\{\boldsymbol{u}_{j,\tau}^o = u_{j,\tau}^o\right\}_{j \in \mathcal{N}_i}^{\tau \le t-1}\right] \tag{19}$$

through the disclosure of oracle actions, i.e., local estimates. We remark that $u_{i,t}^o$ is linear in the previous actions $u_{i,\tau}^o$, $\tau \le t-1$. In order to extract new information, i.e., innovation part, we need to eliminate the previously received information at each instant on the neighboring agents. This brings in additional computational complexity. On the contrary, agents can just send the new information compared to the previously sent information, e.g., $s_{i,t}$. Note that here agents disclose the same information to the neighboring agents. Since we are conditioning on the linear combinations of the conditioned variables without effecting their spanned space, i.e., $s_{i,t}$ is computable from $u_{i,\tau}^o$ for $\tau \le t$ and vice versa, agents can still achieve the oracle performance by reduced computational load, yet.

At time t, agent-i receives local measurement $y_{i,t}$ and sent information from the neighboring agents,

$$r_{i,t} := \mathrm{col}\{s_{j_1,t-1}, \ldots, s_{j_{\pi_i},t-1}\}.$$

We aim to determine the content of the received information $r_{i,t}$ to extract the innovation within them and utilize this innovation in the update of the oracle action.

Initially, at time $t = 1$, agent-i has only access to the local measurement $y_{i,1}$. Then, the oracle action is given by

$$u_{i,1}^o = (I - \Sigma_x H_i'(H_i \Sigma_x H_i' + \Sigma_{n_i})^{-1} H_i)\bar{x}$$
$$+ \Sigma_x H_i'(H_i \Sigma_x H_i' + \Sigma_{n_i})^{-1} y_{i,1}.$$

Let $u_{i,0}^o = \bar{x}$ and $\hat{\Sigma}_{i,0} = \Sigma_x$, and set $B_{i,1} = \hat{\Sigma}_{i,0} H_i'(H_i \hat{\Sigma}_{i,0} H_i' + \Sigma_{n_i})^{-1}$ and $A_{i,1} = I - B_{i,1} H_i$. Then, we obtain

$$u_{i,1}^o = A_{i,1} u_{i,0}^o + B_{i,1} y_{i,1},$$
$$\hat{\Sigma}_{i,1} = A_{i,1} \hat{\Sigma}_{i,0}.$$

Next, instead of sending $u_{i,1}^o$, agent-i sends to the neighboring agents, $j \in \mathcal{N}_i$,

$$s_{i,1} = u_{i,1}^o - A_{i,1} u_{i,0}^o$$
$$= B_{i,1} y_{i,1}.$$

Correspondingly, at time $t = 2$, agent-i receives $y_{i,2}$ and $r_{i,2}$. Let $\boldsymbol{r}_{i,2}$ be the corresponding random vector. Then, conditioning the state and the received information on the previously available information $\boldsymbol{y}_{i,1} = y_{i,1}$, we have

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y}_{i,2} \\ \boldsymbol{r}_{i,2} \end{bmatrix} \middle| \boldsymbol{y}_{i,1} = y_{i,1}$$
$$\sim \mathbb{N}\left(\begin{bmatrix} u_{i,1}^o \\ H_i u_{i,1}^o \\ \bar{H}_{i,1} u_{i,1}^o \end{bmatrix}, \begin{bmatrix} \hat{\Sigma}_{i,1} & \hat{\Sigma}_{i,1} H_i' & \hat{\Sigma}_{i,1} \bar{H}_{i,1}' \\ H_i \hat{\Sigma}_{i,1} & H_i \hat{\Sigma}_{i,1} H_i' + \Sigma_{n_i} & H_i \hat{\Sigma}_{i,1} \bar{H}_{i,1}' \\ \bar{H}_{i,1} \hat{\Sigma}_{i,1} & \bar{H}_{i,1} \hat{\Sigma}_{i,1} H_i' & \bar{H}_{i,1} \hat{\Sigma}_{i,1} \bar{H}_{i,1}' + \bar{G}_{i,1} \end{bmatrix} \right),$$

where $\bar{H}_{i,1} := \mathrm{col}\left\{ B_{j_1,1} H_{j_1}, \ldots, B_{j_{\pi_i},1} H_{j_{\pi_i}} \right\}$ and $\bar{G}_{i,1} = \mathrm{diag}\{G_{i,1}\}$, where $G_{i,1} := \mathrm{col}\left\{ B_{j_1,1} \Sigma_{n_{j_1}} B_{j_1,1}', \ldots, B_{j_{\pi_i},1} \Sigma_{n_{j_{\pi_i}}} B_{j_{\pi_i},1}' \right\}$.

Let $\tilde{H}_{i,1} := \mathrm{col}\{H_i, \bar{H}_{i,1}\}$ and $\tilde{G}_{i,1} := \mathrm{diag}\{\Sigma_{n_i}, \bar{G}_{i,1}\}$ and set

$$\begin{bmatrix} B_{i,2} & C_{i,2} \end{bmatrix} = \hat{\Sigma}_{i,1} \tilde{H}_{i,1}' \left(\tilde{H}_{i,1} \hat{\Sigma}_{i,1} \tilde{H}_{i,1}' + \tilde{G}_{i,1} \right)^{-1},$$
$$A_{i,2} = I - B_{i,2} H_i - C_{i,2} \bar{H}_{i,1}.$$

Then, we obtain

$$u_{i,2}^o = A_{i,2} u_{i,1}^o + B_{i,2} y_{i,2} + C_{i,2} r_{i,2}, \tag{20}$$
$$\hat{\Sigma}_{i,2} = A_{i,2} \hat{\Sigma}_{i,1}$$

and agent-i sends

$$s_{i,2} = u_{i,2}^o - A_{i,2} u_{i,1}^o$$
$$= B_{i,2} y_{i,2} + \sum_{j \in \mathcal{N}_i} C_{i,2}^{(j)} \underbrace{B_{j,1} y_{j,1}}_{= s_{j,1}},$$

where $C_{i,2}^{(j)}$ denotes the corresponding jth block of $C_{i,2}$. Therefore, at time $t = 3$, agent-i receives from $j \in \mathcal{N}_i$:

$$s_{j,2} = B_{j,2} y_{j,2} + \sum_{k \in \mathcal{N}_j \setminus i} (C_{j,2}^{(k)} B_{j,1} y_{j,1}) + C_{j,2}^{(i)} B_{i,1} y_{i,1}. \tag{21}$$

Since the last term on the right hand side of (21) is known by i, we have

$$E[\boldsymbol{r}_{i,2}|\delta_{i,2}^o = \delta_{i,2}^o] = \overbrace{\begin{bmatrix} B_{j_1,2} + \sum_{k \in \mathcal{N}_{j_1} \setminus i} C_{j_1,2}^{(k)} B_{k,2} \\ \vdots \\ B_{j_{\pi_i},2} + \sum_{k \in \mathcal{N}_{j_{\pi_i}} \setminus i} C_{j_{\pi_i},2}^{(k)} B_{k,2} \end{bmatrix}}^{= \bar{H}_{i,2}} u_{i,2}^o + D_{i,2} s_{i,1}, \tag{22}$$

where $D_{i,2} := \mathrm{col}\{C_{j_1,2}^{(i)}, \ldots, C_{j_{\pi_i},2}^{(i)}\}$, and

$$G_{i,2} = \begin{bmatrix} B_{j_1,2} \Sigma_{n_{j_1}} B_{j_1,2}' + \sum_{k \in \mathcal{N}_{j_1} \setminus i} C_{j_1,2}^{(k)} B_{k,1} \Sigma_{n_k} B_{k,1}' (C_{j_1,2}^{(k)})' \\ \vdots \\ B_{j_{\pi_i},2} \Sigma_{n_{j_{\pi_i}}} B_{j_{\pi_i},2}' + \sum_{k \in \mathcal{N}_{j_{\pi_i}} \setminus i} C_{j_{\pi_i},2}^{(k)} B_{k,1} \Sigma_{n_k} B_{k,1}' (C_{j_{\pi_i},2}^{(k)})' \end{bmatrix} \tag{23}$$

By (20), (22), and (23), the next oracle action $u_{i,3}^o$ is given by

$$u_{i,3}^o = A_{i,3} u_{i,2}^o + B_{i,3} y_{i,3} + C_{i,3}(r_{i,3} - D_{i,2} s_{i,1}).$$

Subsequently, agent-i sends $s_{i,3} = u_{i,3}^o - A_{i,3} u_{i,2}^o$ and the received information from $j \in \mathcal{N}_i$ yields

$$s_{j,3} = B_{j,3} y_{j,3} + \sum_{k \in \mathcal{N}_j} C_{j,3}^{(k)} \left(B_{k,2} y_{k,2} + \sum_{l \in \mathcal{N}_k \setminus j} C_{k,2}^{(l)} B_{l,1} y_{l,1} \right)$$
$$= B_{j,3} y_{j,3} + \sum_{k \in \mathcal{N}_j \setminus i} C_{j,3}^{(k)} \left(B_{k,2} y_{k,2} + \sum_{l \in \mathcal{N}_k \setminus j} C_{k,2}^{(l)} B_{l,1} y_{l,1} \right)$$
$$+ C_{j,3}^{(i)} \left(s_{i,2} - C_{i,2}^{(j)} s_{j,1} \right).$$

Then, $\bar{H}_{i,3}$ is given by

$$\bar{H}_{i,3} = \begin{bmatrix} B_{j_1,3} H_{j_1} + C_{j_1,3} \bar{H}_{j_1,2} \\ \vdots \\ B_{j_{\pi_i},3} H_{j_{\pi_i}} + C_{j_{\pi_i},3} \bar{H}_{j_{\pi_i},2} \end{bmatrix} - \begin{bmatrix} C_{j_1,3}^{(i)} \bar{H}_{j_1,2}^{(i)} \\ \vdots \\ C_{j_{\pi_i},3}^{(i)} \bar{H}_{j_{\pi_i},2}^{(i)} \end{bmatrix}. \tag{24}$$

Correspondingly, we have

$$G_{i,3} = \begin{bmatrix} B_{j_1,3} \Sigma_{n_{j_1}} B_{j_1,3}' + C_{j_1,3} \bar{G}_{j_1,2} C_{j_1,3}' \\ \vdots \\ B_{j_{\pi_i},3} \Sigma_{n_{j_{\pi_i}}} B_{j_{\pi_i},3}' + C_{j_{\pi_i},3} \bar{G}_{j_{\pi_i},2} C_{j_{\pi_i},3}' \end{bmatrix}$$
$$- \begin{bmatrix} C_{j_1,3}^{(i)} G_{j_1,2}^{(i)} (C_{j_1,3}^{(i)})' \\ \vdots \\ C_{j_{\pi_i},3}^{(i)} G_{j_{\pi_i},2}^{(i)} (C_{j_{\pi_i},3}^{(i)})' \end{bmatrix}. \tag{25}$$

Therefore, the oracle action can be written as

$$u_{i,4}^o = A_{i,4} u_{i,3}^o + B_{i,4} y_{i,4} + C_{i,4}(r_{i,4} - D_{i,3} s_{i,2} + T_{i,3} r_{i,2}),$$

where $A_{i,4}, B_{i,4}, C_{i,4}, D_{i,3}$ are defined accordingly and

$$T_{i,3} := \begin{bmatrix} C_{j_1,3}^{(i)} C_{i,2}^{(j_1)} & & \\ & \ddots & \\ & & C_{j_{\pi_i},3}^{(i)} C_{i,2}^{(j_{\pi_i})} \end{bmatrix}.$$

Following identical steps, for $t \geq 1$, the OEDOL algorithm is given by

$$u_{i,t}^o = A_{i,t} u_{i,t-1}^o + B_{i,t} y_{i,t} + C_{i,t} w_{i,t}, \tag{26}$$

$$\hat{\Sigma}_{i,t} = A_{i,t} \hat{\Sigma}_{i,t-1}, \tag{27}$$

where $w_{i,t}$ is the innovation term extracted from the received information, which evolves according to

$$w_{i,t} = r_{i,t} - D_{i,t-1} s_{i,t-2} + T_{i,t-1} w_{i,t-2}. \tag{28}$$

The weighting matrices $A_{i,t}$, $B_{i,t}$, $C_{i,t}$, $D_{i,t}$, and $T_{i,t}$ are defined by

$$\begin{bmatrix} B_{i,t} & C_{i,t} \end{bmatrix} = \hat{\Sigma}_{i,t-1} \tilde{H}'_{i,t-1} \times$$
$$\left(\tilde{H}_{i,t-1} \hat{\Sigma}_{i,t-1} \tilde{H}'_{i,t-1} + \tilde{G}_{i,t-1} \right)^{-1}, \tag{29}$$

$$A_{i,t} = I - B_{i,t} H_i - C_{i,t} \bar{H}_{i,t-1}, \tag{30}$$

$$D_{i,t} = \text{col}\left\{ C_{j_1,t}^{(i)}, \dots, C_{j_{\pi_i},t}^{(i)} \right\} \tag{31}$$

$$T_{i,t} = \begin{bmatrix} C_{j_1,t}^{(i)} C_{i,t-1}^{(j_1)} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & C_{j_{\pi_i},t}^{(i)} C_{i,t-1}^{(j_{\pi_i})} \end{bmatrix}, \tag{32}$$

where $\tilde{H}_{i,t} = \text{col}\{H_i, \bar{H}_{i,t}\}$, $\tilde{G}_{i,t} = \text{diag}\{\Sigma_{n_i}, \bar{G}_{i,t}\}$ and $\bar{G}_{i,t} = \text{diag}\{G_{i,t}\}$. By (24) and (25), the intermediate parameters $\bar{H}_{i,t}$ and $G_{i,t}$ evolve according to

$$\bar{H}_{i,t} = \begin{bmatrix} B_{j_1,t} H_{j_1} + C_{j_1,t} \bar{H}_{j_1,t-1} \\ \vdots \\ B_{j_{\pi_i},t} H_{j_{\pi_i}} + C_{j_{\pi_i},t} \bar{H}_{j_{\pi_i},t-1} \end{bmatrix} - \begin{bmatrix} C_{j_1,t}^{(i)} \bar{H}_{j_1,t-1}^{(i)} \\ \vdots \\ C_{j_{\pi_i},t}^{(i)} \bar{H}_{j_{\pi_i},t-1}^{(i)} \end{bmatrix}, \tag{33}$$

$$G_{i,t} = \begin{bmatrix} B_{j_1,t} \Sigma_{n_{j_1}} B'_{j_1,t} + C_{j_1,t} \bar{G}_{j_1,t-1} C'_{j_1,t} \\ \vdots \\ B_{j_{\pi_i},t} \Sigma_{n_{j_{\pi_i}}} B'_{j_{\pi_i},t} + C_{j_{\pi_i},t} \bar{G}_{j_{\pi_i},t-1} C'_{j_{\pi_i},t} \end{bmatrix}$$
$$- \begin{bmatrix} C_{j_1,t}^{(i)} G_{j_1,t-1}^{(i)} \left(C_{j_1,t}^{(i)} \right)' \\ \vdots \\ C_{j_{\pi_i},t}^{(i)} G_{j_{\pi_i},t-1}^{(i)} \left(C_{j_{\pi_i},t}^{(i)} \right)' \end{bmatrix} \tag{34}$$

and we initialize the parameters as $\bar{H}_{j,\tau} = \mathbf{0}$ and $G_{j,\tau} = \mathbf{0}$ for $\tau < 1$. Then, agent-i sends

$$s_{i,t} = u_{i,t}^o - A_{i,t} u_{i,t-1}^o. \tag{35}$$

The detailed description of the algorithm is provided in Table 1.

3.4. Computational complexity

In (26), the combination matrices $A_{i,t}$, $B_{i,t}$, $C_{i,t}$, $D_{i,t-1}$, and $T_{i,t-1}$ are independent of the streaming data although they are time-varying. Hence they can be computed before-hand. In that case, the computational complexity of the iterations for each agent is dominated by the term $C_{i,t} w_{i,t}$. Therefore, the average computational complexity is on the order of $p^2 \pi^2$, where $\pi^2 := 1/m \sum_{i=1}^m \pi_i^2$, i.e., $O(p^2 \pi^2)$. Otherwise, the computational complexity of the algorithm is mainly dominated by the matrix inversion in (29), note that $\bar{G}_{i,t} \in \mathbb{R}^{p\pi_i \times p\pi_i}$, unless the network is sparsely connected, i.e., $\pi_i \ll m$ for $i = 1, \dots, m$. Therefore, over

Table 1
The description of the OEDOL algorithm.

Algorithm: The OEDOL Algorithm.

Initialization:
 For $i = 1$ to m do
 $\quad u_{i,0}^o = \bar{x}$, $\hat{\Sigma}_{i,0} = \Sigma_x$,
 $\quad \bar{H}_{i,\tau} = \mathbf{0}$, $G_{i,\tau} = \mathbf{0}$, and $w_{i,\tau} = \mathbf{0}$ for $\tau < 1$
 End for
 Iterations:
 Do for $t \geq 1$
 $\quad$ For $i = 1$ to m do
 $\quad\quad$ **Construction of Weights:**
 $\quad\quad$ For $j = 1$ to m do
 $\quad\quad\quad$ *Calculate $\bar{H}_{j,t}$ and $G_{j,t}$ by (33) and (34).*
 $\quad\quad\quad$ *Determine combination matrices via (29)–(32).*
 $\quad\quad$ End for
 $\quad\quad$ *Construct $r_{i,t}$ through received $s_{j,t-1}$ for $j \in \mathcal{N}_i$*
 $\quad\quad$ **Extraction of Innovation:**
 $\quad\quad$ $w_{i,t} = r_{i,t} - D_{i,t-1} s_{i,t-2} + T_{i,t-1} w_{i,t-2}$
 $\quad\quad$ **Update:**
 $\quad\quad$ $u_{i,t}^o = A_{i,t} u_{i,t-1}^o + B_{i,t} y_{i,t} + C_{i,t} w_{i,t}$
 $\quad\quad$ $\hat{\Sigma}_{i,t} = A_{i,t} \hat{\Sigma}_{i,t-1}$
 $\quad\quad$ *Disclose $s_{i,t} = u_{i,t}^o - A_{i,t} u_{i,t-1}^o$ to the neighbors.*
 $\quad$ End for

Table 2
A comparison of the computational complexities of the proposed algorithms.

Algorithm	Without weights	Pre-computed weights
ODOL	$O\left((qm)^3\right)$	$O\left((qm)^2\right)$
OEDOL	$O\left(mp^3\pi^3\right)$	$O\left(p^2\pi^2\right)$

a non-sparse network, the average complexity is on the order of $mp^3\pi^3$ (π^3 is defined accordingly) since each agent i computes $B_{j,t}$ and $C_{j,t}$ for $j = 1 \dots, m$. In particular, the complexity is given by $O\left(mp^3\pi^3\right)$, while it is $O\left((qm)^3\right)$ for the ODOL algorithm. Note that over tree networks, we have $m - 1$ edges and correspondingly average neighborhood size is small. Hence, disclosure of local estimates over tree networks also reduces the computational complexity compared to the time-stamped disclosure strategy in general (in addition to the substantial reduction in communication). In Table 2, we tabulate the computational complexities of the introduced algorithms.

We point out that diffusion or consensus based algorithms have relatively low complexity, i.e., on the order of p^2 in the least-mean-square based algorithms and on the order of p^3 in quasi-Newton based algorithms, since exchanged information is handled irrespective of the content. Such algorithms also present appealing performance for certain applications in addition to the low computational complexity. However, they do not achieve the oracle performance.

In the following, we analyze whether the agents can still achieve the oracle performance through the exchange of local estimates over the networks not in tree structure.

3.5. Tree networks involving cell structures

While constructing the spanning tree, we cancel certain communication links in order to avoid multi-path information propagation. However, we also observe that in a fully connected network agents can achieve the oracle performance through the disclosure of local observations. In particular, since all of the agents are connected, each agent can receive the observations across the network directly. Correspondingly, in a fully connected network, we can achieve identical performance with the ODOL algorithm only through the disclosure of the local estimates as stated in the following corollary formally.
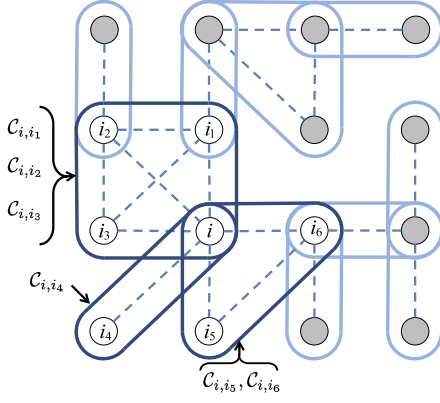
Fig. 5. An example tree network involving cell structures.

Corollary 3.1. *Consider the team-problem over a fully connected network, in which $r = p$. Then, exchange of local estimates can lead to the team-optimal solution, i.e., agents can achieve the oracle performance.*

Proof. Over a fully connected network, $\kappa_i = 1$ and the oracle action is given by

$$u_{i,t}^o = E\left[\mathbf{x}|\{\mathbf{y}_{i,\tau} = y_{i,\tau}\}^{\tau \le t}, \{\mathbf{y}_{j,\tau} = \mathbf{y}_{j,\tau}\}_{j \in \mathcal{N}_i}^{\tau \le t-1}\right] \quad (35)$$

and we can also obtain (35) by (11) since we have

$$\Delta_{j,i,t}^o = \delta_{j,t}^o \setminus \{\delta_{i,t}^o \setminus \{y_{i,t}\}\},$$

which implies that $\Delta_{j,i,t}^o$ is constructible from $\delta_{i,\tau}^o$ and $\delta_{j,\tau}^o$ for $\tau \le t$. The proof is concluded. $\square$

We point out that the team-optimal strategies can fail if a link or node failure occurs. However, once a link failure is detected, team-optimal strategies can be recomputed by eliminating the failed link in the new network configuration. Hence, through such strategies, we can increase the robustness of the team against link and node failures.

We define a "cell structure" as a sub-network in which all agents are connected to each other. Intuitively, considering a cell structure as a "single" agent, the cell (i.e., all the agents in the cell) can be involved in the tree such that the agents can still achieve the oracle performance through the disclosure of local estimates (although there may be loops in the cell). We list the features of the cell structures, e.g., seen in Fig. 5, as follows:

- Agents out of a cell can connect to at most one of the agents within that cell.
- A cell structure consists of at least 2 agents.
- An agent can belong to more than one cell.
- Two different agents cannot belong to more than one cell at the same time.
- All of the agents belong to at least a cell in a connected network.
- Each agent has also the knowledge of the cells of the other agents.
- Each agent labels its cells from its own and its first order neighbor's point of view. As an example, for agent-i, $\mathcal{C}_{i,i_1}$ denotes the cell involving both i and i_1. Note that if the same cell also includes i_2, $\mathcal{C}_{i,i_1} = \mathcal{C}_{i,i_2}$.

The following theorem shows that agents can achieve the oracle performance over tree networks involving cell structures through the disclosure of the local estimates.

Theorem 3.2. *Consider the team problem over tree networks involving cell structures. Then, exchange of local estimates can lead to the team-optimal solution, i.e., agents can achieve the oracle performance.*

Proof. Initially, we have $\Delta_{j,i,1}^o = \delta_{j,1} = \{y_{j,1}\}$ and the oracle action is also given by (11) over this network topology. Note that the information received by j at $t = 2$ is given by $\delta_{j,2}^o =$

$$\left\{y_{j,2}, \left\{\overbrace{y_{k,1}}^{=\Delta_{k,j,0}^o}\right\}_{k \in \mathcal{N}_j}, \delta_{j,1}^o\right\}, \text{ which yields}$$

$$\Delta_{j,i,2}^o = \delta_{j,2}^o \setminus \left\{\delta_{j,1}^o \cup \bigcup_{k \in \mathcal{C}_{i,j}} \{y_{k,1}\}\right\},$$

$$= \delta_{j,2}^o \setminus \left\{\delta_{j,1}^o \cup \bigcup_{k \in \mathcal{C}_{i,j}} \Delta_{k,j,1}^o\right\}$$

and $\Delta_{j,j,t}^o = \varnothing$ by definition. Due to the cell structure, we have

$$\bigcup_{k \in \mathcal{C}_{j,i}} \Delta_{k,j,1}^o = \Delta_{i,j,1}^o \cup \bigcup_{k \in \mathcal{C}_{i,j} \setminus j} \Delta_{k,i,1}^o,$$

$$= \delta_{i,1}^o \cup \bigcup_{k \in \mathcal{C}_{i,j} \setminus j} \delta_{k,1}^o.$$

Correspondingly, for $t > 0$ we have

$$\Delta_{j,i,t}^o = \delta_{j,t}^o \setminus \left\{\delta_{j,t-1}^o \cup \Delta_{i,j,t-1}^o \cup \bigcup_{k \in \mathcal{C}_{i,j} \setminus j} \Delta_{k,i,t-1}^o\right\} \quad (36)$$

and $\Delta_{j,i,t}^o$ is constructible by the sets $\delta_{i,\tau}^o$ and $\delta_{j,\tau}^o$ for $j \in \mathcal{N}_i$ and $\tau \le t$. Note that over tree networks, $\mathcal{C}_{i,j} \setminus j = \varnothing$ for $i = 1, \ldots, m$, $j \in \mathcal{N}_i$, and (36) leads to (18). Hence, for $t > 0$ we obtain (19) and the proof is concluded. $\square$

Note that the network can have loops within the cell structures and agents can still achieve the oracle performance through the diffusion of the local estimates. This increases the robustness of the team strategies against the link failures. In the sequel, we provide the sub-optimal extensions of the algorithms for practical applications.

4. Sub-optimal approaches

Minimization of the cost function (4) optimally requires relatively excessive computations. We aim to mitigate the problem sub-optimally yet in a computationally efficient approach while achieving comparable performance with the optimal case. As an example, we can approximate the cost measure (4) through $T_o \ge \max\{\kappa_i\}_{i=1,\ldots,m}$ size time-windowing as follows
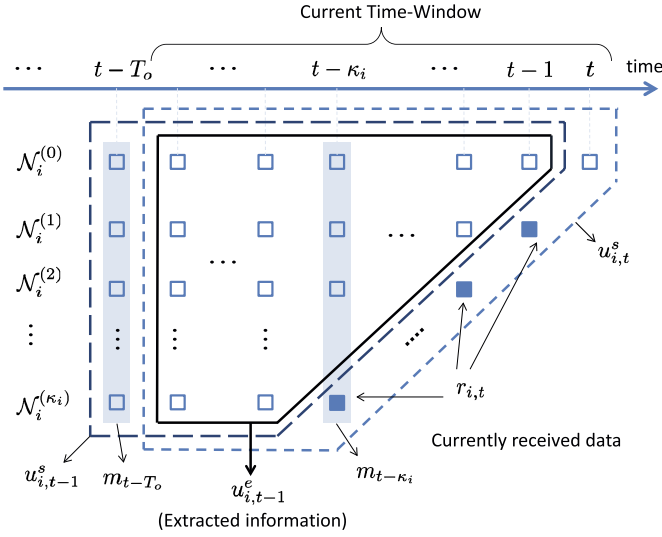
$$\sum_{t=1}^{T} \min_{\eta_{i,t} \in \Omega_{i,t}^s} E\|\mathbf{x} - \eta_{i,t}(\delta_{i,t}^s)\|^2, \quad (37)$$

where agent-i has the information set

$$\delta_{i,t}^s := \left\{\{y_{i,\tau}\}^{t-T_o < \tau \le t}, \{y_{j,\tau}\}_{j \in \mathcal{N}_i^{(1)}}^{t-T_o < \tau \le t-1}, \ldots, \right.$$

$$\left. \{y_{j,\tau}\}_{j \in \mathcal{N}_i^{(\kappa_i)}}^{t-T_o < \tau \le t-\kappa_i}\right\}, \quad (38)$$

if $t \ge T_o$ and other cases are defined accordingly, $\sigma_{i,t}^s$ is the sigma-algebra generated by $\delta_{i,t}^s$, and $\Omega_{i,t}^o$ denotes the set of all

Fig. 6. Information aggregation illustration. Small squares represent the information across the agents in the corresponding neighborhood and time.

$\sigma_{i,t}^s$-measurable functions from $\mathbb{R}^{q\kappa} \times \mathbb{R}^{q\pi_i(\kappa-1)} \times \ldots \times \mathbb{R}^{q\pi_i^{\kappa_i}(\kappa-\kappa_i)}$ to $\mathbb{R}^p$. Then, team-optimal action is given by

$$u_{i,t}^s = E[\boldsymbol{x}|\delta_{i,t}^s = \delta_{i,t}^s],$$

where $\delta_{i,t}^s$ is the set of corresponding random parameters.

In Fig. 6, we illustrate the time-windowing approach for time stamped information disclosure. We define a memory element m_t denoting the expectation of the state conditioned on all measurement at time t, i.e., $m_t := E[\boldsymbol{x}|\{\boldsymbol{y}_{i,t} = y_{i,t}\}_{i=1}^m]$. We consider that $\bar{x} = \mathbf{0}$. Then, we can write $m_t = My_t$, where $M := \Sigma_x H' \left(H\Sigma_x H' + \Sigma_n\right)^{-1}$ and $y_t := \mathrm{col}\{y_{1,t}, \ldots, y_{m,t}\}$.

Let $u_{i,t-1}^e$ represent the extracted information from the estimate $u_{i,t-1}^s$ via the memory element m_{t-T_o}, i.e.,

$$u_{i,t-1}^e := E\left[\boldsymbol{x}\middle| \{\boldsymbol{y}_{i,\tau} = y_{i,\tau}\}^{t-T_o+1<\tau\leq t-1}, \ldots, \right.$$
$$\left. \{\boldsymbol{y}_{j,\tau} = y_{j,\tau}\}_{j\in\mathcal{N}_i^{(\kappa_i)}}^{t-T_o+1<\tau\leq t-\kappa_i-1} \right]$$

if $t \geq T_o$ and other cases are defined accordingly. Then, we can write $u_{i,t-1}^e = N_i y_{i,t-1}^e$, where $N_i = \Sigma_x \hat{H}_i' \left(\hat{H}_i \Sigma_x \hat{H}_i' + \hat{\Sigma}_{n_i}\right)^{-1}$ and $y_{i,t-1}^e$ is the vector collecting the measurements within that window (see Fig. 6). Additionally, we define

$$\hat{H}_i := \mathrm{col}\left\{H_i, H_i, H_i^{(1)}, H_i, H_i^{(1)}, H_i^{(2)}, \ldots, H_i, H_i^{(1)}, \ldots, H_i^{(\kappa_i)}\right\}$$

and $H_i^{(k)} := \mathrm{col}\left\{H_j\right\}_{j\in\mathcal{N}_i^{(k)}}$. Correspondingly, we have

$$\hat{\Sigma}_{n_i} := \mathrm{diag}\left\{\Sigma_{n_i}, \Sigma_{n_i}, \Sigma_{n_i}^{(1)}, \Sigma_{n_i}, \Sigma_{n_i}^{(1)}, \Sigma_{n,i}^{(2)}, \ldots, \right.$$
$$\left. \Sigma_{n_i}, \Sigma_{n_i}^{(1)}, \ldots, \Sigma_{n_i}^{(\kappa_i)}\right\}$$

and $\Sigma_{n_i}^{(k)} := \mathrm{diag}\left\{\Sigma_{n_j}\right\}_{j\in\mathcal{N}_i^{(k)}}$. Therefore, for $t \geq T_o$, we have

$$u_{i,t-1}^s = E[\boldsymbol{x}|\boldsymbol{u}_{i,t-1}^e = u_{i,t-1}^e, \boldsymbol{m}_{t-T_o} = m_{t-T_o}].$$

Note that $E[\boldsymbol{x}|\boldsymbol{m}_{t-T_o} = m_{t-T_o}] = m_{t-T_o}$ and $\hat{\Sigma} := E[(\boldsymbol{x} - m_{t-T_o})(\boldsymbol{x} - m_{t-T_o})'] = \Sigma_x H'(H\Sigma H' + \Sigma_n)^{-1}H\Sigma = (I - MH)\Sigma$. Then, the distribution of $\boldsymbol{x}$ and $\boldsymbol{u}_{i,t-1}^e$ conditioned on $\boldsymbol{m}_{t-T_o} = m_{t-T_o}$ is given by

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u}_{i,t-1}^e \end{bmatrix} | \boldsymbol{m}_{t-T_o}$$
$$= m_{t-T_o} \sim \mathbb{N}\left(\begin{bmatrix} m_{t-T_o} \\ N_i \hat{H}_i m_{t-T_o} \end{bmatrix}, \begin{bmatrix} \hat{\Sigma} & \hat{\Sigma}\hat{H}_i' N_i' \\ N_i \hat{H}_i \hat{\Sigma} & N_i \hat{H}_i \hat{\Sigma}\hat{H}_i' N_i' + N_i \hat{\Sigma}_{n_i} N_i' \end{bmatrix}\right)$$

and we obtain

$$u_{i,t-1}^s = m_{t-T_o} + \hat{\Sigma}\hat{H}_i' N_i'(N_i\hat{H}_i\hat{\Sigma}\hat{H}_i'N_i' + N_i\hat{\Sigma}_{n_i}N_i')^{-1}$$
$$\times (u_{i,t-1}^e - N_iH_im_{t-T_o})$$
$$= K_i m_{t-T_o} + L_i u_{i,t-1}^e, \tag{39}$$

where

$$L_i := \hat{\Sigma}\hat{H}_i' N_i'(N_i\hat{H}_i\hat{\Sigma}\hat{H}_i'N_i' + N_i\hat{\Sigma}_{n_i}N_i')^{-1},$$
$$K_i := I - L_iN_i\hat{H}_i.$$

Next, we aim to compute $u_{i,t}^s$ through $u_{i,t-1}^s$ and

$$\delta_{i,t}^s = \{y_{i,t-1}^e, y_{i,t}, r_{i,t}\},$$

where $r_{i,t}$ is a vector consisting of currently received measurements as seen in Fig. 6. Let $\bar{H}_i := \mathrm{col}\{H_i, H_i^{(1)}, \ldots, H_i^{(\kappa_i)}\}$ and $\bar{\Sigma}_i := \mathrm{col}\{\Sigma_{n_i}, \Sigma_{n_i}^{(1)}, \ldots, \Sigma_{n_i}^{(\kappa_i)}\}$. Since $E[(\boldsymbol{x} - u_{i,t-1}^e)(\boldsymbol{x} - u_{i,t-1}^e)'] = \Sigma_x\hat{H}_i'(\hat{H}_i\Sigma_x\hat{H}_i' + \hat{\Sigma}_n)^{-1}\hat{H}_i\Sigma_x = (I - N_i\hat{H}_i)\Sigma_x$, we have

$$u_{i,t}^s = u_{i,t-1}^e + \Sigma_{x,i}\bar{H}_i'(\bar{H}_i\Sigma_{x,i}\bar{H}_i' + \bar{\Sigma}_{n_i})^{-1}\left(\begin{bmatrix} y_{i,t} \\ r_{i,t} \end{bmatrix} - \bar{H}_i u_{i,t-1}^e\right),$$

where $\Sigma_{x,i} := (I - N_i\bar{H}_i)\Sigma_x$. By (39), $u_{i,t-1}^e = L_i^{-1}(u_{i,t-1}^s - K_im_{t-T_o})$, and we define the combination matrices

$$\begin{bmatrix} B_i & C_i \end{bmatrix} = \Sigma_{x,i}\bar{H}_i'\left(\bar{H}_i\Sigma_{x,i}\bar{H}_i' + \bar{\Sigma}_{n_i}\right)^{-1}, \tag{40}$$
$$A_i := \left(I - \begin{bmatrix} B_i & C_i \end{bmatrix} \bar{H}_i\right)L_i^{-1}, \tag{41}$$
$$D_i := A_iK_iM. \tag{42}$$

Therefore, for $t \geq T_o$, the sub-optimal distributed online learning (SDOL) algorithm is given by

$$u_{i,t}^s = A_iu_{i,t-1}^s + B_iy_{i,t} + C_ir_{i,t} - D_iy_{t-T_o}. \tag{43}$$

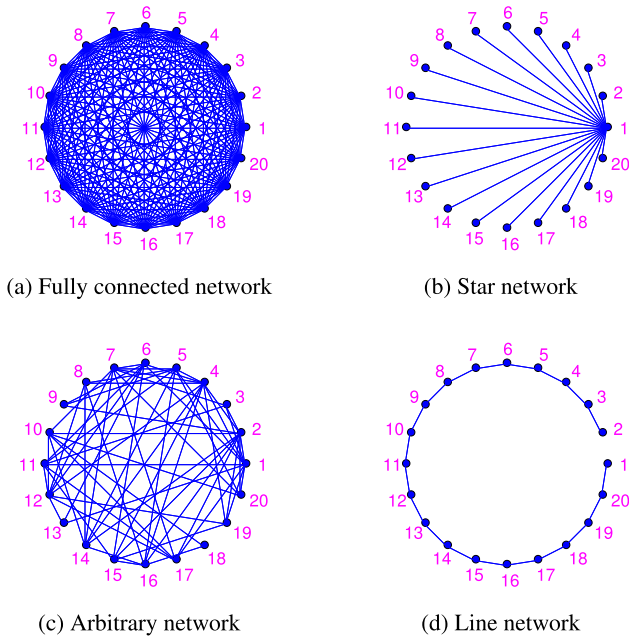In the following we provide several remarks about the practical implementation of the SDOL algorithm.

Remark 4.1.

- We point out that in the update (43) the matrices A_i, B_i, C_i, and D_i are independent from the data and they are time invariant for $t \geq T_o$. Hence, they can be pre-computed and installed onto the agents. Then, the SDOL algorithm basically takes the linear average of the previous estimate, the current measurement and received data, and measurements at time $t - T_o$.

- Different from the conventional approaches, the SDOL algorithm requires to memorize previous observations. Note that if $q < p/m$, storing measurements individually rather than a linear combination, i.e., m_t, might be more efficient in terms of memory usage.

- The computational complexity of the SDOL algorithm is $O(p^2 + 2pmq)$ (or say $O(p^2)$ for $q \ll p/m$) due to matrix–vector multiplications in (43), and the algorithm requires disclosure of a data vector with $(m-1) \times q$ dimensions.

- Finally, even though agent-i uses update (43) for $t < T_o$ assuming that[3] $y_{j,\tau} = 0$ for $j = 1, \ldots, m$ and $\tau < 1$, we have

---

[3] The action is no longer team-optimal because of the assumption.

Table 3
The description of the SDOL-T_o algorithm.

Algorithm: The SDOL-T_o algorithm
Initialization:
$u_{i,0}^s = \mathbf{0}$ for all i
$y_{i,\tau} = \mathbf{0}$ for $\tau < 1$ and all i
Calculate combination matrices via (40)–(42).
Update:
Do for $t \geq 1$
For $i = 1$ to m do
Construct $r_{i,t}$ through received $s_{j,t-1}$ for $j \in \mathcal{N}_i$.
Store $r_{i,t}$ and $y_{i,t}$ in the memory accordingly.
$u_{i,t}^s = A_i u_{i,t-1}^s + B_i y_{i,t} + C_i r_{i,t} - D_i y_{t-T_o}$.
Erase y_{t-T_o} from the memory.
Diffuse $s_{i,t}$ to the neighboring agents.
End for

(a) Fully connected network (b) Star network

(c) Arbitrary network (d) Line network

Fig. 7. Different network configurations.

$$E[(x - u_{i,\tau}^s)(x - u_{i,\tau}^s)']$$
$$= \Sigma_{x,i} - \Sigma_{x,i} \bar{H}_i' (\bar{H}_i \Sigma_{x,i} \bar{H}_i' + \bar{\Sigma}_{n_i})^{-1} \bar{H}_i \Sigma_{x,i}$$
$$= A_i L_i \Sigma_{x,i} \qquad (44)$$

for $\tau \geq T_o$ since at each iteration, agent-i excludes the impact of y_{t-T_o} on $u_{i,t}^s$. This sharp change results in as a breaking point on the learning curve as seen in Fig. 15.

In Table 3, we tabulate a detailed description of the SDOL algorithm. Note that SDOL algorithm is based on the aggregation of information. Correspondingly, we can apply the time-windowing approach to the disclosure of local estimates and formulate a sub-optimal efficient distributed online learning algorithm (SEDOL) over tree networks.

In the sequel, we also provide illustrative simulations showing the enhanced tracking performance due to the proposed algorithm over several distributed network scenarios.

5. Illustrative examples

In this section, we examine the performance of team optimal actions under different scenarios through numerical examples. To this end, we consider three main network structures, namely fully
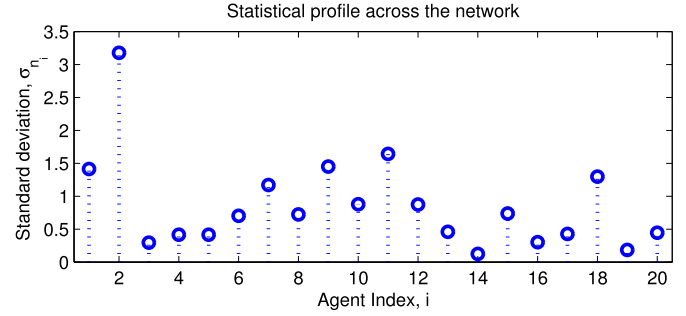
Fig. 8. Standard deviation of measurement noise across the network.

connected, star, and line networks with $m = 20$ agents, and a randomly generated network seen in Fig. 7. In a fully connected network, each agent has a connection with each other and any new measurement can be collected at each other after just one hop. This is the best possible connection such that the corresponding achievable team cost is the smallest. In a star network, there exists a pseudo-centralized agent, e.g., agent-1 in Fig. 7b, and all the other agents are connected to that agent. Therefore, the furthest distance between any two agents is 2, e.g., there exist 2-hops between agent-2 and agent-3 (agent-2 to agent-1 then agent-1 to agent-3) in Fig. 7b. We have generated the network in Fig. 7c randomly such that number of neighbors of agent-i, π_i, is chosen uniformly between 2 to 10. Therefore, the generated network can have loops, i.e., may not be a tree. Furthermore, in a line network, the connections between the agents form a line as seen in Fig. 7d. Contrary to the fully connected network, a line network has the least possible connection such that any two agents are connected via a certain number of hops. Note that there exist $m - 1 = 19$ hops between agent-1 and agent-2 in Fig. 7d.

We analyze the performance of a team of agents over those network structures. We set $p = 10$, $q = 1$, $\bar{x} = \mathbf{0}$, $\Sigma_x = I$, and the standard deviation of the zero-mean measurement noise is chosen randomly from a folded normal distribution as seen in Fig. 8. Furthermore, we choose H_i randomly from normal distribution, $\mathbb{N}(\mathbf{0}, I)$. In order to observe the change of oracle cost ($r \to \infty$) for different time horizons, we introduce

$$J(T) = \min_{\substack{\eta_{i,t} \in \Omega_{i,t}, \\ i=1,\dots,m, t=1,\dots,T}} \sum_{t=1}^{T} \sum_{i=1}^{m} E\|\boldsymbol{x} - \eta_{i,t}(\delta_{i,t}^o)\|^2, \qquad (45)$$

which is a function of the length of time horizon, T. We re-emphasize that in (45), each action at $t = 1, \dots, T$ has a cumulative impact on the cost. Let $\eta_{i,t}^o$, for $i = 1, \dots, m$ and $t = 1, \dots, T$, be the oracle strategies for $J(T)$. Then, we introduce a terminal cost function defined by

$$P(T) = \sum_{i=1}^{m} E\|\boldsymbol{x} - \eta_{i,T}^o(\delta_{i,T}^o)\|^2, \qquad (46)$$

which is the impact of final action on $J(T)$.

In Fig. 9, we compare $J(T)$, for $T = 1, \dots, 20$, over the fully connected, star, arbitrary, and line networks in Fig. 7 by ensemble averaging empirical cost of the ODOL algorithm over 100 independent trials. Note that if the associated network is a tree network, e.g., the star network is a tree, the same cost could be attained through the OEDOL algorithm as shown analytically in Section 3. We point out that Fig. 9 is not a time evolution of a learning algorithm. In particular, for each time horizon T, we compare the corresponding oracle cost over certain networks. As pointed out before, because of the most possible connection between the agents, the fully connected network results in the least oracle cost for
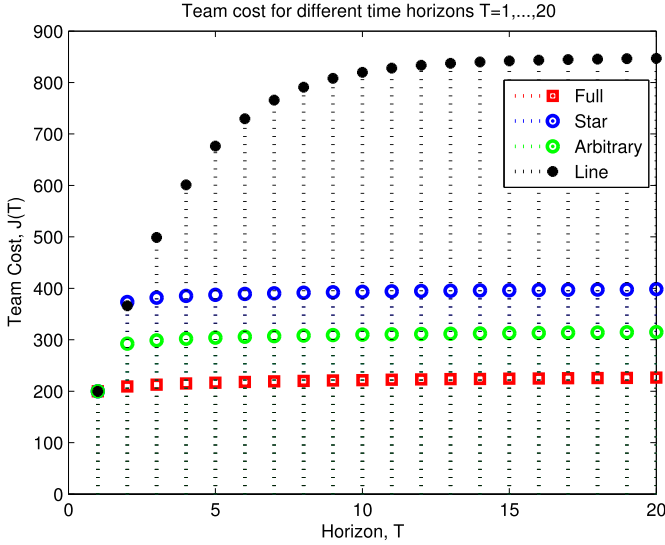
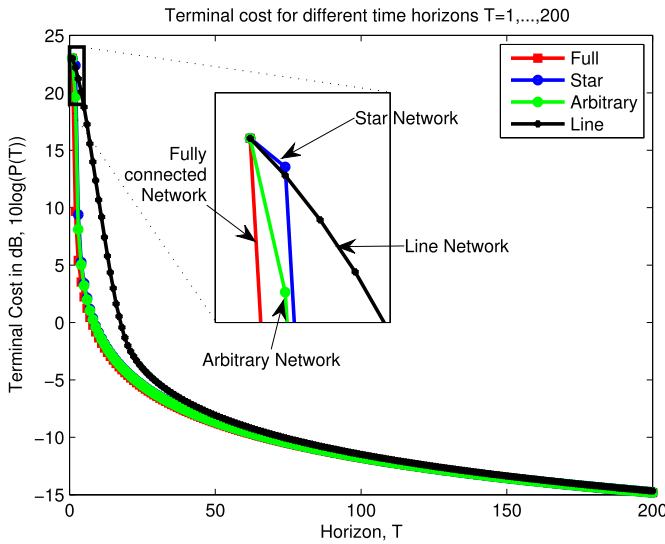Fig. 9. Comparison of oracle cost for different time horizons over certain networks.

Fig. 10. Comparison of terminal cost for different time horizons over certain networks.

(a) Arbitrary network   (b) Corresponding spanning tree

Fig. 11. An arbitrary network and a corresponding spanning tree.
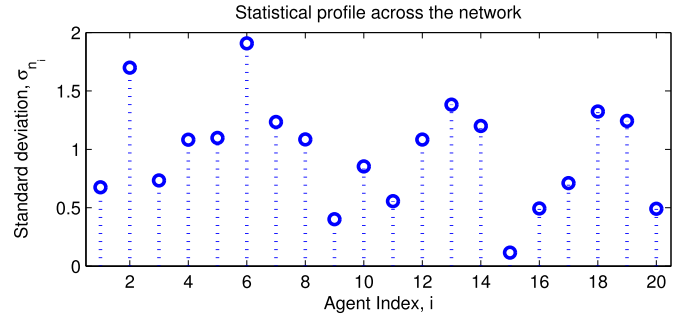
Fig. 12. Standard deviation of noise across the network in Fig. 11.

each time horizon T while the line network possesses the largest cost. Additionally, because of the cumulation of the impact of the actions at each time, for larger time horizon the corresponding oracle cost is also larger, i.e., $J(T)$ is a non-decreasing function of T (indeed increasing function since measurement noise is a white process). The arbitrary network is relatively more connected. In particular, in the arbitrary network, neighborhood size of agents is chosen uniformly between 2 and 10 while in the star network, neighborhood size of agents except agent-1 is 2. We observe that oracle cost over the arbitrary network is smaller than the oracle cost over the star network. For $T = 1$, all network configurations lead to the same oracle cost since agents can exchange information beginning at time $t = 2$. Furthermore, in Fig. 9, for time horizon $T = 2$, the oracle cost in the star network is larger than the cost in the line network. This is because even though in the star network any two agents are connected through at most 2 hops, over a horizon $T = 2$, the information can only propagate over 1 hop and in the line network all the agents except agents 1 and 2 have 2 neighbors each. For horizon $T > 2$, the information can now propagate over more than 1 hop and correspondingly, the cost in the star network is less than the cost in the line network.
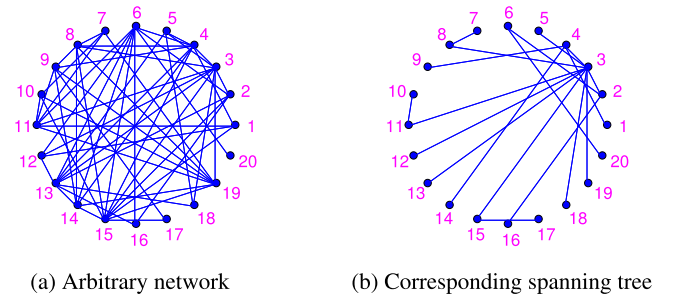
In addition to the comparison of oracle cost, $J(T)$, for different time horizons over certain networks, in Fig. 10, we compare the terminal cost, $P(T)$, in order to observe the cumulative impact of the actions in the beginning stages of the horizon, e.g., $t \ll T$. We again ensemble averaged empirical cost of the ODOL algorithm over 100 independent trials. Even though a line and a fully connected network are two extreme connections for a network, in Fig. 10, the terminal costs, $P(T)$, are asymptotically, as $T \rightarrow \infty$, the same. We recall that this is pointed out in Remark 3.1 and attributed to the negligible difference between the available information at each agent over fully connected and line networks as the information sets grow. We also point out that in the zoomed-out plot, for $T = 2$, the terminal cost in the star network is larger than the terminal cost in the line network while the terminal cost in star network becomes substantially smaller for larger T as explained above for Fig. 9.

Importantly, in Figs. 9 and 10, the oracle and terminal costs lead to counter-intuitive results as time horizon grows. In particular, while the oracle costs over different networks are separated as time horizon grows, the terminal costs converge to each other. The cumulation of the costs of each action plays a significant role in these distinct results. By yielding relatively large cost, the actions in the early stages have crucial impact on the team cost. Therefore, the sequential algorithms aiming to learn the underlying state in time through a trial and error based approach lead to larger team cost over horizon since trials at early stages lead to significantly large costs. In the following, we analyze such circumstances.

We consider a randomly generated network in Fig. 11a, constructed as the network in Fig. 7c. This network is not a tree and has loops. We construct a spanning tree of this network, see Fig. 11b, based on the paths from the agent having the largest neighborhood size, i.e., $i = \arg\max_j \pi_j$, and then eliminating the multi-paths [21]. The statistical profile of the measurement noise is plotted in Fig. 12. Here, we consider the diffusion recursive least squares (D-RLS) algorithm [29] with Laplacian combination rule [30] for the incremental update and relative-variance combination rule for the spatial update. In D-RLS, agents use a recursive least squares (RLS) update locally, diffuse local estimate to the neighbors at each time instant, and combine received estimates and the local
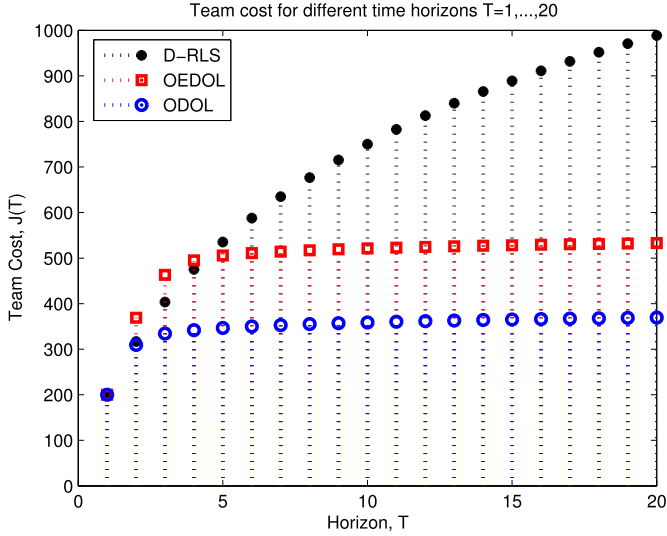
Fig. 13. Comparison of team cost of the algorithms for different time horizon.
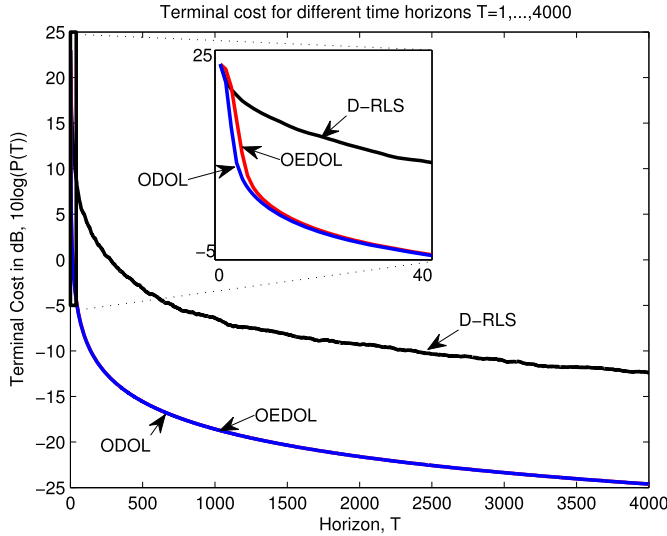

Fig. 14. Comparison of terminal cost of the algorithms for different time horizon.

Fig. 15. Comparison of terminal cost of the SDOL-50 and SDOL-100 algorithms.

one linearly through certain combination weights, e.g., agent-i has the weight $\lambda_{i,j}$ for the information received from the neighbor-j:

$$\lambda_{i,j} = \begin{cases} \dfrac{\sigma_{n_j}^{-2}}{\sum_{k \in \mathcal{N}_i \cup \{i\}} \sigma_{n_k}^{-2}} & \text{if } j \in \mathcal{N}_i \cup \{i\} \\ 0 & \text{otherwise.} \end{cases} \qquad (47)$$

In Fig. 13, we compare the ensemble average of the empirical cost (over 100 independent trials) of the ODOL and D-RLS algorithms over Fig. 11a and the OEDOL algorithm over Fig. 11b for different time horizon. Note that in the OEDOL algorithm, agents exchange local estimates while in the ODOL algorithm, agents exchange information through time-stamped approach. The OEDOL algorithm can achieve the oracle performance over a tree network. Therefore, we construct a spanning tree of the arbitrarily generated network and the OEDOL algorithm achieves the oracle performance of the spanning tree. Since we eliminate certain links while constructing the spanning tree, the oracle performance, i.e., team-cost, is worse than the oracle performance over the original network. However, even though the OEDOL algorithm operates over the spanning tree, the algorithm has performed better than the D-RLS algorithm operating over the original network. Addition-
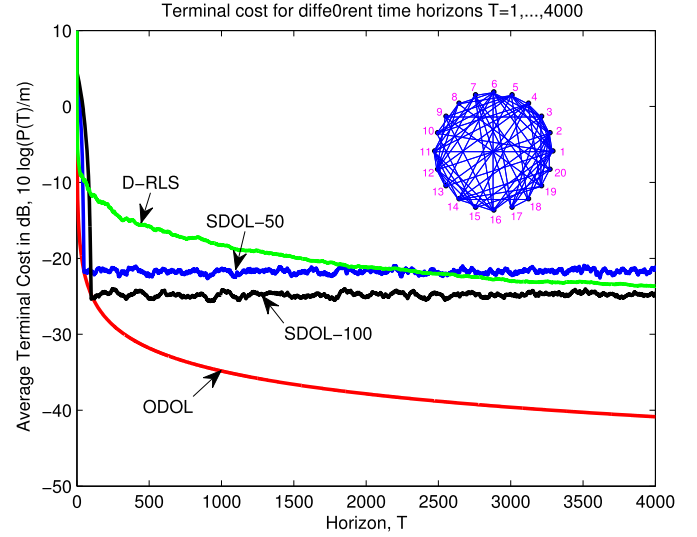
ally, in Fig. 14, we compare the terminal cost of the algorithms for larger time horizon, e.g., $T = 1, \ldots, 4000$. We observe that even though the OEDOL algorithm operates over the spanning tree, the terminal costs of the ODOL and OEDOL algorithms are close to each other for large time horizon, e.g., $T > 40$. We also note that as $T \to \infty$, the terminal cost decreases since noise processes are white.

Finally, in Fig. 15 we examine the impact of the time-windowing depth, e.g., $T_0 = 50, 100$, on the terminal cost for different time horizon over an arbitrarily generated network of 20 agents seen on the upper-right of Fig. 15. We note out that for $T < T_0$, the terminal cost of the SDOL algorithms differs from the terminal cost of the ODOL algorithm, i.e., the oracle performance since in the SDOL algorithms, agents uses the update (43) for $t < T_0$ assuming that all measurements $y_{j,\tau} = 0$ for $j = 1, \ldots, m$ and $\tau < 1$. As also pointed out in Remark 4.1, by excluding the impact of these measurements on the future actions through sliding time-window, the terminal cost of the SDOL-T_0 algorithm eventually reaches the terminal cost of the ODOL algorithm at $T = T_0$ and for larger time window, this performance is maintained. Furthermore, depending on T_0, the agents can achieve smaller terminal cost.

6. Conclusion

Distributed algorithms have attracted significant attention due to their wide spread applicability to highly complex structures from biological systems to social and economical networks. However, there are still challenges for disclosure and utilization of information among agents. We have considered this problem as a team problem, in which each agent takes actions, e.g., which information to disclose and how to construct the local estimate. We introduced the ODOL algorithm achieving the oracle cost for Gaussian state and noise statistics through a time-stamped approach. Importantly, we have shown that the exchange of the local estimates is sufficient to achieve the oracle cost only over certain network topologies, e.g., over the introduced tree networks involving cell structures. Furthermore, we have introduced the OEDOL algorithm, which achieves the oracle cost through the exchange of local estimates over tree networks. Finally, we have introduced a time-windowing approach for practical applications due to reduced complexity. The sub-optimal approaches possess recursive updates with time-invariant combination weights that should be calculated only once.

Some future directions of research on this topic include team problems for distributed MMSE estimation of a dynamic state and formulation of optimal information exchange in team decision problems for linear continuous-time systems [31].

Acknowledgments

References

[1] M.K. Banavar, J.J. Zhang, B. Chakraborty, H. Kwon, Y. Li, H. Jiang, A. Spanias, C. Tepedelenlioglu, C. Chakrabarti, A. Papandreou-Suppappola, An overview of recent advances on distributed and agile sensing algorithms and implementation, Digit. Signal Process. 39 (2015) 1–14.

[2] A.H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, Z.J. Towfic, Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior, IEEE Signal Process. Mag. 30 (3) (2013) 155–171.

[3] S. Kar, J.M.F. Moura, Consensus+innovations distributed inference over networks: cooperation and sensing in networked systems, IEEE Signal Process. Mag. 30 (3) (2013) 99–109.

[4] J. Fernandez-Bes, L.A. Azpicueta-Ruiz, J. Arenas-Garcia, M.T.M. Silva, Distributed estimation in diffusion networks using affine least-squares combiners, Digital Signal Processing 36 (2015) 1–14.

[5] V. Krishnamurthy, Decentralized activation in sensor networks-global games and adaptive filtering games, Digital Signal Processing 21 (5) (2011) 638–647.

[6] A. Jadbabaie, P. Molavi, A. Sandroni, A. Tahbaz-Salehi, Non-Bayesian social learning, Games Econ. Behav. 76 (1) (2012) 210–225.

[7] D. Acemoglu, A. Ozdaglar, Opinion dynamics and learning in social networks, Dyn. Games Appl. 1 (2011) 3–49.

[8] M. Jackson, Social and Economic Networks, Princeton University Press, Princeton, N.J., 2008.

[9] C.G. Lopes, A.H. Sayed, Diffusion least-mean squares over adaptive networks: formulation and performance analysis, IEEE Trans. Signal Process. 56 (7) (2008) 3122–3136.

[10] F.S. Cattivelli, A.H. Sayed, Diffusion strategies for distributed Kalman filtering and smoothing, IEEE Trans. Autom. Control 55 (9) (2010) 2069–2084.

[11] S. Shahrampour, A. Rakhlin, A. Jadbabaie, Online learning of dynamic parameters in social networks, in: Neural Information Processing Systems, 2013, pp. 2013–2021.

[12] G. Mateos, I.D. Schizas, G.B. Giannakis, Distributed recursive least-squares for consensus-based in-network adaptive estimation, IEEE Trans. Signal Process. 57 (11) (2009) 4583–4588.

[13] M.O. Sayin, S.S. Kozat, Single bit and reduced dimension diffusion strategies over distributed networks, IEEE Signal Process. Lett. 20 (10) (2013) 976–979.

[14] M.O. Sayin, S.S. Kozat, Compressive diffusion strategies over distributed networks for reduced communication load, IEEE Trans. Signal Process. 62 (20) (2014) 5308–5323.

[15] V.D. Blondel, J.M. Hendrickx, A. Olshevsky, J.N. Tsitsiklis, Convergence in multi-agent coordination, consensus, and flocking, in: Proceedings of 44th IEEE Conference on Decision and Control, CDC-ECC, 2005, pp. 2996–3000.

[16] D.S. Scherber, H.C. Papadopoulos, Locally constructed algorithms for distributed computations in ad-hoc networks, in: Proceedings of Information Processing Sensor Networks, IPSN, 2004, pp. 11–19.

[17] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (6) (1953) 1087–1092.

[18] A.H. Sayed, Fundamentals of Adaptive Filtering, Wiley, New York, 2003.

[19] B.D.O. Anderson, J.B. Moore, Optimal Filtering, Prentice–Hall, Inc., Englewood Cliffs, NJ, 1979.

[20] S. Skiena, Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Addison–Wesley, Reading, MA, 1990.

[21] B.Y. Wu, K.-M. Chao, Spanning Trees and Optimization Problems, Discrete Mathematics and Its Applications, CRC Press, New York, 2004.

[22] R.G. Gallager, P.A. Humblet, P.M. Spira, A distributed algorithm for minimum-weight spanning trees, ACM Trans. Program. Lang. Syst. 5 (1) (1983) 66–71.

[23] D. Peleg, Distributed Computing: A Locality-Sensitive Approach, SIAM, Philadelphia, PA, 2000.

[24] M. Elkin, Unconditional lower bounds on the time-approximation tradeoffs for the distributed minimum spanning tree problem, in: Proceedings of 36th ACM Symposium on Theory of Computing, STOC, 2004, pp. 331–340.

[25] M. Khan, G. Pandurangan, V.S.A. Kumar, Distributed algorithms for constructing approximate minimum spanning trees in wireless sensor networks, IEEE Trans. Parallel Distrib. Syst. 20 (1) (2009) 124–139.

[26] J. Chen, A.H. Sayed, Optimal combination rules for adaptation and learning over networks, in: Proceedings of 50th Annual Allerton Conference on Communication, Control, and Computing, Allerton, 2012, pp. 1535–1542.

[27] J. Chen, A.H. Sayed, On the learning behavior of adaptive networks – part I: transient analysis, IEEE Trans. Inf. Theory 61 (6) (2015) 3487–3517.

[28] J. Chen, A.H. Sayed, On the learning behavior of adaptive networks – part II: performance analysis, IEEE Trans. Inf. Theory 61 (6) (2015) 3518–3548.

[29] F.S. Cattivelli, C.G. Lopes, A.H. Sayed, Diffusion recursive least-squares for distributed estimation over adaptive networks, IEEE Trans. Signal Process. 56 (5) (2008) 1865–1877.

[30] R. Olfati-Saber, R.M. Murray, Consensus problems in networks of agents with switching topology and time-delays, IEEE Trans. Autom. Control 49 (9) (2004) 1520–1533.

[31] A. Bagchi, T. Başar, Team decision theory for linear continuous-time systems, IEEE Trans. Autom. Control AC-25 (6) (1980) 1154–1161.

Muhammed O. Sayin received the B.S. and M.S. degrees in electrical and electronics engineering from Bilkent University, Ankara, Turkey, in 2013 and 2015, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC). His current research interests include signaling games, dynamic games and decision theory, strategic decision making, and stochastic optimization.

Suleyman S. Kozat received the B.S. degree with full scholarship and high honors from Bilkent University, Turkey. He received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana Champaign, Urbana, IL. After graduation, Dr. Kozat joined IBM Research, T.J. Watson Research Lab, Yorktown, New York, as a Research Staff Member (and later became a Project Leader) in the Pervasive Speech Technologies Group, where he focused on problems related to statistical signal processing and machine learning. While doing his Ph.D., he was also working as a Research Associate at Microsoft Research, Redmond, Washington, in the Cryptography and Anti-Piracy Group. He holds several patent inventions due to his research accomplishments at IBM Research and Microsoft Research. Dr. Kozat is currently an Associate Professor at the Electrical and Electronics Engineering Department at Bilkent University. Dr. Kozat is the elected President of the IEEE Signal Processing Society, Turkey Chapter. Dr. Kozat coauthored more than 100 papers in refereed high impact journals and conference proceedings and has several patent inventions (currently used in several different Microsoft and IBM products such as the MSN and the ViaVoice). He holds many international and national awards. Overall, his research interests include cyber security, anomaly detection, big data, data intelligence, adaptive filtering and machine learning algorithms for signal processing.

Tamer Başar is with the University of Illinois at Urbana-Champaign, where he holds the academic positions of Swanlund Endowed Chair; Center for Advanced Study Professor of Electrical and Computer Engineering; Research Professor at the Coordinated Science Laboratory; and Research Professor at the Information Trust Institute. He is also the Director of the Center for Advanced Study. He received B.S.E.E. from Robert College, Istanbul, and M.S., M.Phil., and Ph.D. from Yale University. He is a member of the US National Academy of Engineering, member of the European Academy of Sciences, and Fellow of IEEE, IFAC (International Federation of Automatic Control) and SIAM (Society for Industrial and Applied Mathematics), and has served as president of IEEE CSS (Control Systems Society), ISDG (International Society of Dynamic Games), and AACC (American Automatic Control Council). He has received several awards and recognitions over the years, including the highest awards of IEEE CSS, IFAC, AACC, and ISDG, the IEEE Control Systems Award, and a number of international honorary doctorates and professorships. He has over 750 publications in systems, control, communications, and dynamic games, including books on non-cooperative dynamic game theory, robust control, network security, wireless and communication networks, and stochastic networked control. He was the Editor-in-Chief of Automatica between 2004 and 2014, and is currently editor of several book series. His current research interests include stochastic teams, games, and networks; distributed algorithms; security; and cyber-physical systems.