SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2017, Vol. 60 032204:1–032204:15
doi: 10.1007/s11432-016-0284-2

Distributed incremental bias-compensated RLS estimation over multi-agent networks

Jian LOU, Lijuan JIA*, Ran TAO & Yue WANG

School of Information and Electronics, Beijing Institute of Technology, Beijing 100081*, China*

Received July 11, 2016; accepted August 25, 2016; published online February 7, 2017

Abstract In this paper, we study the problem of distributed bias-compensated recursive least-squares (BC-RLS) estimation over multi-agent networks, where the agents collaborate to estimate a common parameter of interest. We consider the situation where both input and output of each agent are corrupted by unknown additive noise. Under this condition, traditional recursive least-squares (RLS) estimator is biased, and the bias is induced by the input noise variance. When the input noise variance is available, the effect of the noise-induced bias can be removed at the expense of an increase in estimation variance. Fortunately, it has been illustrated that distributed collaboration between agents can effectively reduce the variance and can improve the stability of the estimator. Therefore, a distributed incremental BC-RLS algorithm and its simplified version are proposed in this paper. The proposed algorithms can collaboratively obtain the estimates of the unknown input noise variance and remove the effect of the noise-induced bias. Then consistent estimation of the unknown parameter can be achieved in an incremental fashion. Simulation results show that the incremental BC-RLS solutions outperform existing solutions in some enlightening ways.

Keywords distributed parameter estimation, bias compensation, incremental schemes, recursive least-squares, multi-agent networks

Citation Lou J, Jia L J, Tao R, et al. Distributed incremental bias-compensated RLS estimation over multi-agent networks. Sci China Inf Sci, 2017, 60(3): 032204, doi: 10.1007/s11432-016-0284-2

1 Introduction

A distributed multi-agent network is a collection of agents with adaptation and learning abilities. The agents are connected through a topology and share information with their neighbors in a certain manner, which can be utilized to collaboratively solve inference and optimization problems [1]. Compared with traditional centralized solutions, distributed solutions do not require a powerful fusion center to process the data from every agent. As a result, distributed solutions can effectively reduce both computations and communications. On the other hand, in a centralized solution, if the fusion center breaks down, it will lead to a failure of the whole network. In comparison, the solutions over distributed multi-agent networks can avoid this problem and can be more robust to agent and link failure [2]. Therefore, the study of distributed multi-agent networks has been a popular topic in recent years and has been applied to a wide range of fields such as sensor networks, precision agriculture, environmental monitoring, etc. [3–5].

* Corresponding author (email: jlj@bit.edu.cn)

Three mainstream distributed cooperation strategies are incremental strategies [6–14], consensus strategies [15], and diffusion strategies [16–21]. When adopting incremental strategies, an annular path should be determined firstly. Then the agents transmit information in sequential order. Each agent communicates with only one of its neighbors, and thus the data are processed in a cyclic manner [6]. Compared with other distributed strategies, incremental strategies are less robust to agent and link failures, since any disconnection between agents can lead to a failure over the entire network [1]. Furthermore, the determination of an annular path is an NP-hard problem. However, incremental strategies do have many appealing advantages. For instance, the incremental solution can significantly reduce communications and computations, and the amount of communication per iteration per agent will not increase with the size of the network [2]. At the same time, incremental solutions can obtain better estimates, since the output of an incremental network comes from the last agent in the cycle, where the data from the whole network are used to update the local parameter [10]. Therefore, when the network only contains a small number of agents, or when the deployment of the agents can be controlled, the cyclic techniques can be relevant options [12]. In consensus strategies, each agent can exchange information with all of its neighbors. Thus, consensus networks are more robust to agent and link failures, and the NP-hard problem can be avoided. However, the asymmetry in the consensus update can cause an unstable growth in the state of the network [15]. Besides, the consensus mode is not suitable for real-time signal processing in time-varying environments [8]. In diffusion solutions, agents can also communicate with all of their neighbors. The data can be diffused to each agent more thoroughly. Compared with consensus networks, diffusion networks are more stable and more robust. At the same time, the amount of communication is higher than in an incremental solution. Nevertheless, the communications can be reduced by allowing agents to communicate only with a subset of their neighbors based on certain criterion [6]. In this paper, we focus on incremental strategies.

A series of distributed algorithms has been proposed to solve the problem of distributed estimation over multi-agent networks, where agents collaborate to estimate a common parameter of interest. In [15, 22], diffusion-based least-mean-squares (LMS) algorithms are proposed, and it is shown that the diffusion LMS algorithms outperform the non-cooperative LMS algorithms, which illustrates the benefits of distributed cooperation. In [17], a diffusion RLS algorithm is proposed, which can obtain good performance compared to the global solution. In [2,6,16], Sayed et al. propose incremental LMS algorithms and incremental RLS algorithms. They analyze and compare the performance of centralized LMS processing and incremental LMS processing. Simulation results show that incremental solutions can outperform centralized solutions in some ways. In [14], Khalili et al. propose an incremental LMS adaptive network that exploits knowledge of observation quality (in terms of observation noise variance), which has faster convergence rate. In [10], Khalili et al. propose an incremental LMS algorithm that can be applied to the processing of complex signals. Both simulated and real-world wind data are used in the performance evaluations, which illustrate that the incremental-based solution outperforms the non-cooperation solution in terms of convergence rate and estimation accuracy. Nevertheless, all of the above algorithms are proposed with perfect information exchange. In practice, both input and output can be corrupted by additive noise from quantization error, sampling error, channel noise, etc. In [23, 24], considering the situation where both regression data and measurements are corrupted by additive noise, diffusion bias-compensated RLS algorithms are proposed. It is shown that the noise on both regressors and measurements leads to biased estimates, and the bias can be removed at the expense of an increase in local estimation variance. At the same time, it is demonstrated that the variance can be effectively reduced by diffusion. In [7], Khalili et al. study the effect of noisy links on the steady-state performance of incremental RLS solutions, where expressions are derived to describe incremental RLS estimation with noisy information. In [25], Sayed et al. study the mean-square performance of diffusion LMS algorithms in the presence of various sources of imperfect information exchanges, quantization errors, and model non-stationarities. It is concluded that the noise over regression data plays a more important role than other sources of imperfection in deteriorating the performance of the network. However, all of the methods in [7, 23–25] assume that the noise variance is known a priori, whereas the noise variance is unknown in many practical cases [26]. Therefore, it is necessary to develop a method that can obtain the unknown noise variance firstly and then remove the effect of the noise-induced bias in a distributed fashion.

In this paper, considering the situation where both input and output of each agent in the network are corrupted by unknown additive noise, an incremental-based bias-compensated RLS (Inc BC-RLS) algorithm is proposed. A simplified version of the Inc BC-RLS (S-Inc BC-RLS) algorithm is also studied, which can further reduce communications. It is shown that the bias is induced by the input noise. When the input noise variance is unknown a priori, a distributed approach is developed to achieve the real-time estimation of the unknown noise variance. Then the effect of the noise-induced bias is removed in an incremental fashion. At the same time, in computer simulations, we compare the performance of proposed algorithms with existing algorithms (non-cooperative BC-RLS, centralized BC-RLS, and diffusion BC-RLS algorithms). Simulation results show that the proposed algorithms can obtain the consistent estimation of the unknown parameter, and that the proposed algorithms outperform existing algorithms in some revealing ways.

The contribution of this paper is summarized as follows. Firstly, we derive the incremental-based BC-RLS algorithms with more relaxed conditions that both input and output of each agent can be corrupted by additive noise, and the additive noise can be unknown a priori. Secondly, we work out a distributed method that can estimate the unknown noise variance over the network. The method used to estimate the input noise variance introduces more variables. Therefore, we study how agents exploit the information involved in the procedure of noise estimation and how agents transmit these variables to their neighbors. It shows that the transmission of the variables can significantly influence the performance of the network, but how to achieve the transmission is not trivial. We discuss the principles of transmitting variables when implementing the incremental-based BC-RLS algorithms. Thirdly, we compare the proposed algorithms to existing algorithms. Computer simulations show some valuable results.

This paper is organized as follows. In Section 2, a problem statement is made to analyze the noise-induced bias of multi-agent networks when both input and output of each agent are corrupted by additive noise. In Section 3, incremental-based BC-RLS algorithms are proposed, and a real-time noise estimation method is studied. Simulation results are given in Section 4, and conclusion is made in Section 5.

Notation. In this paper, we use small boldface letters and capital boldface letters to denote vectors and matrices respectively. The normal font is used for scalars. $\mathbb{N}$ denotes the field of natural numbers, and $\mathbb{R}$ denotes the field of real numbers. We also use $\widehat{\cdot}$ to denote an estimator. $\bar{\cdot}$ is used to denoted noise-free measurements, and $\cdot^o$ is used to denote the true value of parameter. The superscript T denotes the transpose operator, and $\mathrm{E}(\cdot)$ denotes the expectation operator. Besides, $||\cdot||$ is used as the 2-norm operator.

2 Problem statement

A multi-agent network topology is shown in Figure 1, where M agents are included. Its task is to estimate a common parameter of interest $\boldsymbol{w}^o \in \mathbb{R}^{L\times 1}$, where L is the length of an FIR model. For each agent k in the network, a noisy FIR system is modeled as shown in Figure 2.

At time $i \in \mathbb{N}$, each agent $k \in \{1, 2, \ldots, M\}$ has access to noisy input $u_k(i)$ and noisy output $d_k(i)$, which can be expressed as

$$u_k(i) = \bar{u}_k(i) + n_k(i), \tag{1}$$

$$d_k(i) = \bar{d}_k(i) + v_k(i), \tag{2}$$

where $\bar{u}_k(i)$ and $\bar{d}_k(i)$ are noise-free input and output, $n_k(i)$ and $v_k(i)$ are input noise and output noise respectively.

The noisy FIR system is constructed based on following assumptions:

- The length L of the FIR model is known.
- $\bar{u}_k(i)$ is independent and identically distributed (i.i.d) over time and independent over space.
- $n_k(i)$ is independent and identically distributed (i.i.d) over time and independent over space, with zero-mean and variances $\sigma^2_{n_k}$.

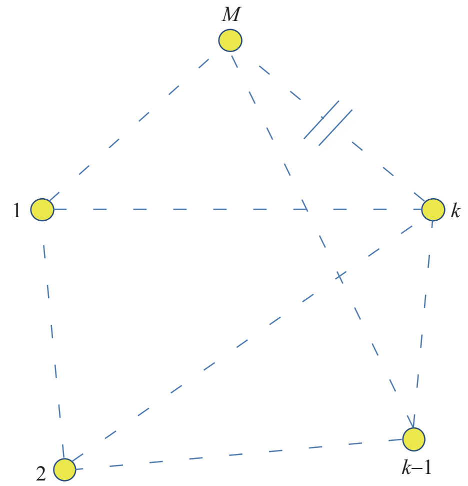

Figure 1 (Color online) A multi-agent network with M agents.

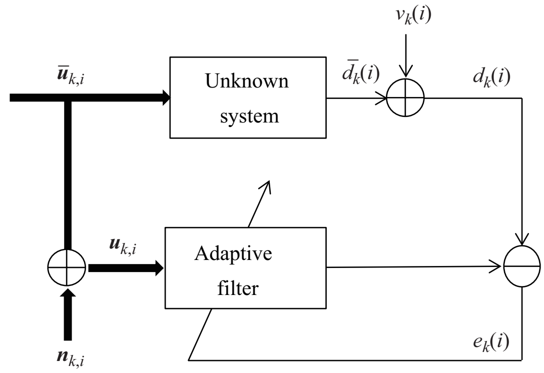

Figure 2 An FIR system identification model with noisy input-output data for each agent.

- $v_k(i)$ is independent and identically distributed (i.i.d) over time and independent over space, with zero-mean and variances $\sigma^2_{v_k}$.
- $\bar{u}_k(i)$, $n_l(j)$, and $v_p(m)$ are independent for all k, i, l, j, p, and m.

Vectors $\boldsymbol{w}^o \in \mathbb{R}^{L\times 1}$, $\bar{\boldsymbol{u}}_{k,i} \in \mathbb{R}^{L\times 1}$, $\boldsymbol{n}_{k,i} \in \mathbb{R}^{L\times 1}$, and $\boldsymbol{u}_{k,i} \in \mathbb{R}^{L\times 1}$ are defined as follows:

$$\boldsymbol{w}^o = [w_0\ w_1 \cdots w_{L-1}]^{\mathrm{T}}, \tag{3}$$

$$\bar{\boldsymbol{u}}_{k,i} = [\bar{u}_k(i)\ \bar{u}_k(i-1) \cdots \bar{u}_k(i-L+1)]^{\mathrm{T}}, \tag{4}$$

$$\boldsymbol{n}_{k,i} = [n_k(i)\ n_k(i-1) \cdots n_k(i-L+1)]^{\mathrm{T}}, \tag{5}$$

$$\boldsymbol{u}_{k,i} = [u_k(i)\ u_k(i-1) \cdots u_k(i-L+1)]^{\mathrm{T}} = \bar{\boldsymbol{u}}_{k,i} + \boldsymbol{n}_{k,i}. \tag{6}$$

Then the noisy FIR model can be expressed as

$$d_k(i) = \boldsymbol{u}^{\mathrm{T}}_{k,i}\boldsymbol{w}^o + z_k(i), \tag{7}$$

where

$$z_k(i) = v_k(i) - \boldsymbol{n}^{\mathrm{T}}_{k,i}\boldsymbol{w}^o. \tag{8}$$

Therefore, in this paper, the objective of the multi-agent network is to collaboratively obtain the consistent estimation of the unknown parameter $\boldsymbol{w}^o$ using the noisy measurements $u_k(i)$ and $d_k(i)$ from each agent k.

When applying the least-squares (LS) estimator, the unknown parameter $\boldsymbol{w}^o$ can be estimated by minimizing the global cost

$$\min_{\boldsymbol{w}} \sum_{j=1}^{i} \left(\sum_{k=1}^{M} |d_k(j) - \boldsymbol{u}^{\mathrm{T}}_{k,j}\boldsymbol{w}|^2 \right). \tag{9}$$

For each agent k, it degenerates into a local optimization problem expressed as follows:

$$\widehat{\boldsymbol{w}}_{k,i} = \arg\min_{\boldsymbol{w}} \sum_{j=1}^{i} \left(d_k(j) - \boldsymbol{u}^{\mathrm{T}}_{k,j}\boldsymbol{w}\right)^2. \tag{10}$$

According to the least-squares criterion [27], the solution of (10) is given by

$$\widehat{\boldsymbol{w}}^{\mathrm{LS}}_{k,i} = \left(\sum_{j=1}^{i} \boldsymbol{u}_{k,j}\boldsymbol{u}^{\mathrm{T}}_{k,j} \right)^{-1} \sum_{j=1}^{i} \boldsymbol{u}_{k,j} d_k(j). \tag{11}$$

Substituting (7) into (11), we can get

$$\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} = \boldsymbol{w}^o + \left(\sum_{j=1}^{i} \boldsymbol{u}_{k,j}\boldsymbol{u}_{k,j}^{\mathrm{T}}\right)^{-1} \sum_{j=1}^{i} \boldsymbol{u}_{k,j} z_k(j). \tag{12}$$

Taking the limit of (12), we have

$$\lim_{i\to\infty} \widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} = \boldsymbol{w}^o + \mathrm{E}\left[\boldsymbol{u}_{k,j}\boldsymbol{u}_{k,j}^{\mathrm{T}}\right]^{-1} \mathrm{E}\left[\boldsymbol{u}_{k,j} z_k(j)\right]. \tag{13}$$

Then the bias of agent k can be expressed as

$$\boldsymbol{b}_k = \mathrm{E}\left[\boldsymbol{u}_{k,j}\boldsymbol{u}_{k,j}^{\mathrm{T}}\right]^{-1} \mathrm{E}\left[\boldsymbol{u}_{k,j} z_k(j)\right]. \tag{14}$$

Considering the assumptions, we have

$$\mathrm{E}\left[\boldsymbol{u}_{k,j} z_k(j)\right] = -\sigma_{n_k}^2 \boldsymbol{w}^o. \tag{15}$$

At time i, $\boldsymbol{w}^o$ can be replaced by its existing estimate $\widehat{\boldsymbol{w}}_{k,i-1}$ from previous iteration $i-1$. According to (14) and (15), the estimates of the bias $\widehat{\boldsymbol{b}}_{k,i}$ can be computed by

$$\widehat{\boldsymbol{b}}_{k,i} = -i\sigma_{n_k}^2 \boldsymbol{P}_{k,i} \widehat{\boldsymbol{w}}_{k,i-1}, \tag{16}$$

where

$$\boldsymbol{P}_{k,i} = \left(\sum_{j=1}^{i} \boldsymbol{u}_{k,j}\boldsymbol{u}_{k,j}^{\mathrm{T}}\right)^{-1}. \tag{17}$$

From (16), we can learn that when considering the situation where both the input and the output of each agent are corrupted by additive noise, the LS estimator is biased, and the bias is induced by the input noise variance $\sigma_{n_k}^2$ of each agent k. When $\sigma_{n_k}^2$ is known a priori, the bias can be directly computed by (16). However, in many practical cases, $\sigma_{n_k}^2$ is unknown. Therefore, in order to obtain the unbiased estimation, it is crucial to work out a method that can estimate the input noise variance of each agent k.

In the following section, we firstly assume that $\sigma_{n_k}^2$ is available. A local noncooperative bias-compensated RLS (BC-RLS) algorithm is proposed to solve the optimization problem of (10). Then the algorithm is further developed in an incremental fashion to obtain the unbiased estimates of $\boldsymbol{w}^o$ by minimizing the global cost as (9). Later we consider the situation where $\sigma_{n_k}^2$ is unknown a priori, and a method is provided to estimate $\sigma_{n_k}^2$. After that, we propose two incremental-based algorithms, which can firstly estimate the unknown noise variance and then remove the effect of the noise-induced bias.

3 Incremental-based BC-RLS algorithms

For each agent k in the multi-agent network, the local RLS algorithm [28] can be expressed as

$$\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} = \widehat{\boldsymbol{w}}_{k,i-1}^{\mathrm{LS}} + \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}\left(d_k(i) - \boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{k,i-1}^{\mathrm{LS}}\right)}{1 + \boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}}, \tag{18}$$

$$\boldsymbol{P}_{k,i} = \boldsymbol{P}_{k,i-1} - \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}}{1 + \boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}}. \tag{19}$$

According to (13), (14), and (16), when the input noise variance is available, the local BC-RLS estimates $\widehat{\boldsymbol{w}}_{k,i}$ of each agent k can be obtained recursively as follows:

$$\widehat{\boldsymbol{w}}_{k,i} = \widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} - \widehat{\boldsymbol{b}}_{k,i} = \widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} + i\sigma_{n_k}^2 \boldsymbol{P}_{k,i} \widehat{\boldsymbol{w}}_{k,i-1}. \tag{20}$$

Eqs. (18)–(20) can be regarded as the noncooperative BC-RLS (Nco BC-RLS) algorithm of each local agent k. However, as mentioned above, the compensation for the bias can lead to an increase in the estimation variance, whereas the variance can be significantly reduced by combining information with neighbor agents. Therefore, it is beneficial to develop a distributed collaborative BC-RLS algorithm to remove the bias. Especially when the network only contains a few agents, or when the deployment of the agents is controlled, the cyclic techniques can be relevant options [12]. As a result, considering the features of incremental strategies, which have been discussed in the Introduction Section, we focus on incremental-based BC-RLS algorithms in this paper.

On the other hand, when evaluating the performance of distributed algorithms, the centralized algorithm can always be a good reference. Therefore, before we propose the incremental-based BC-RLS algorithms, we firstly give a brief introduction to the centralized BC-RLS algorithm.

3.1 Centralized BC-RLS algorithm

As shown in Figure 3, centralized solutions need a powerful fusion center to process the data from each agent. Based on different criterion, the fusion center allocates different weights to each agent. In this paper, we allocate the same weight to the data from every agent. After processing the data, the fusion center sends the estimated results back to every agent. Therefore, the centralized solution requires a large amount of communication and computation resources. With the size of the network increasing, the communications will also increase rapidly. When both the input and the output of each agent are corrupted by additive noise, the BC-RLS algorithm based on the centralized strategy is expressed as follows:

$$\widehat{\boldsymbol{w}}_i^{\mathrm{LS}} = \widehat{\boldsymbol{w}}_{i-1}^{\mathrm{LS}} + \frac{\frac{1}{M}\sum_{k=1}^{M}\boldsymbol{P}_{i-1}\boldsymbol{u}_{k,i}\left(d_k(i) - \boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{i-1}^{\mathrm{LS}}\right)}{1+\frac{1}{M}\sum_{k=1}^{M}\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{i-1}\boldsymbol{u}_{k,i}}, \tag{21}$$

$$\boldsymbol{P}_i = \boldsymbol{P}_{i-1} - \frac{\frac{1}{M}\sum_{k=1}^{M}\boldsymbol{P}_{i-1}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{i-1}}{1+\frac{1}{M}\sum_{k=1}^{M}\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{i-1}\boldsymbol{u}_{k,i}}, \tag{22}$$

$$\widehat{\boldsymbol{w}}_i = \widehat{\boldsymbol{w}}_i^{\mathrm{LS}} + i\sigma_n^2\boldsymbol{P}_i\widehat{\boldsymbol{w}}_{i-1}. \tag{23}$$

Eqs. (21)–(23) constitute the centralized BC-RLS (Cen BC-RLS) algorithm. It is shown that the centralized strategy requires global estimates $\widehat{\boldsymbol{w}}_i^{\mathrm{LS}}$ and $\widehat{\boldsymbol{w}}_{i-1}$ to achieve the update at every iteration. Therefore, the Cen BC-RLS algorithm is not a distributed solution.

3.2 Incremental BC-RLS algorithm

As shown in Figure 4, incremental schemes require us to determine a cyclic path at first. Then the agents are renumbered from agent 1 to agent M. For every iteration, the cyclic path starts from agent 1 and stops at agent M. Therefore, the data are processed in sequential order, where each agent communicates only with one neighbor. For each iteration i, the initial values of agent 1 are computed by the results from previous iteration $i-1$, while the estimated results in an incremental solution are the output of agent M. For example, the transmission of traditional RLS estimates $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ with incremental strategy can be described as

$$\begin{cases} \widehat{\boldsymbol{w}}_{1,i}^{\mathrm{LS}} &= \boxed{\widehat{\boldsymbol{w}}_{i-1}^{\mathrm{LS}}} &+ & J_1(\widehat{\boldsymbol{w}}_{i-1}^{\mathrm{LS}}),\\ \widehat{\boldsymbol{w}}_{2,i}^{\mathrm{LS}} &= \widehat{\boldsymbol{w}}_{1,i}^{\mathrm{LS}} &+ & J_2(\widehat{\boldsymbol{w}}_{1,i}^{\mathrm{LS}}),\\ \widehat{\boldsymbol{w}}_{3,i}^{\mathrm{LS}} &= \widehat{\boldsymbol{w}}_{2,i}^{\mathrm{LS}} &+ & J_3(\widehat{\boldsymbol{w}}_{2,i}^{\mathrm{LS}}),\\ & \vdots & & \\ \boxed{\widehat{\boldsymbol{w}}_{i}^{\mathrm{LS}}} &= \widehat{\boldsymbol{w}}_{M-1,i}^{\mathrm{LS}} &+ & J_M(\widehat{\boldsymbol{w}}_{M-1,i}^{\mathrm{LS}}). \end{cases} \tag{24}$$

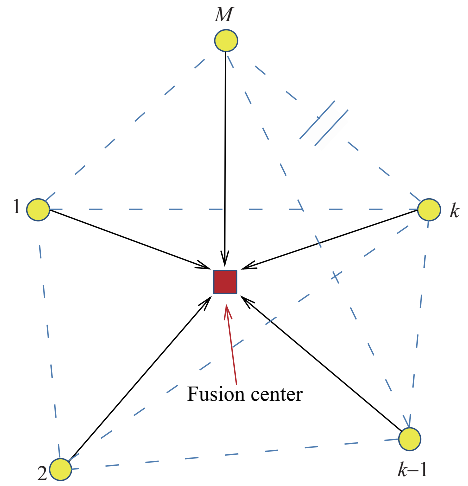

Figure 3 (Color online) Centralized cooperation strategies.

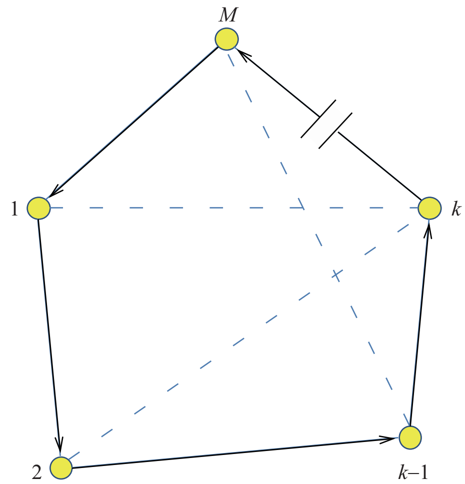

Figure 4 (Color online) Incremental cooperation strategies.

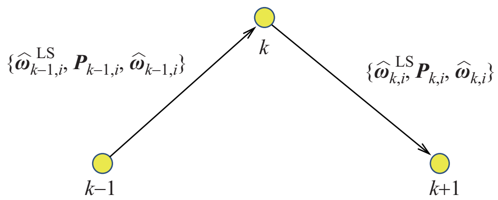

Figure 5 (Color online) Standard incremental cooperation strategy.

Here, for ease of presentation, we define $J_k(\widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i})$ as

$$J_k(\widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i}) = \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}\left(d_k(i) - \boldsymbol{u}^{\text{T}}_{k,i}\widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i}\right)}{1 + \boldsymbol{u}^{\text{T}}_{k,i}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}}. \tag{25}$$

When $k = 1$, we have

$$J_1(\widehat{\boldsymbol{w}}^{\text{LS}}_{i-1}) = \frac{\boldsymbol{P}_{M,i-1}\boldsymbol{u}_{1,i}\left(d_1(i) - \boldsymbol{u}^{\text{T}}_{1,i}\widehat{\boldsymbol{w}}^{\text{LS}}_{M,i-1}\right)}{1 + \boldsymbol{u}^{\text{T}}_{1,i}\boldsymbol{P}_{M,i-1}\boldsymbol{u}_{1,i}}. \tag{26}$$

In (24), at iteration i, agent 1 updates its estimate based on the existing estimate $\widehat{\boldsymbol{w}}^{\text{LS}}_{i-1}$, which is obtained from the output of agent M from previous iteration $\widehat{\boldsymbol{w}}^{\text{LS}}_{M,i-1}$. Similarly, $\widehat{\boldsymbol{w}}^{\text{LS}}_{i}$ is equal to $\widehat{\boldsymbol{w}}^{\text{LS}}_{M,i}$, which is the output of agent M at iteration i. For agent $k \in \{2, \ldots, M\}$, it updates its local estimate based on the estimate from its predecessor agent $k-1$, which can be expressed as

$$\widehat{\boldsymbol{w}}^{\text{LS}}_{k,i} = \widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i} + J_k(\widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i}). \tag{27}$$

From (25) and (27), we can learn that each agent k only needs its local information $\{\boldsymbol{u}_{k,i}, d_k(i)\}$ and the information $\{\widehat{\boldsymbol{w}}^{\text{LS}}_{k-1,i}, \boldsymbol{P}_{k-1,i}\}$ from its predecessor agent rather than the global estimates $\{\widehat{\boldsymbol{w}}^{\text{LS}}_{i}, \boldsymbol{P}_i\}$ to achieve the update. Therefore, the incremental solution is decentralized.

When it comes to BC-RLS algorithm, we also study the cooperation among agents by transmitting certain variables. According to (18)–(20), we learn that all of the variables $\widehat{\boldsymbol{w}}^{\text{LS}}_{k,i}$, $\boldsymbol{P}_{k,i}$, and $\widehat{\boldsymbol{w}}_{k,i}$ can be computed recursively with respect to time index i. In other words, the computation of these variables in Nco BC-RLS algorithm belongs to temporal processing. However, in order to obtain better estimates, we exploit the spatial diversity of a multi-agent network. According to incremental schemes, we transmit all of the variables $\widehat{\boldsymbol{w}}^{\text{LS}}_{k,i}$, $\boldsymbol{P}_{k,i}$, and $\widehat{\boldsymbol{w}}_{k,i}$ to the local neighbor, as shown in Figure 5. For each agent k, it achieves its local update based on the variables from its predecessor agent $k-1$ rather than its local variables from previous iteration $i-1$. Therefore, the iterations are implemented over space index k, and

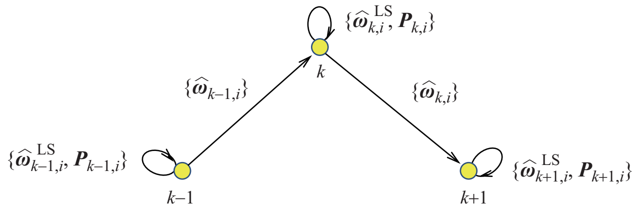

Figure 6 (Color online) Simplified incremental cooperation strategy.

the computation of each variable belongs to spatial processing. As a result, the spatial diversity is fully utilized. On the other hand, since all the variables computed out of the noisy raw data $\{d_k(i), \boldsymbol{u}_{k,i}\}$ are transmitted, the effect of the additive noise is also transmitted. In consequence, when implementing the incremental scheme, the input noise variance of each agent k is equal to the sum of the initial input noise variance in each agent before cooperation, which can be expressed as

$$\sigma^2_{n_k^{\rm coop}} = \sum_{k=1}^{M} \sigma^2_{n_k}, \tag{28}$$

where $\sigma^2_{n_k^{\rm coop}}$ denotes the real input noise variance at each agent when performing the incremental scheme.

The incremental BC-RLS (Inc BC-RLS) algorithm is expressed as follows:

$$\begin{aligned}
&\widehat{\boldsymbol{w}}_{0,i}^{\rm LS} \leftarrow \widehat{\boldsymbol{w}}_{i-1}^{\rm LS};\ \boldsymbol{P}_{0,i} \leftarrow \boldsymbol{P}_{i-1};\ \widehat{\boldsymbol{w}}_{0,i} \leftarrow \widehat{\boldsymbol{w}}_{i-1};\\
&\text{for } k = 1 : M\\
&\quad \widehat{\boldsymbol{w}}_{k,i}^{\rm LS} = \widehat{\boldsymbol{w}}_{k-1,i}^{\rm LS} + \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}(d_k(i) - \boldsymbol{u}_{k,i}^{\rm T}\widehat{\boldsymbol{w}}_{k-1,i}^{\rm LS})}{1 + \boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}};\\
&\quad \boldsymbol{P}_{k,i} = \boldsymbol{P}_{k-1,i} - \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}}{1 + \boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}};\\
&\quad \widehat{\boldsymbol{w}}_{k,i} = \widehat{\boldsymbol{w}}_{k,i}^{\rm LS} + i\sigma^2_{n_k^{\rm coop}}\boldsymbol{P}_{k,i}\widehat{\boldsymbol{w}}_{k-1,i};\\
&\text{end}\\
&\widehat{\boldsymbol{w}}_{i}^{\rm LS} \leftarrow \widehat{\boldsymbol{w}}_{M,i}^{\rm LS};\ \boldsymbol{P}_{i} \leftarrow \boldsymbol{P}_{M,i};\ \widehat{\boldsymbol{w}}_{i} \leftarrow \widehat{\boldsymbol{w}}_{M,i}.
\end{aligned} \tag{29}$$

From (29), we know that compared with the Cen BC-RLS algorithm, the Inc BC-RLS algorithm does not require global estimates. Each agent only uses the estimates from its local neighbor in every iteration. The Inc BC-RLS algorithm is therefore a fully distributed solution. At the same time, incremental solutions are expected to obtain even better estimates, since the output of an incremental network comes from the last agent M in the cycle, where the data from the whole network are used to update the local parameter. However, in some cases, the amount of communication requires further reducing. Therefore, under this condition, we also develop a simplified version of the Inc BC-RLS algorithm in the following subsection.

3.3 Simplified incremental BC-RLS algorithm

As shown in Figure 6, a simplified incremental strategy is proposed. In the simplified version, we merely transmit the bias-compensated estimates $\{\widehat{\boldsymbol{w}}_{k,i}\}$ instead of $\{\widehat{\boldsymbol{w}}_{k,i}^{\rm LS}, \boldsymbol{P}_{k,i}\}$ to the local neighbor. Therefore, only $\widehat{\boldsymbol{w}}_{k,i}$ is iterated over space index k. For the other variables, $\widehat{\boldsymbol{w}}_{k,i}^{\rm LS}$ and $\boldsymbol{P}_{k,i}$, they are iterated over time i with local noisy measurements only. Through this way, the amount of communication can be effectively reduced. On the other hand, when implementing the simplified incremental strategy, the effect of additive noise will not be transmitted, since only bias-compensated estimates $\{\widehat{\boldsymbol{w}}_{k,i}\}$ are spread. As a result, each agent in the network is only corrupted by its own background noise, and the effect of the additive noise will not be accumulated over the network. The simplified incremental BC-RLS (S-Inc BC-RLS) algorithm is summarized as (30).

Until now we have proposed two types of incremental-based BC-RLS algorithms. However, we should notice that no matter which incremental BC-RLS algorithm is adopted, it is necessary to guarantee a coincident transmission of $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$, as shown in Figures 5 and 6. From (29) and (30), we can learn that the bias-compensated estimate $\widehat{\boldsymbol{w}}_{k,i}$ is determined by both the first-term and the second-term on the right-hand side, where $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$ are included respectively. Here, $\sigma_{n_k}^2$ is not considered, since it is known a priori, which is available before cooperation. It is shown that only when both of $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$ are transmitted or not transmitted, the bias-compensated estimate $\widehat{\boldsymbol{w}}_{k,i}$ can be computed correctly. Otherwise, the asymmetric transmission will lead to incorrect results, since $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$ do not match with each other. In other words, when a coincident transmission cannot be guaranteed, the biased LS estimates and its noise-induced bias are not matched. Therefore, bias-compensated estimates cannot be computed correctly under this condition. On the other hand, in (29) and (30), we assume that the input noise variance $\sigma_{n_k}^2$ is available for each agent k. Then the bias-compensated estimates $\{\widehat{\boldsymbol{w}}_{k,i}\}$ can be computed directly. However, in many practical cases, the input noise variance is unknown a priori, and therefore the noise-induced bias cannot be obtained. In order to overcome this difficulty, in the following part, we present a method that can be used to estimate the unknown input noise variance over the multi-agent network:

$$\begin{aligned}
&\widehat{\boldsymbol{w}}_{0,i}^{\mathrm{LS}} \leftarrow \widehat{\boldsymbol{w}}_{i-1}^{\mathrm{LS}};\ \boldsymbol{P}_{0,i} \leftarrow \boldsymbol{P}_{i-1};\ \widehat{\boldsymbol{w}}_{0,i} \leftarrow \widehat{\boldsymbol{w}}_{i-1};\\
&\text{for } k=1:M\\
&\quad \widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} = \widehat{\boldsymbol{w}}_{k,i-1}^{\mathrm{LS}} + \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}(d_k(i)-\boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{k,i-1}^{\mathrm{LS}})}{1+\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}};\\
&\quad \boldsymbol{P}_{k,i} = \boldsymbol{P}_{k,i-1} - \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}}{1+\boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}};\\
&\quad \widehat{\boldsymbol{w}}_{k,i} = \widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}} + i\sigma_{n_k}^2\boldsymbol{P}_{k,i}\widehat{\boldsymbol{w}}_{k-1,i};\\
&\text{end}\\
&\widehat{\boldsymbol{w}}_{i}^{\mathrm{LS}} \leftarrow \widehat{\boldsymbol{w}}_{M,i}^{\mathrm{LS}};\ \boldsymbol{P}_{i} \leftarrow \boldsymbol{P}_{M,i};\ \widehat{\boldsymbol{w}}_{i} \leftarrow \widehat{\boldsymbol{w}}_{M,i}.
\end{aligned} \tag{30}$$

3.4 Input noise variance estimation

Based on the above discussion, we know that when both the input and the output of each agent are corrupted by additive noise, the traditional LS estimator is biased. It shows that the bias is induced by the input noise variance $\sigma_{n_k}^2$ and can be removed as long as $\sigma_{n_k}^2$ is available. Nevertheless, the input noise variance can be unknown a priori in practice. Therefore, in order to obtain bias-compensated estimates, it is necessary to work out a method to compute the unknown input noise variance of each agent k. We present a method that can obtain the estimates of $\sigma_{n_k}^2$ over the network in real-time as follows.

Introduce an auxiliary vector $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2 \cdots \alpha_L]^{\mathrm{T}}$, and define its estimator as

$$\widehat{\boldsymbol{\alpha}}_{k,i} = \left(\sum_{j=1}^{i}\boldsymbol{u}_{k,j}\boldsymbol{u}_{k,j}^{\mathrm{T}}\right)^{-1}\sum_{j=1}^{i}\boldsymbol{u}_{k,j}d_k(j-1). \tag{31}$$

Then the one-step backward predictor can be expressed as

$$\widehat{d}_k(j-1|\boldsymbol{u}_{k,j}) = \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}. \tag{32}$$

The backward prediction error (BPE) $\eta_k(j)$ can therefore be defined as

$$\eta_k(j) = d_k(j-1) - \widehat{d}_k(j-1|\boldsymbol{u}_{k,j}) = d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}. \tag{33}$$

Similarly, the least-squares estimation error (LSE) $e_k(j)$ is defined as

$$e_k(j) = d_k(j) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}. \tag{34}$$

Then we can obtain the cross-correlation function $f_k(i)$ between LSE and BPE as

$$f_k(i) = \sum_{j=1}^{i} e_k(j)\eta_k(j) = \sum_{j=1}^{i} \left(d_k(j) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}\right)\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right). \tag{35}$$

According to the principle of orthogonality in least-squares condition, we have

$$\sum_{j=1}^{i} \boldsymbol{u}_{k,j}\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right) = \boldsymbol{0}. \tag{36}$$

The cross-correlation function $f_k(i)$ can be rewritten as

$$f_k(i) = \sum_{j=1}^{i} d_k(j)\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right). \tag{37}$$

Substituting (7) and (8) to (37), we get

$$\begin{aligned} f_k(i) &= \sum_{j=1}^{i} \left(\boldsymbol{u}_{k,j}^{\mathrm{T}}\boldsymbol{w}^o + z_k(j)\right)\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right) \\ &= \sum_{j=1}^{i} z_k(j)\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right) \\ &= \sum_{j=1}^{i} \left(v_k(j) - \boldsymbol{n}_{k,j}^{\mathrm{T}}\boldsymbol{w}^o\right)\left(d_k(j-1) - \boldsymbol{u}_{k,j}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right). \end{aligned} \tag{38}$$

According to the assumptions, when $i \to \infty$, we have

$$\lim_{i\to\infty} \frac{f_k(i)}{i} = \sigma_{n_k}^2 \boldsymbol{w}^{o\mathrm{T}} \lim_{i\to\infty} \widehat{\boldsymbol{\alpha}}_{k,i}. \tag{39}$$

Therefore, the unknown noise variance can be estimated in real-time as follows:

$$\widehat{\sigma}_{n_k}^2(i) = \frac{\widehat{f}_k(i)}{i\widehat{\boldsymbol{w}}_{k,i-1}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}}, \tag{40}$$

where $\widehat{f}_k(i)$ and $\widehat{\boldsymbol{\alpha}}_{k,i}$ can be recursively computed by

$$\begin{aligned} \widehat{\boldsymbol{\alpha}}_{k,i} &= \widehat{\boldsymbol{\alpha}}_{k,i-1} + \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}(d_k(i-1) - \boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i-1})}{1 + \boldsymbol{u}_{k,i}^{\mathrm{T}}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}}, \\ \widehat{f}_k(i) &= \widehat{f}_k(i-1) + \left(d_k(i) - \boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}\right)\left(d_k(i-1) - \boldsymbol{u}_{k,i}^{\mathrm{T}}\widehat{\boldsymbol{\alpha}}_{k,i}\right). \end{aligned} \tag{41}$$

According to (40) and (41), we obtain the estimate of $\sigma_{n_k}^2$ in real-time. At the same time, two variables, $\widehat{\boldsymbol{\alpha}}_{k,i}$ and $\widehat{f}_k(i)$, are introduced. When adopting incremental strategies, the transmission of these variables can also influence the algorithm performance, which deserves a further discussion as follows.

Under the condition where the input noise variance is unknown a priori, we replace $\sigma_{n_k}^2$ by $\widehat{\sigma}_{n_k}^2(i)$ in (29) and (30). As a result, besides $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$, the transmission of $\widehat{\sigma}_{n_k}^2(i)$ should also be considered. Therefore, from (40) and (41), the transmission of $\widehat{\boldsymbol{\alpha}}_{k,i}$ and $\widehat{f}_k(i)$ can have significant effect on the performance of the proposed algorithms. Similarly to $\widehat{\boldsymbol{w}}_{k,i}^{\mathrm{LS}}$ and $\boldsymbol{P}_{k,i}$, $\widehat{\boldsymbol{\alpha}}_{k,i}$ and $\widehat{f}_k(i)$ can also be computed recursively over time index i. According to the two transmission fashions shown in Figures 5 and 6, $\widehat{\boldsymbol{\alpha}}_{k,i}$ and $\widehat{f}_k(i)$ are transmitted with other variables in the Inc BC-RLS algorithm, while they are not transmitted in the S-Inc BC-RLS algorithm. Therefore, in the Inc BC-RLS algorithm, the estimated input noise variance is equal to the sum of the initial input noise variance in each agent. However, in the S-Inc BC-RLS algorithm, the estimated input noise variance of each agent k is only equal to its own input noise variance.

3.5 Communication costs

In this part, we study the amount of communication required by the Inc BC-RLS algorithm and the S-Inc BC-RLS algorithm for every iteration at every agent. In the Inc BC-RLS algorithm, $\widehat{\boldsymbol{w}}_{k,i}^{\rm LS}$, $\boldsymbol{P}_{k,i}$, $\widehat{\boldsymbol{\alpha}}_{k,i}$, $\widehat{f}_k(i)$, and $\widehat{\boldsymbol{w}}_{k,i}$ are transmitted to the adjacent agent in the cycle. Considering their dimensions, we can know that these variables contain $(L + L \times L + L + 1 + L)$ scalars. Therefore, the total number of communications required at every iteration per agent is $\left(L^2 + 3L + 1\right)$. As a result, the Inc BC-RLS algorithm requires transmission complexity $O(L^2)$. In the S-Inc BC-RLS algorithm, only bias-compensated estimate $\widehat{\boldsymbol{w}}_{k,i}$ is transmitted. Thus, the total number of communicated scalars per iteration per agent is merely L. The transmission complexity required by the S-Inc BC-RLS algorithm is $O(L)$. Compared to the Inc BC-RLS algorithm, the amount of communication is reduced by $\left(L^2 + 2L + 1\right)$. Notice that when adopting incremental-based algorithms, the number of communications for every iteration at every agent is uncorrelated with the size of the network, while it will increase with the size of centralized network [2]. The S-Inc BC-RLS algorithm can thus be a good candidate when communications are highly constrained.

3.6 Summary of the two proposed algorithms

The two incremental-based algorithms are summarized in Algorithms 1 and 2 respectively.

Algorithm 1 Inc BC-RLS algorithm

1: **Initialization**

$\widehat{\boldsymbol{w}}_{k,0} = \boldsymbol{0}, \widehat{\boldsymbol{w}}_{k,0}^{\rm LS} = \boldsymbol{0}, \widehat{\boldsymbol{\alpha}}_{k,0} = \boldsymbol{0}, \widehat{f}_k(0) = 0;$

$\boldsymbol{P}_{k,0} = \delta^{-1}\boldsymbol{I}_L$, δ is a small positive number and $\boldsymbol{I}_L$ is an $L \times L$ identity matrix;

2: **For each time $i > 0$, do**

$\widehat{\boldsymbol{w}}_{0,i}^{\rm LS} \leftarrow \widehat{\boldsymbol{w}}_{i-1}^{\rm LS};\ \boldsymbol{P}_{0,i} \leftarrow \boldsymbol{P}_{i-1};\ \widehat{\boldsymbol{w}}_{0,i} \leftarrow \widehat{\boldsymbol{w}}_{i-1};\ \widehat{\boldsymbol{\alpha}}_{0,i} \leftarrow \widehat{\boldsymbol{\alpha}}_{i-1};\ \widehat{f}_0(i) \leftarrow \widehat{f}(i-1);$

for $k = 1 : M$

$\widehat{\boldsymbol{w}}_{k,i}^{\rm LS} = \widehat{\boldsymbol{w}}_{k-1,i}^{\rm LS} + \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}(d_k(i) - \boldsymbol{u}_{k,i}^{\rm T}\widehat{\boldsymbol{w}}_{k-1,i}^{\rm LS})}{1+\boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}};$

$\boldsymbol{P}_{k,i} = \boldsymbol{P}_{k-1,i} - \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}}{1+\boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}};$

$\widehat{\boldsymbol{\alpha}}_{k,i} = \widehat{\boldsymbol{\alpha}}_{k-1,i} + \frac{\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}(d_k(i-1) - \boldsymbol{u}_{k,i}^{\rm T}\widehat{\boldsymbol{\alpha}}_{k-1,i})}{1+\boldsymbol{u}_{k,i}^{\rm T}\boldsymbol{P}_{k-1,i}\boldsymbol{u}_{k,i}};$

$\widehat{f}_k(i) = \widehat{f}_{k-1}(i) + (d_k(i) - \boldsymbol{u}_{k,i}^{\rm T}\widehat{\boldsymbol{w}}_{k,i}^{\rm LS})(d_k(i-1) - \boldsymbol{u}_{k,i}^{\rm T}\widehat{\boldsymbol{\alpha}}_{k,i});$

$\widehat{\sigma}_{n_k}^2(i) = \frac{\widehat{f}_k(i)}{i\widehat{\boldsymbol{w}}_{k-1,i}^{\rm T}\widehat{\boldsymbol{\alpha}}_{k,i}};$

$\widehat{\boldsymbol{w}}_{k,i} = \widehat{\boldsymbol{w}}_{k,i}^{\rm LS} + i\widehat{\sigma}_{n_k}^2(i)\boldsymbol{P}_{k,i}\widehat{\boldsymbol{w}}_{k-1,i};$

end

$\widehat{\boldsymbol{w}}_i^{\rm LS} \leftarrow \widehat{\boldsymbol{w}}_{M,i}^{\rm LS};\ \boldsymbol{P}_i \leftarrow \boldsymbol{P}_{M,i};\ \widehat{\boldsymbol{w}}_i \leftarrow \widehat{\boldsymbol{w}}_{M,i};\ \widehat{\boldsymbol{\alpha}}_i \leftarrow \widehat{\boldsymbol{\alpha}}_{M,i};\ \widehat{f}(i) \leftarrow \widehat{f}_M(i);$

3: **Repeat Step 2 until convergence.**

4 Simulation results

The multi-agent network topology is shown as Figure 7, which consists of $M = 10$ agents. A cyclic path is determined to transmit the data by the direction of arrows. For every agent in the network, a noisy FIR model is constructed with $L = 5$ and $\boldsymbol{w}^o = [-0.3; -0.9; 0.8; -0.7; 0.6]$. For each agent $k \in \{1, 2, \ldots, M\}$, the noise-free input $\bar{u}_k(i)$ and the input noise $n_k(i)$ are stationary white noise processes with variances $\sigma_{\bar{u}_k}^2 = 1$ and $\sigma_{n_k}^2$ set as shown in Figure 8. Simulation results are the averages of 100 independent trials, and the number of iterations is chosen as 5000.

Table 1 shows the estimates of $\boldsymbol{w}^o$ obtained by the incremental RLS (Inc RLS) solution in [16], the Inc BC-RLS algorithm, and the S-Inc BC-RLS algorithm. We can learn from Table 1 that the results of the Inc RLS algorithm are biased when both the input and the output are corrupted by additive noise. On the contrary, the proposed incremental-based BC-RLS algorithms can obtain the consistent estimation of the unknown parameter. Besides, with the increase of the iteration number, the estimation accuracy is higher.

Algorithm 2 S-Inc BC-RLS algorithm

1: **Initialization**

$\widehat{\boldsymbol{w}}_{k,0} = \boldsymbol{0}, \widehat{\boldsymbol{w}}^{\rm LS}_{k,0} = \boldsymbol{0}, \widehat{\boldsymbol{\alpha}}_{k,0} = \boldsymbol{0}, \widehat{f}_k(0) = 0;$

$\boldsymbol{P}_{k,0} = \delta^{-1}\boldsymbol{I}_L$, δ is a small positive number and $\boldsymbol{I}_L$ is an $L \times L$ identity matrix;

2: **For each time** $i > 0$**, do**

$\widehat{\boldsymbol{w}}^{\rm LS}_{0,i} \leftarrow \widehat{\boldsymbol{w}}^{\rm LS}_{i-1};\ \boldsymbol{P}_{0,i} \leftarrow \boldsymbol{P}_{i-1};\ \widehat{\boldsymbol{w}}_{0,i} \leftarrow \widehat{\boldsymbol{w}}_{i-1};\ \widehat{\boldsymbol{\alpha}}_{0,i} \leftarrow \widehat{\boldsymbol{\alpha}}_{i-1};\ \widehat{f}_0(i) \leftarrow \widehat{f}(i-1);$

for $k = 1 : M$

$\widehat{\boldsymbol{w}}^{\rm LS}_{k,i} = \widehat{\boldsymbol{w}}^{\rm LS}_{k,i-1} + \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}(d_k(i) - \boldsymbol{u}^{\rm T}_{k,i}\widehat{\boldsymbol{w}}^{\rm LS}_{k,i-1})}{1+\boldsymbol{u}^{\rm T}_{k,i}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}};$

$\boldsymbol{P}_{k,i} = \boldsymbol{P}_{k,i-1} - \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}\boldsymbol{u}^{\rm T}_{k,i}\boldsymbol{P}_{k,i-1}}{1+\boldsymbol{u}^{\rm T}_{k,i}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}};$

$\widehat{\boldsymbol{\alpha}}_{k,i} = \widehat{\boldsymbol{\alpha}}_{k,i-1} + \frac{\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}(d_k(i-1) - \boldsymbol{u}^{\rm T}_{k,i}\widehat{\boldsymbol{\alpha}}_{k,i-1})}{1+\boldsymbol{u}^{\rm T}_{k,i}\boldsymbol{P}_{k,i-1}\boldsymbol{u}_{k,i}};$

$\widehat{f}_k(i) = \widehat{f}_k(i-1) + (d_k(i) - \boldsymbol{u}^{\rm T}_{k,i}\widehat{\boldsymbol{w}}^{\rm LS}_{k,i})(d_k(i-1) - \boldsymbol{u}^{\rm T}_{k,i}\widehat{\boldsymbol{\alpha}}_{k,i});$

$\widehat{\sigma}^2_{n_k}(i) = \frac{\widehat{f}_k(i)}{i\widehat{\boldsymbol{w}}^{\rm T}_{k-1,i}\widehat{\boldsymbol{\alpha}}_{k,i}};$

$\widehat{\boldsymbol{w}}_{k,i} = \widehat{\boldsymbol{w}}^{\rm LS}_{k,i} + i\widehat{\sigma}^2_{n_k}(i)\boldsymbol{P}_{k,i}\widehat{\boldsymbol{w}}_{k-1,i};$

end

$\widehat{\boldsymbol{w}}^{\rm LS}_{i} \leftarrow \widehat{\boldsymbol{w}}^{\rm LS}_{M,i};\ \boldsymbol{P}_{i} \leftarrow \boldsymbol{P}_{M,i};\ \widehat{\boldsymbol{w}}_{i} \leftarrow \widehat{\boldsymbol{w}}_{M,i};\ \widehat{\boldsymbol{\alpha}}_{i} \leftarrow \widehat{\boldsymbol{\alpha}}_{M,i};\ \widehat{f}(i) \leftarrow \widehat{f}_M(i);$

3: **Repeat Step 2 until convergence.**

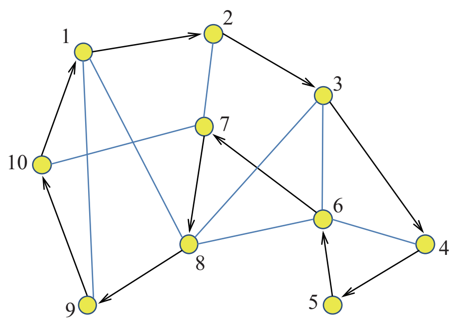

Figure 7 (Color online) The multi-agent network topology.

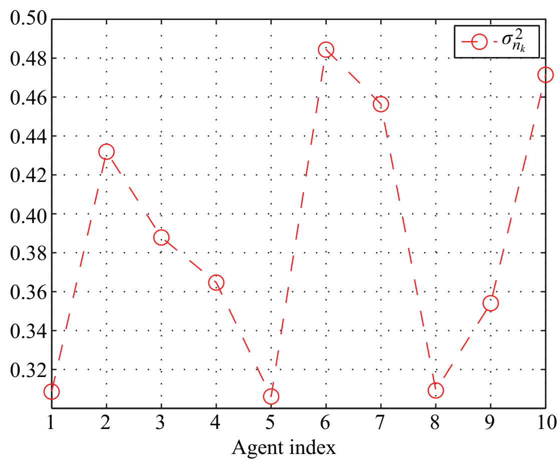

Figure 8 (Color online) Input noise variance profile.

Table 1 Estimates of $\boldsymbol{w}^o$

i	True values	w^o_1	w^o_2	w^o_3	w^o_4	w^o_5
		−0.3	−0.9	0.8	−0.7	0.6
	Inc RLS	−0.2172	−0.6489	0.5758	−0.5043	0.4322
3000	Inc BC-RLS	−0.3023	−0.9018	0.8003	−0.7009	0.6004
	S-Inc BC-RLS	−0.3024	−0.9003	0.8007	-0.6996	0.6011
	Inc RLS	−0.2172	−0.6485	0.5756	-0.5040	0.4323
4000	Inc BC-RLS	−0.3021	−0.9010	0.7999	−0.7002	0.6005
	S-Inc BC-RLS	−0.3014	−0.9013	0.8008	−0.6989	0.6007
	Inc RLS	−0.2169	−0.6485	0.5759	−0.5041	0.4323
5000	Inc BC-RLS	−0.3014	−0.9009	0.8000	−0.7003	0.6005
	S-Inc BC-RLS	−0.3009	−0.9015	0.8016	−0.6997	0.6016

Figure 9 shows the estimated results of the unknown input noise variance at agent M obtained by the Inc BC-RLS algorithm. Similar results can also be obtained at other agents but omitted due to space limitations. We can learn that the input noise variance can be estimated in real-time with high accuracy. Its value is equal to the sum of the initial input noise variance at every agent, which matches well

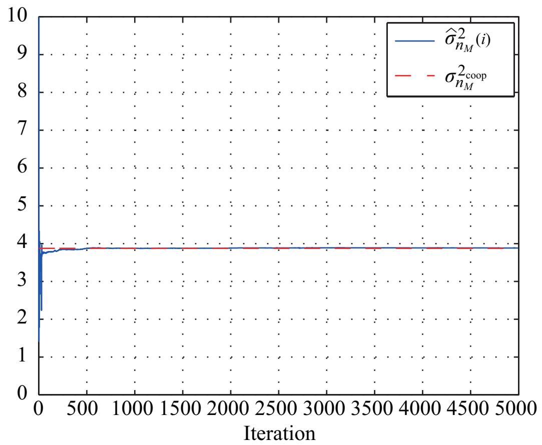

Figure 9 (Color online) The estimated results of the input noise variance at agent M obtained by the Inc BC-RLS algorithm.

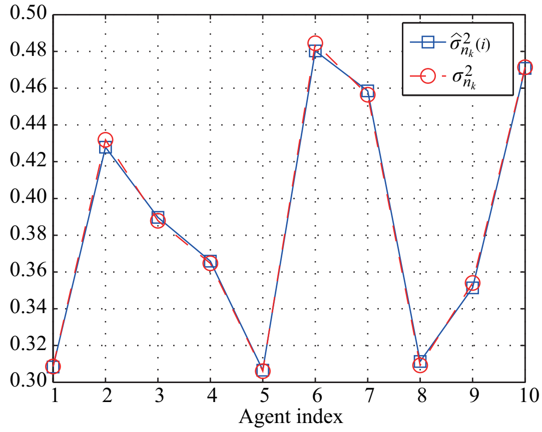

Figure 10 (Color online) The estimated results of the input noise variances for each agent in steady-state obtained by the S-Inc BC-RLS algorithm.

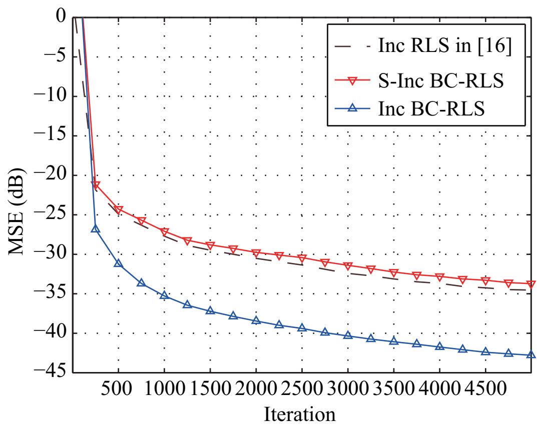

Figure 11 (Color online) MSE curves obtained by the Inc RLS, the Inc BC-RLS, and the S-Inc BC-RLS algorithms when $\sigma^2_{n_k} = 0$.

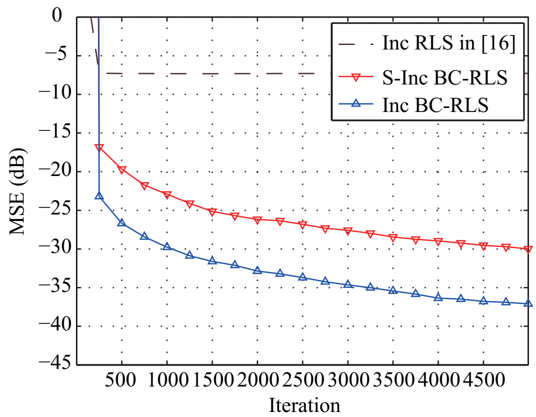

Figure 12 (Color online) MSE curves obtained by the Inc RLS, the Inc BC-RLS, and the S-Inc BC-RLS algorithms when $\sigma^2_{n_k} \neq 0$.

with (28). Figure 10 shows the estimates of the input noise variances at every agent in steady-state when adopting the S-Inc BC-RLS algorithm, which illustrates the effectiveness of the noise estimation method in the proposed algorithm.

In this paper, we also use mean square error (MSE) to stand for the estimation accuracy, and it is defined as

$$\mathrm{MSE} = ||\hat{\boldsymbol{w}}_{k,i} - \boldsymbol{w}^o||. \tag{42}$$

Figure 11 shows the MSE curves of the existing Inc RLS algorithm in [16], the Inc BC-RLS algorithm, and the S-Inc BC-RLS algorithm when regressors are noise-free. It shows that when $\sigma^2_{n_k} = 0$, all of the Inc RLS algorithm and the proposed algorithms can achieve the unbiased estimation of the unknown parameter. Meanwhile, it shows that the Inc BC-RLS algorithm can obtain the estimates with smaller MSE values. Figure 12 shows the MSE curves obtained by these algorithms when $\sigma^2_{n_k}$ is set as shown in Figure 8. It can be learned that the Inc RLS algorithm is biased when $\sigma^2_{n_k} \neq 0$, whereas the proposed Inc BC-RLS algorithm and its simplified version can effectively remove the effect of noise-induced bias and achieve the unbiased estimation.

Figure 13 depicts the MSE curves of the proposed algorithms under the conditions where $\sigma^2_{n_k}$ is known and unknown respectively. From Figure 13 we can learn that for the S-Inc BC-RLS algorithm, when $\sigma^2_{n_k}$

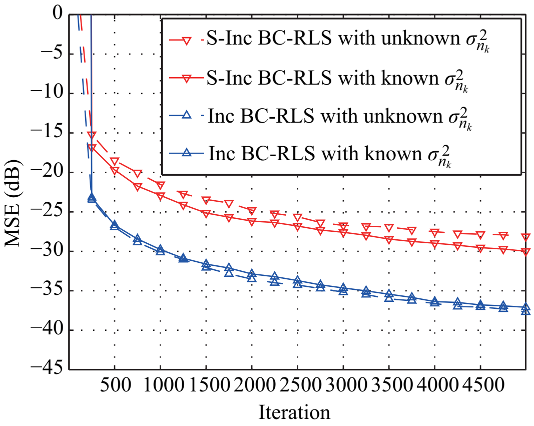

Figure 13 (Color online) MSE curves of the Inc BC-RLS and the S-Inc BC-RLS algorithms under the conditions where $\sigma_{n_k^2}$ is known and unknown.

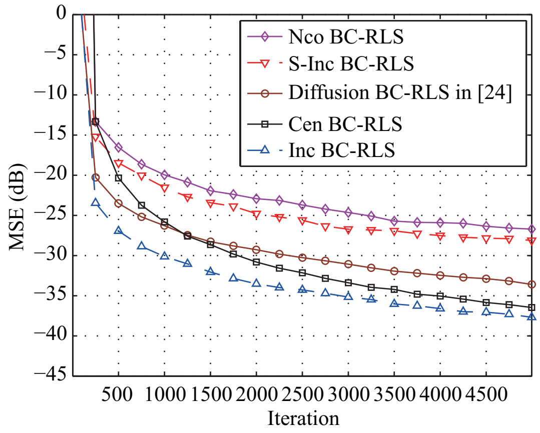

Figure 14 (Color online) MSE curves of the BC-RLS algorithms with different cooperation schemes.

is known, it can achieve the consistent estimation of the unknown parameter. When $\sigma^2_{n_k}$ is unknown, the S-Inc BC-RLS algorithm can still obtain the unbiased estimation with an acceptable estimation accuracy. On the other hand, for the Inc BC-RLS algorithm, the two curves almost overlap with each other, which indicates that the Inc BC-RLS algorithm can obtain same good estimates no matter whether $\sigma^2_{n_k}$ is known or unknown a priori. Therefore, compared to the algorithms applied in the situation where $\sigma^2_{n_k}$ is known a priori, the proposed algorithms can still work effectively when $\sigma^2_{n_k}$ is unknown.

Figure 14 shows the curves obtained from the BC-RLS algorithms with different cooperation schemes. It shows that the proposed incremental-based BC-RLS algorithms can obtain more accurate estimates than the Nco BC-RLS algorithm, which illustrates the benefits of distributed cooperation. Compared to the diffusion BC-RLS algorithm in [24], the MSE values of the Inc BC-RLS algorithm are smaller. Compared to the Cen BC-RLS algorithm, the Inc BC-RLS algorithm can achieve the unbiased estimation with higher estimation accuracy and faster convergence rate.

5 Conclusion

In this paper, two types of incremental-based bias-compensated recursive least-squares algorithms are proposed to achieve the collaborative parameter estimation over multi-agent networks. Considering the situation where both input and output of each agent are corrupted by unknown additive noise, many existing estimators are biased or cannot work. It shows that the bias is induced by the input noise variance. The proposed algorithms can obtain the unknown input noise variance and remove the effect of the noise-induced bias in an incremental fashion. Then the unbiased estimation of the unknown parameter can be achieved, which is proved by computer simulation. Compared to the noncooperative algorithm, the distributed incremental algorithms can obtain better estimates, which illustrate the benefits of distributed cooperation. Meanwhile, the Inc BC-RLS algorithm outperforms both centralized BC-RLS and diffusion BC-RLS algorithms in terms of estimation accuracy and convergence rate. Even though the estimation accuracy of the S-Inc BC-RLS algorithm is not as good as that of the Inc BC-RLS algorithm, its amount of communication is reduced by $L^2 + 2L + 1$. Therefore, the S-Inc BC-RLS algorithm can also be a relevant option when communication resources are limited.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 61421001).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Sayed A H. Adaptation, learning, and optimization over networks. Found Trends March Learn, 2014, 7: 312–318
2 Cattivelli F S, Sayed A H. Analysis of spatial and incremental LMS processing for distributed estimation. IEEE Trans Signal Process, 2011, 59: 1465–1480
3 Estrin D, Girod L, Pottie G, et al. Instrumenting the world with wireless sensor networks. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Salt Lake City, 2001. 2033–2036
4 Sayed A H, Lopes C G. Distributed processing over adaptive networks. In: Proceedings of the 9th International Symposium on Signal Processing and its Applications (ISSPA), Sharjah, 2007. 1–3
5 Brust M R, Akbas M I, Turgut D. Multi-hop localization system for environmental monitoring in wireless sensor and actor networks. Concurrency Comput Pract Exper, 2013, 25: 701–717
6 Lopes C G, Sayed A H. Incremental adaptive strategies over distributed networks. IEEE Trans Signal Process, 2007, 55: 4064–4077
7 Khalili A, Tinati M A, Rastegarnia A. Analysis of incremental RLS adaptive networks with noisy links. IEICE Electron Express, 2011, 8: 623–628
8 Bogdanovic N, Plata-Chaves J, Berberidis K. Distributed incremental-based LMS for node-specific parameter estimation over adaptive networks. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, 2013. 5425–5429
9 Plata -Chaves J, Bogdanovic N, Berberidis K. Distributed incremental-based RLS for node-specific parameter estimation over adaptive networks. In: Proceedings of the 21st European Signal Processing Conference (EUSIPCO), Marrakech, 2013. 1–5
10 Khalili A, Rastegarnia A, Bazzi W M, et al. Derivation and analysis of incremental augmented complex least mean square algorithm. IET Signal Process, 2015, 9: 312–319
11 Bogdanovic N, Plata -Chaves J, Berberidis K. Distributed incremental-based LMS for node-specific adaptive parameter estimation. IEEE Trans Signal Process, 2014, 62: 5382–5397
12 Khalili A, Rastegarnia A, Bazzi W M, et al. Tracking performance of incremental augmented complex least mean square adaptive network in the presence of model non-stationarity. IET Signal Process, 2016, 10: 798–804
13 Rastegarnia A, Tinati M A, Khalili A. Performance analysis of quantized incremental LMS algorithm for distributed adaptive estimation. Signal Process, 2010, 90: 2621–2627
14 Rastegarnia A, Khalili A. Incorporating observation quality information into the incremental LMS adaptive networks. Arab J Sci Eng, 2014, 39: 987–995
15 Tu S Y, Sayed A H. Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks. IEEE Trans Signal Process, 2012, 60: 6217–6234
16 Sayed A H, Lopes C G. Adaptive processing over distributed networks. IEICE Trans Fund Electron Commun Comput Sci, 2007, 90: 1504–1510
17 Cattivelli F S, Lopes C G, Sayed A H. Diffusion recursive least-squares for distributed estimation over adaptive networks. IEEE Trans Signal Process, 2008, 56: 1865–1877
18 Abdolee R, Champagne B, Sayed A H. A diffusion LMS strategy for parameter estimation in noisy regressor applications. In: Proceedings of the 20th European Signal Processing Conference (EUSIPCO), Bucharest, 2012. 749–753
19 Sayed A H, Tu S Y, Chen J S, et al. Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior. IEEE Signal Process Mag, 2013, 30: 155–171
20 Chen J S, Sayed A H. Distributed Pareto optimization via diffusion strategies. IEEE J Sel Top Sign Proces, 2013, 7: 205–220
21 Sayed A H. Diffusion adaptation over networks. In: Academic Press Library in Signal Processing. Boston: Academic Press Elsevier, 2014. 323–454
22 Cattivelli F S, Sayed A H. Diffusion LMS strategies for distributed estimation. IEEE Trans Signal Process, 2010, 58: 1035–1048
23 Bertrand A, Moonen M, Sayed A H. Diffusion-based bias-compensated RLS for distributed estimation over adaptive sensor networks. In: Proceedings of the 19th European Signal Processing Conference (EUSIPCO), Barcelona, 2011. 1025–1029
24 Bertrand A, Moonen M, Sayed A H. Diffusion bias-compensated RLS estimation over adaptive networks. IEEE Trans Signal Process, 2011, 59: 5212–5224
25 Zhao X C, Tu S Y, Sayed A H. Diffusion adaptation over networks under imperfect information exchange and non-stationary data. IEEE Trans Signal Process, 2012, 60: 3460–3475
26 Jia L J, Tao R, Wang Y, et al. Forward/backward prediction solution for adaptive noisy FIR filtering. Sci China Ser F-Inf Sci, 2009, 52: 1007–1014
27 Ljung L. System Identification. Boston: Birkhauser, 1998. 163–173
28 Haykin S. Adaptive Filter Theory. 4th ed. Upper Saddle River: Prentice Hall, 2002. 442–444