# Distributed Low-Rank Adaptive Estimation Algorithms Based on Alternating Optimization

**Article** *in* Signal Processing · September 2017

DOI: 10.1016/j.sigpro.2017.09.023



**3 authors:**

Songcen Xu
Huawei Technologies
**37** PUBLICATIONS **1,016** CITATIONS
SEE PROFILE

Rodrigo De Lamare
Pontifícia Universidade Católica do Rio de Janeiro
**699** PUBLICATIONS **11,184** CITATIONS
SEE PROFILE

H. Vincent Poor
Princeton University
**3,053** PUBLICATIONS **133,270** CITATIONS
SEE PROFILE

# Distributed Low-Rank Adaptive Estimation Algorithms Based on Alternating Optimization

Songcen Xu*, Rodrigo C. de Lamare, *Senior Member, IEEE,* and H. Vincent Poor, *Fellow, IEEE*

***Abstract*—This paper presents a novel distributed low-rank scheme and adaptive algorithms for distributed estimation over wireless networks. The proposed distributed scheme is based on a transformation that performs dimensionality reduction at each agent of the network followed by transmission of a reduced set of parameters to other agents and reduced-dimension parameter estimation. Distributed low-rank joint iterative estimation algorithms based on alternating optimization strategies are developed, which can achieve significantly reduced communication overhead and improved performance when compared with existing techniques. A computational complexity analysis of the proposed and existing low-rank algorithms is presented along with an analysis of the convergence of the proposed techniques. Simulations illustrate the performance of the proposed strategies in applications of wireless sensor networks and smart grids.**

***Index Terms*—Dimensionality reduction, distributed estimation techniques, low-rank algorithms, wireless sensor networks, smart grids.**

## I. Introduction

DISTRIBUTED strategies have become fundamental for parameter estimation in wireless networks and applications such as sensor networks [1], [2], [3] and smart grids [4], [5]. Distributed techniques deal with the extraction of information from data collected at nodes that are distributed over a geographic area [1]. In this context, a specific sensor node or agent in the network collects processed data from its neighbors and combines them with its local information to generate improved estimates. However, when the unknown parameter vector to be estimated has a large number of parameters, the network requires a large communication bandwidth between neighboring nodes to transmit their local estimates. This problem limits the usefulness of existing algorithms in applications with large data sets as the convergence speed is dependent on the number of parameters [2], [6], [7]. Hence, distributed dimensionality reduction has become an important tool for distributed inference problems with large data sets.

In order to perform dimensionality reduction or compression, several algorithms have been proposed in the literature in the context of distributed quantized Kalman filtering [8], [9], quantized consensus algorithms [10], distributed principal subspace estimation [11], the single bit strategy [12] and Krylov subspace optimization techniques [13]. However, these distributed algorithms [8]-[13] have drawbacks such as high computational complexity, unsatisfactory performance and implementation issues. Available distributed approaches for dimensionality reduction or compression [8]-[13] have trade-offs between the amount of cooperation, communication and system performance. This calls for the development of cost-effective techniques that can approach the performance of theoretical bounds for parameter estimation, have flexibility and high-compression capability, and exhibit low computational complexity. In this context, low-rank techniques are powerful tools to perform dimensionality reduction, which have been applied to spread-spectrum systems [14], [15], [16], [17], [18], [19], multi-input-multi-output (MIMO) systems [20], [21] and beamforming [22], [23], [24]. However, limited research has been carried out on distributed low-rank estimation, in which the distributed principal subspace estimation [11] and the Krylov subspace optimization [13] techniques are recent contributions. Related approaches to low–rank techniques include compressive sensing-based strategies [25],[26], which exploit sparsity to reduce the number of parameters for estimation, distributed dictionary learning [27], [28], [29], [30], which employs a bilinear dimensionality-reduction factorization scheme similar to some low-rank schemes but assumes no regression vectors, and attribute-distributed learning [31], which employs agents and a fusion center to meet communication constraints. Another important tool in recent related work is the principle of alternating optimization [32], [33], which consists of fixing a set of parameters, adjusting the remaining parameters and then proceeding in cycles [18], [21], [34], [35], [36], [37]. Ling and Ribeiro have studied dynamic decentralized optimization using the alternating direction method of multipliers [34]. Bai *et al.* have examined alternating optimization procedures to design sensing matrices and dictionaries for compressive sensing. Yan *et al.* have developed an alternating optimization for multigraph matching, whereas Magnusson *et al.* [37] have studied convergence of nonconvex optimization problems.

In this paper, we propose a scheme for distributed signal processing and distributed low–rank algorithms for parameter estimation. In particular, the proposed algorithms are based on an alternating optimization strategy [32], [33], [18], [21] and are called the distributed reduced-rank joint iterative optimization normalized least mean squares (DRJIO–NLMS) algorithm and the distributed reduced-rank joint iterative optimization recursive least squares (DRJIO–RLS) algorithm. In contrast to prior work on low-rank techniques [18]-[24] and distributed methods [8]-[13], distributed adaptive techniques based on the alternating optimization strategy are investigated. The proposed low-rank strategies are distributed and perform dimensionality reduction without costly decompositions at each agent. The proposed DRJIO–NLMS and DRJIO–RLS al-

S. Xu* is with the Communications Research Group, Department of Electronics, University of York, YO10 5DD York, U.K. (e-mail: songcen.xu@york.ac.uk).

R. C. de Lamare is with CETUC / PUC-Rio, Brazil and the Department of Electronics, University of York, U.K. (e-mail: rodrigo.delamare@york.ac.uk).

H. V. Poor is with the Department of Electrical Engineering, Princeton University, Princeton NJ 08544 USA (e-mail: poor@princeton.edu).

Part of this work has been presented at the 2014 European Signal Processing Conference, Lisbon, Portugal. EDICS:NEG-ADLE, NEG-ASAL, NEG-FUSE, SPE-DETC

gorithms are flexible with regards to the amount of information that is exchanged, have low cost and high performance. We also present a computational complexity analysis of the proposed and existing low-rank algorithms along with an analysis of the convergence of the proposed techniques. Applications to parameter estimation in wireless sensor networks and smart grids are then studied.

The main contributions of this work can be summarized as:

- Distributed low-rank adaptive algorithms based on alternating optimization .
- An analysis of the convergence and the computational complexity of the proposed distributed algorithms.
- A study of the proposed and existing distributed algorithms in wireless sensor networks and smart grids.

This paper is organized as follows: In Section II, the system model and the problem statement are described. In Section III, the proposed distributed dimensionality reduction and adaptive processing scheme is presented. Section IV details the proposed distributed low-rank algorithms. In Section V, an analysis of the convergence of the proposed algorithms is carried out along with a study of their computational complexity. Simulation results are presented and discussed in Section VI, whereas conclusions are drawn in Section VII.

## II. System Model and Problem Statement

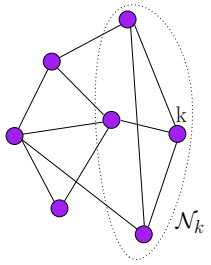


Fig. 1. Network topology with N nodes

A distributed network with N nodes, which have limited processing capabilities, is considered with a partially connected topology as illustrated in Fig. 1. A diffusion protocol in which nodes from the same neighborhood communicate with each other at every iteration is employed [2], although other strategies such as incremental [1] and consensus-based [4] could also be used. A partially connected network means that nodes can exchange information only with their neighbors determined by the connectivity topology. In contrast, a fully connected network means that, data broadcast by a node can be captured by all other nodes in the network [38]. At every time instant $i$, each node $k$ takes a scalar measurement $d_k(i)$ according to

$$d_k(i) = \boldsymbol{\omega}_0^H \boldsymbol{x}_k(i) + n_k(i), \quad i = 1, 2, \ldots, \mathrm{I}, \tag{1}$$

where $\boldsymbol{x}_k(i)$ is the $M \times 1$ input signal vector with zero mean and variance $\sigma_{x,k}^2$ that is also observed by node $k$, $n_k(i)$ is the noise sample measured at node $k$ which has zero mean and variance $\sigma_{n,k}^2$. Observing (1), we can see that the measurements for all nodes are related to an unknown parameter vector $\boldsymbol{\omega}_0$ with size $M \times 1$, that would be estimated by the network. The aim of such a network is to compute an estimate of $\boldsymbol{\omega}_0$ in a distributed fashion, which can minimize the global cost function

$$J(\boldsymbol{\omega}_k(i)) = \sum_{k=1}^{N} \mathbb{E}\big|d_k(i) - {\boldsymbol{\omega}_k}^H(i)\boldsymbol{x}_k(i)\big|^2, \tag{2}$$

where $\mathbb{E}$ denotes expected value and ${\boldsymbol{\omega}_k}^H(i)$ is the estimator at time $i$. To solve this problem, one suitable technique is the adapt–then–combine (ATC) diffusion strategy [2] described by

$$\begin{cases} \boldsymbol{\psi}_k(i) = \boldsymbol{\omega}_k(i-1) + \mu_k \boldsymbol{x}_k(i)\big[d_k(i) - \boldsymbol{\omega}_k^H(i-1)\boldsymbol{x}_k(i)\big]^*, \\ \boldsymbol{\omega}_k(i) = \sum\limits_{l \in \mathcal{N}_k} c_{kl}\boldsymbol{\psi}_l(i), \end{cases} \tag{3}$$

where $\mu_k$ is the step size, $\mathcal{N}_k$ indicates the set of neighbors for node $k$, $\boldsymbol{\psi}_k(i)$ is the local estimator, $|\mathcal{N}_k|$ denotes the cardinality of $\mathcal{N}_k$ and $c_{kl} > 0$ are the combination coefficients, which are calculated in this work using the Metropolis rule [2] given by

$$\begin{cases} c_{kl} = \frac{1}{max(|\mathcal{N}_k|,|\mathcal{N}_l|)}, & \text{if } k \neq l \text{ are linked} \\ c_{kl} = 0, & \text{for } k \text{ and } l \text{ not linked} \\ c_{kk} = 1 - \sum\limits_{l \in \mathcal{N}_k/k} c_{kl}, & \text{for } k = l \end{cases} \tag{4}$$

and should satisfy

$$\sum_{l} c_{kl} = 1, l \in \mathcal{N}_k \forall k. \tag{5}$$

Note that other combination rules can also be employed. With this adaptation strategy, when the dimension of the unknown parameter vector $\boldsymbol{\omega}_0$ is large, this could lead to a high communication overhead between each neighbor node and the learning speed of the network is reduced. In order to reduce the communication overhead, accelerate the learning and optimize the distributed processing, we incorporate at the $k$th node of the network distributed low-rank strategies based on alternating optimization techniques.

## III. Distributed Dimensionality Reduction and Adaptive Processing

The proposed distributed dimensionality reduction scheme, depicted in Fig.2, employs a transformation matrix $\boldsymbol{S}_{D_k}(i)$ to process the input signal $\boldsymbol{x}_k(i)$ with dimensions $M \times 1$ and projects it onto a lower $D \times 1$ dimensional subspace $\bar{\boldsymbol{x}}_k(i)$, where $D \ll M$. Following this procedure, a low-rank estimator $\bar{\boldsymbol{\omega}}_k(i)$ is computed, and the $\bar{\boldsymbol{\omega}}_k(i)$ is transmitted by each node. In particular, the transformation matrix $\boldsymbol{S}_{D_k}(i)$ and low-rank estimator $\bar{\boldsymbol{\omega}}_k(i)$ will be jointly optimized in the proposed scheme according to the mean squared error (MSE) criterion.

Specifically, we start the description of the method with an $M \times D$ matrix $\boldsymbol{S}_{D_k}(i)$, which carries out a dimensionality reduction on the input signal of each agent as given by

$$\bar{\boldsymbol{x}}_k(i) = \boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i), \tag{6}$$

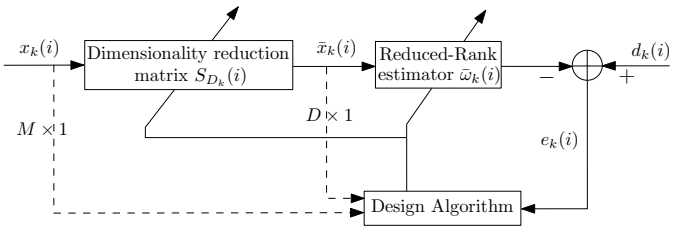


Fig. 2. Proposed dimensionality reduction scheme at each node or agent

where, in what follows, all $D$–dimensional quantities are designated with an overbar. The design of $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$ corresponds to the optimization problem given by

$$\begin{aligned}\left\{\boldsymbol{S}_{D_k}^{\text{opt}}, \bar{\boldsymbol{\omega}}_k^{\text{opt}}\right\} = \min_{\boldsymbol{S}_{D_k}(i), \bar{\boldsymbol{\omega}}_k(i)} \sum_{k=1}^{N} \mathbb{E}[|d_k(i) - \bar{\boldsymbol{\omega}}_k{}^H(i)\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)|^2] \\ + \delta E[||d_k^*(i)\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)||^2] \\ + \gamma E[\sum_{d=1}^{D} \boldsymbol{e}_d^H \boldsymbol{S}_{D_k}^H(i)\boldsymbol{I}_{M,D}\boldsymbol{e}_d]\end{aligned} \tag{7}$$

where $^*$ denotes complex conjugation, $\bar{\boldsymbol{\omega}}_k(i)$ is the low-rank estimator, the $M \times 1$ vectors $\boldsymbol{e}_d$ contain one in the $d$th entry and zeros elsewhere, the $M \times D$ matrix $\boldsymbol{I}_{M,D}$ contains a $D$-dimensional identity matrix on the top and zeros elsewhere, and the parameters $\gamma$ and $\delta$ are regularization terms that ensure the solution has rank $D$. In what follows, we describe the adaptation step that computes the parameters of $\bar{\boldsymbol{\omega}}_k(i)$ and $\boldsymbol{S}_{D_k}(i)$ based on an alternating minimization strategy, which consists of fixing one set of parameters and then minimizing the other.

By fixing $\bar{\boldsymbol{\omega}}_k(i)$ and minimizing (7) with respect to $\boldsymbol{S}_{D_k}(i)$, we arrive at the following expression:

$$\boldsymbol{S}_{D_k}(i) = \boldsymbol{R}_k^{-1}(i)\boldsymbol{P}_{D_k}(i)\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}^{-1}(i), \tag{8}$$

where the covariance matrix of the input signal vector at node $k$ $\boldsymbol{x}_k(i)$ is assumed to be full-rank and is given by

$$\boldsymbol{R}_k(i) = \mathbb{E}[\boldsymbol{x}_k(i)\boldsymbol{x}_k^H(i)], \tag{9}$$

the cross-correlation matrix is given by

$$\boldsymbol{P}_{D_k}(i) = \mathbb{E}[d_k^*(i)(\boldsymbol{x}_k(i)\bar{\boldsymbol{\omega}}_k^H(i) + \gamma \boldsymbol{I}_{M,D})] \tag{10}$$

and the covariance matrix of the reduced-rank parameter vector is described by

$$\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}(i) = \bar{\boldsymbol{\omega}}_k(i)\bar{\boldsymbol{\omega}}_k^H(i) + \delta \boldsymbol{I}_D. \tag{11}$$

We then fix $\boldsymbol{S}_{D_k}(i)$ and minimize (7) with respect to $\bar{\boldsymbol{\omega}}_k(i)$, which results in

$$\bar{\boldsymbol{\omega}}_k(i) = \bar{\boldsymbol{R}}_k^{-1}(i)\bar{\boldsymbol{p}}_k(i), \tag{12}$$

where the covariance matrix of the reduced-rank input signal vector that is also assumed to be full-rank is expressed by

$$\bar{\boldsymbol{R}}_k(i) = \mathbb{E}[\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)\boldsymbol{x}_k^H(i)\boldsymbol{S}_{D_k}(i)] = \mathbb{E}[\bar{\boldsymbol{x}}_k(i)\bar{\boldsymbol{x}}_k^H(i)] \tag{13}$$

and the cross-correlation vector is given by

$$\bar{\boldsymbol{p}}_k(i) = \mathbb{E}[d_k^*(i)\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)] = \mathbb{E}[d_k^*(i)\bar{\boldsymbol{x}}_k(i)]. \tag{14}$$

The reduced-dimension parameter vector $\bar{\boldsymbol{\omega}}_k(i)$ computed at each agent is then transmitted as a local low-rank estimator $\bar{\boldsymbol{\psi}}_k(i)$ to the other agents according to the network topology. At the receiver of each agent, there is a combination and reconstruction step in which the received data from neighboring nodes is combined to obtain a low-rank estimator:

$$\bar{\boldsymbol{\omega}}_k(i) = \sum_{l \in \mathcal{N}_k} c_{kl}\bar{\boldsymbol{\psi}}_l(i), \tag{15}$$

The full-dimension estimator is then obtained through a rank-$D$ approximation:

$$\boldsymbol{\omega}_k(i) = \boldsymbol{S}_{D_k}(i)\bar{\boldsymbol{\omega}}_k(i), \tag{16}$$

which is derived in the Appendix.

The associated low-rank MSE is obtained by substituting the expressions obtained in (12) and (8) into the cost function and is described by [18]

$$\text{MSE} = \sigma_{d_k}^2 - \bar{\boldsymbol{p}}_k^H(i)\bar{\boldsymbol{R}}_k^{-1}(i)\bar{\boldsymbol{p}}_k(i) \tag{17}$$

where $\sigma_{d_k}^2 = \mathbb{E}[|d_k(i)|^2]$. Because there is no closed-form expression for $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$ as they depend on each other, a strategy to compute the parameters is needed. The proposed strategy is based on an alternating optimization of $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$. The rank $D$ must be set by the designer to ensure appropriate performance taking into account the bias-variance tradeoff [15]. Furthermore, for the selection of $D$ the reader is referred to [39] for rank selection methods. In the next section, we develop a distributed low-rank algorithm to compute the parameters of interest.

## IV. Proposed Distributed Low-Rank Algorithms

In this section, we present the proposed distributed low-rank adaptive algorithms for distributed estimation, namely DRJIO–NLMS and DRJIO–RLS. Unlike prior work [11], [12], [13], the proposed algorithms do not require:

- Additional cost to perform eigen–decompositions [11]
- Extra adaptive processing at the local node [12]
- Multiple matrix-vector multiplications to to build the Krylov subspace [13]
- Costly convex optimization at the local node, which introduces extra complexity [13].

The objective of the DRJIO–NLMS and DRJIO–RLS algorithms is to perform compression/decompression and distributed parameter estimation subject to the constraint of transmitting only $D < M$ parameters. The algorithms are flexible, have low cost, very fast convergence speed and assume that the parameter $D$ is given. Alternatively, a model-order selection algorithm [16], [22], [40] can be employed to compute $D$ for each node. In particular, the algorithms rely on an alternating optimization strategy which consists of fixing a set of parameters $\boldsymbol{S}_{D_k}(i)$, updating the other set of parameters $\bar{\boldsymbol{\omega}}_k(i)$, then fixing $\bar{\boldsymbol{\omega}}_k(i)$ and updating $\boldsymbol{S}_{D_k}(i)$. This alternating approach with the recursions for $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$ is carried out in cycles until convergence is achieved.

### A. Proposed DRJIO–NLMS algorithm

In the DRJIO–NLMS algorithm, the parameters in (7) are optimized by an alternating procedure that adjusts one of the parameters while keeping the other parameter fixed using NLMS recursions. Therefore, the proposed DRJIO-NLMS algorithm solves the optimization problem in (7) in an alternating fashion. Using the method of steepest descent, computing the gradient terms of the cost function in (7) with respect to $\boldsymbol{S}_{D_k}(i)$, replacing the expected value with instantaneous estimates and considering the recursions in an alternating fashion, we arrive at the proposed DRJIO–NLMS algorithm:

$$\begin{aligned}\boldsymbol{S}_{D_k}(i) = \boldsymbol{S}_{D_k}(i-1) &+ \eta(i)e_k^*(i)\boldsymbol{x}_k(i)\bar{\boldsymbol{\omega}}_k^H(i-1) \\ &+ \eta(i)\big(\gamma d_k^*(i)\boldsymbol{I}_{M,D} - \delta\boldsymbol{x}_k(i)\boldsymbol{x}_k^H(i)\boldsymbol{S}_{D_k}(i-1)\big),\end{aligned} \tag{18}$$

$$\bar{\boldsymbol{\omega}}_k(i) = \bar{\boldsymbol{\omega}}_k(i-1) + \mu(i)e_k^*(i)\bar{\boldsymbol{x}}_k(i), \tag{19}$$

where $e_k(i) = d_k(i) - \bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{S}_{D_k}^H(i-1)\boldsymbol{x}_k(i)$, $\mu(i) = \frac{\mu_0}{\boldsymbol{x}_k^H(i)\boldsymbol{x}_k(i)}$ and $\eta(i) = \frac{\eta_0}{\bar{\boldsymbol{\omega}}_k^H(i-1)\bar{\boldsymbol{\omega}}_k(i-1)\boldsymbol{x}_k^H(i)\boldsymbol{x}_k(i)}$ are the time–varying step sizes. The normalization makes the setting of the convergence factors easier, improves the convergence speed and facilitates the comparison with other distributed LMS–type algorithms. The recursions are computed in an alternating way with one iteration per time instant at each node.

The proposed DRJIO–NLMS algorithm includes two steps, namely, adaptation step and combination and reconstruction step, which are performed using an alternating procedure which is detailed next.

- Adaptation step

For the adaptation step, at each time instant $i$=1,2, . . . , I, each node $k$=1,2, …, N, starts from generating a local low-rank estimator through

$$\bar{\boldsymbol{\psi}}_k(i) = \bar{\boldsymbol{\omega}}_k(i-1) + \mu(i)e_k^*(i)\bar{\boldsymbol{x}}_k(i), \tag{20}$$

where $e_k(i) = d_k(i) - \bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)$. This local low-rank estimator $\bar{\boldsymbol{\psi}}_k(i)$ will be transmitted to all its neighboring nodes under the network topology structure.

Then, each node $k$=1,2, …, N, will locally update its dimensionality reduction matrix according to (18) and keep it locally. Note that $\boldsymbol{S}_{D_k}(i)$ only employs the low-rank estimators from neighboring nodes and the local reference signal $d_k(i)$.

- Combination and reconstruction step

At each time instant $i$=1,2, . . . , I, the combination and reconstruction step starts after the adaptation step. Each node will combine the local low-rank estimators from its neighboring nodes and itself through

$$\bar{\boldsymbol{\omega}}_k(i) = \sum_{l\in\mathcal{N}_k} c_{kl}\bar{\boldsymbol{\psi}}_l(i), \tag{21}$$

to compute the low-rank estimator $\bar{\boldsymbol{\omega}}_k(i)$.

After the last iteration $I$, each node will reconstruct a full–dimensional estimator $\boldsymbol{\omega}_k(I)$ through the rank-$D$ approximation given by

$$\boldsymbol{\omega}_k(I) = \boldsymbol{S}_{D_k}(I)\bar{\boldsymbol{\omega}}_k(I). \tag{22}$$

In conclusion, during the distributed processing steps, only the local low-rank estimator $\bar{\boldsymbol{\psi}}_k(i)$ will be transmitted through the network. The proposed DRJIO–NLMS algorithm is detailed in Table I.

TABLE I
THE DRJIO–NLMS ALGORITHM

Initialize: $\bar{\boldsymbol{\omega}}_k(0) = \boldsymbol{0}$, $\boldsymbol{S}_{D_k}(0) = \boldsymbol{I}_{M,D}$
For each time instant $i$=1,2, . . . , I
  For each node $k$=1,2, …, N
    $\bar{\boldsymbol{\psi}}_k(i) = \bar{\boldsymbol{\omega}}_k(i-1) + \mu(i)e_k^*(i)\bar{\boldsymbol{x}}_k(i)$
    where $e_k(i) = d_k(i) - \bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{S}_{D_k}^H(i)\boldsymbol{x}_k(i)$
    % $\bar{\boldsymbol{\psi}}_k(i)$ is the local low-rank estimator and will be
    % sent to all neighboring nodes of node $k$ under the network
    % topology structure.
    $\boldsymbol{S}_{D_k}(i) = \boldsymbol{S}_{D_k}(i-1) + \eta(i)e_k^*(i)\boldsymbol{x}_k(i)\bar{\boldsymbol{\omega}}_k(i-1)$
      $+\eta(i)\big(\gamma d_k^*(i)\boldsymbol{I}_{M,D} - \delta\boldsymbol{x}_k(i)\boldsymbol{x}_k^H(i)\boldsymbol{S}_{D_k}(i-1)\big)$
    % The dimensionality reduction matrix $\boldsymbol{S}_{D_k}(i)$
    % will be updated and kept locally.
  end
  For each node $k$=1,2, …, N
    $\bar{\boldsymbol{\omega}}_k(i) = \sum_{l\in\mathcal{N}_k} c_{kl}\bar{\boldsymbol{\psi}}_l(i)$
    % The low-rank estimator $\bar{\boldsymbol{\omega}}_k(i)$
    % will be updated and kept locally.
  end
end
After the last iteration $I$
For each node $k$=1,2, …, N
  % Reconstruction.
  $\boldsymbol{\omega}_k(I) = \boldsymbol{S}_{D_k}(I)\bar{\boldsymbol{\omega}}_k(I)$
  where $\boldsymbol{\omega}_k(I)$ is the final full–rank estimator.
end

### B. Proposed DRJIO–RLS algorithm

In this subsection, we develop the DRJIO–RLS algorithm for computing $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$, which is inspired by the derivation of the standard recursive least squares (RLS) algorithm. The main differences are that we have two sets of recursions that update the parameters: one for $\boldsymbol{S}_{D_k}(i)$, which performs compression/decompression, and another for $\bar{\boldsymbol{\omega}}_k(i)$, which performs parameter estimation; and the recursions are distributed and computed in an alternating fashion. Therefore, we first fix $\bar{\boldsymbol{\omega}}_k(i)$ in the derivation and then derive a set of RLS recursions to compute the parameters for $\boldsymbol{S}_{D_k}(i)$. Subsequently, we fix $\boldsymbol{S}_{D_k}(i)$ and derive a set of RLS recursions to compute the parameters for $\bar{\boldsymbol{\omega}}_k(i)$. The DRJIO–RLS algorithm consists of an adaptation step, which computes $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$, and a combination and reconstruction step, which is identical to that of the DRJIO-NLMS algorithm.

- Adaptation step

In order to derive the proposed algorithm, we first define

$$\boldsymbol{P}_k(i) \triangleq \boldsymbol{R}_k^{-1}(i), \tag{23}$$

$$\boldsymbol{P}_{D_k}(i) \triangleq \lambda\boldsymbol{P}_{D_k}(i-1) + d_k^*(i)\boldsymbol{x}_k(i)\bar{\boldsymbol{\omega}}_k^H(i), \tag{24}$$

$$\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) \triangleq \bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}^{-1}(i-1), \tag{25}$$

and rewrite the expression in (8) as follows

$$\begin{aligned}\boldsymbol{S}_{D_k}(i) &= \boldsymbol{R}_k^{-1}(i)\boldsymbol{P}_{D_k}(i)\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}^{-1}(i-1) \\ &= \boldsymbol{P}_k(i)\boldsymbol{P}_{D_k}(i)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) \\ &= \lambda\boldsymbol{P}_k(i)\boldsymbol{P}_{D_k}(i-1)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) + d_k^*(i)\boldsymbol{P}_k(i)\boldsymbol{x}_k(i)\bar{\boldsymbol{\omega}}_k^H(i)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) \\ &= \boldsymbol{S}_{D_k}(i-1) + \boldsymbol{k}_k(i)\Big[d_k^*(i)\boldsymbol{t}_k^H(i) - \boldsymbol{x}_k^H(i)\boldsymbol{S}_{D_k}(i-1)\Big],\end{aligned} \tag{26}$$

where the $D \times 1$ vector $\boldsymbol{t}_k(i) = \boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i)\bar{\boldsymbol{\omega}}_k(i)$ and the $M \times 1$ Kalman gain vector is

$$\boldsymbol{k}_k(i) = \frac{\lambda^{-1}\boldsymbol{P}_k(i-1)\boldsymbol{x}_k(i)}{1+\lambda^{-1}\boldsymbol{x}_k^H(i)\boldsymbol{P}_k(i-1)\boldsymbol{x}_k(i)}. \tag{27}$$

In addition, the update for the $M \times M$ matrix $\boldsymbol{P}_k(i)$ employs the matrix inversion lemma [6] as follows:

$$\boldsymbol{P}_k(i) = \lambda^{-1}\boldsymbol{P}_k(i-1) - \lambda^{-1}\boldsymbol{k}_k(i)\boldsymbol{x}_k^H(i)\boldsymbol{P}_k(i-1) \tag{28}$$

and the $D \times 1$ vector $\boldsymbol{t}_k(i)$ is updated as

$$\boldsymbol{t}_k(i) = \frac{\lambda^{-1}\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1)\bar{\boldsymbol{\omega}}_k(i-1)}{1+\lambda^{-1}\bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1)\bar{\boldsymbol{\omega}}_k(i-1)}. \tag{29}$$

The matrix inversion lemma [6] is then used to update the $D \times D$ matrix $\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i)$ as described by

$$\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) = \lambda^{-1}\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1) - \lambda^{-1}\boldsymbol{t}_k(i)\bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1). \tag{30}$$

Equations (23)–(30) constitute the key steps of the proposed DRJIO-RLS algorithm for computing $\boldsymbol{S}_{D_k}(i)$.

To derive the expression for updating $\bar{\boldsymbol{\omega}}_k(i)$, the following associated quantities are defined

$$\bar{\boldsymbol{\Phi}}_k(i) \triangleq \bar{\boldsymbol{R}}_k^{-1}(i) \tag{31}$$

$$\bar{\boldsymbol{p}}_k(i) = \lambda\bar{\boldsymbol{p}}_k(i-1) + d_k^*(i)\boldsymbol{x}_k(i). \tag{32}$$

Then, equation (12) will be rewritten as

$$\begin{aligned}\bar{\boldsymbol{\omega}}_k(i) &= \bar{\boldsymbol{R}}_k^{-1}(i)\bar{\boldsymbol{p}}_k(i)\\ &= \bar{\boldsymbol{\Phi}}_k(i)\bar{\boldsymbol{p}}_k(i)\\ &= \lambda\bar{\boldsymbol{\Phi}}_k(i)\bar{\boldsymbol{p}}_k(i-1) + d_k^*(i)\bar{\boldsymbol{\Phi}}_k(i)\boldsymbol{x}_k(i)\\ &= \bar{\boldsymbol{\omega}}_k(i-1) + \bar{\boldsymbol{k}}_k(i)\Big[d_k^*(i) - \bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\omega}}_k(i-1)\Big],\end{aligned} \tag{33}$$

where the $D \times 1$ Kalman gain vector is given by

$$\bar{\boldsymbol{k}}_k(i) = \frac{\lambda^{-1}\bar{\boldsymbol{\Phi}}_k(i-1)\bar{\boldsymbol{x}}_k(i)}{1+\lambda^{-1}\bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\Phi}}_k(i-1)\bar{\boldsymbol{x}}_k(i)}. \tag{34}$$

and the update for the matrix inverse $\bar{\boldsymbol{\Phi}}_k(i)$ employs the matrix inversion lemma [6]

$$\bar{\boldsymbol{\Phi}}_k(i) = \lambda^{-1}\bar{\boldsymbol{\Phi}}_k(i-1) - \lambda^{-1}\bar{\boldsymbol{k}}_k(i)\bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\Phi}}_k(i-1). \tag{35}$$

Equations (31)–(35) are the key steps of the proposed DRJIO-RLS algorithm for computing $\bar{\boldsymbol{\omega}}_k(i)$. Since the combination and reconstruction step is identical to that of DRJIO-NLMS we omit it here. The proposed DRJIO–RLS algorithm is detailed in Table II.

## V. Analysis of the proposed algorithms

In this section, the computational complexity of the proposed algorithms is detailed and an analysis of sufficient conditions for convergence and a convergence proof to the optimal low-rank estimator are developed. Regarding our convergence proof, it is worth noting that the proof described in [32] was performed for a non-adaptive scenario, whereas the work in [33] was carried out for an adaptive setting. Our proof has been developed for the distributed adaptive case.

TABLE II
THE DRJIO-RLS ALGORITHM

Initialize: $\bar{\boldsymbol{\omega}}_k(0)$=0
$\boldsymbol{P}_k(0) = \delta^{-1}\boldsymbol{I}_{M\times M}$, $\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(0) = \delta^{-1}\boldsymbol{I}_{D\times D}$,
$\bar{\boldsymbol{\Phi}}_k(0) = \delta^{-1}\boldsymbol{I}_{D\times D}$ and $\delta$ = small positive constant
For each time instant $i$=1,2, . . . , I
For each node $k$=1,2, …, N
$\boldsymbol{k}_k(i) = \frac{\lambda^{-1}\boldsymbol{P}_k(i-1)\boldsymbol{x}_k(i)}{1+\lambda^{-1}\boldsymbol{x}_k^H(i)\boldsymbol{P}_k(i-1)\boldsymbol{x}_k(i)}$
$\boldsymbol{t}_k(i) = \frac{\lambda^{-1}\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1)\bar{\boldsymbol{\omega}}_k(i-1)}{1+\lambda^{-1}\bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1)\bar{\boldsymbol{\omega}}_k(i-1)}$
$\boldsymbol{S}_{D_k}(i) = \boldsymbol{S}_{D_k}(i-1) + \boldsymbol{k}_k(i)\Big[d_k^*(i)\boldsymbol{t}_k^H(i) - \boldsymbol{x}_k^H(i)\boldsymbol{S}_{D_k}(i-1)\Big]$
$\boldsymbol{P}_k(i) = \lambda^{-1}\boldsymbol{P}_k(i-1) - \lambda^{-1}\boldsymbol{k}_k(i)\boldsymbol{x}_k^H(i)\boldsymbol{P}_k(i-1)$
$\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i) = \lambda^{-1}\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1) - \lambda^{-1}\boldsymbol{t}_k(i)\bar{\boldsymbol{\omega}}_k^H(i-1)\boldsymbol{Q}_{\bar{\boldsymbol{\omega}}_k}(i-1)$
$\bar{\boldsymbol{k}}_k(i) = \frac{\lambda^{-1}\bar{\boldsymbol{\Phi}}_k(i-1)\bar{\boldsymbol{x}}_k(i)}{1+\lambda^{-1}\bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\Phi}}_k(i-1)\bar{\boldsymbol{x}}_k(i)}$
$\bar{\boldsymbol{\psi}}_k(i) = \bar{\boldsymbol{\omega}}_k(i-1) + \bar{\boldsymbol{k}}_k(i)\Big[d_k^*(i) - \bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\omega}}_k(i-1)\Big]$
$\bar{\boldsymbol{\Phi}}_k(i) = \lambda^{-1}\bar{\boldsymbol{\Phi}}_k(i-1) - \lambda^{-1}\bar{\boldsymbol{k}}_k(i)\bar{\boldsymbol{x}}_k^H(i)\bar{\boldsymbol{\Phi}}_k(i-1)$
end
For each node $k$=1,2, …, N
$\bar{\boldsymbol{\omega}}_k(i) = \sum_{l\in\mathcal{N}_k} c_{kl}\bar{\boldsymbol{\psi}}_l(i)$
end
end
For each node $k$=1,2, …, N
% Reconstruction.
$\boldsymbol{\omega}_k(I) = \boldsymbol{S}_{D_k}(I)\bar{\boldsymbol{\omega}}_k(I)$
where $\boldsymbol{\omega}_k(I)$ is the final full–rank estimator.
end

### A. Computational Complexity Analysis

Here, we evaluate the computational complexity of the proposed DRJIO–NLMS and DRJIO–RLS algorithms. The computational complexity of the proposed DRJIO–NLMS algorithm is $O(DM)$, while the proposed DRJIO–RLS algorithm has a complexity $O(M^2+D^2)$, where $O(\cdot)$ is used to classify algorithms according to how their requirements in arithmetic operations grow as the input size grows. The distributed NLMS algorithm [2] requires $O(M)$, while the complexity of the distributed RLS algorithm [41] is $O(M^2)$. For the Krylov Subspace NLMS [13] the complexity reaches $O(DM^2)$, while for the distributed principal subspace estimation algorithms [11], the complexity is $O(M^3)$. Thus, the proposed DRJIO–NLMS algorithm has a much lower computational complexity, and because we consider $D \ll M$, it has a comparable cost to the distributed NLMS algorithm [2]. The computational complexity of the model-order selection algorithm of [40] with extended filters requires $3(D_{\max} - D_{\min}) + 1$ additions and a sorting algorithm to find the best model order. An additional and very important aspect of distributed low-rank algorithms is that the dimensionality reduction results in a decrease in the number of transmitted parameters from $M$ to $D$, which corresponds to a less stringent bandwidth requirement.

The details of the computational complexity of the the proposed and the existing algorithms, are shown in Table III, where $M$ is the length of the unknown parameter that needs to be estimated, $D$ is the reduced dimension and $|\mathcal{N}_k|$ is the cardinality of $\mathcal{N}_k$. To further illustrate the computational complexity for different algorithms, we present the main trends in terms of the number of multiplications for the proposed and existing algorithms in Fig. 3. For the parameters, we consider

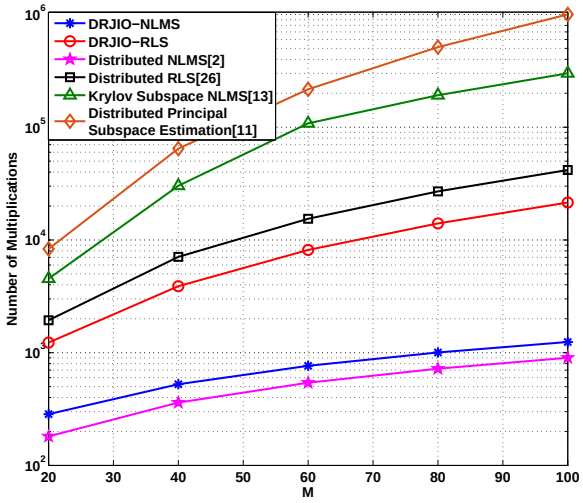


Fig. 3. Complexity in terms of multiplications

a network with $N = 20$ nodes, take node 14 as an example, and set $D = 5$ and $|\mathcal{N}_k| = 5$.

### B. Sufficient Conditions for Convergence

To start the analysis, we assume that the transformation matrix $\boldsymbol{S}_{D_k}(i)$ that performs compression and decompression/reconstruction and the low-rank parameter estimator $\boldsymbol{\omega}_k(i)$ aim to estimate the optimum pair $\boldsymbol{S}_{D_{k,\text{opt}}}$ and $\bar{\boldsymbol{\omega}}_{k,\text{opt}}(i)$ containing a common set of parameters of interest in the network. Then, to develop the analysis and proofs, we need to define a metric space and the Hausdorff distance that will extensively be used. A metric space is an ordered pair $(\mathcal{M}, r)$, where $\mathcal{M}$ is a nonempty set, and $r$ is a metric on $\mathcal{M}$, i.e., a function $r : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ such that, for any $x, y, z$, and $\mathcal{M}$, the following conditions hold.

1) $r(x, y) \geq 0$.
2) $r(x, y) = 0$ if $x = y$.
3) $r(x, y) = r(y, x)$.
4) $r(x, y) \leq r(x, y) + r(y, z)$ (triangle inequality).

The Hausdorff distance measures how far two subsets of a metric space are from each other and is defined by

$$r_H(X, Y) = \max\left\{ \sup_{x\in X} \inf_{y\in Y} r(x, y), \sup_{y\in Y} \inf_{x\in X} r(x, y) \right\}. \tag{36}$$

The proposed algorithms can be stated as alternating minimization strategies performed in a distributed fashion and expressed as

$$\boldsymbol{S}_{D_k}(i) \in \arg \min_{\boldsymbol{S}^{\text{opt}}_{D_k} \in \underline{\boldsymbol{S}}_{D_k}(i)} D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}_k(i)\right) \quad \text{for } k = 1, 2, \ldots, N \tag{37}$$

where $D_k(\cdot)$ is a distance metric and

$$\bar{\boldsymbol{\omega}}_k(i) \in \arg \min_{\bar{\boldsymbol{\omega}}^{\text{opt}}_k \in \underline{\bar{\boldsymbol{\omega}}}_k(i)} D_k\left(\boldsymbol{S}_{D_k}(i), \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right) \quad \text{for } k = 1, 2, \ldots, N \tag{38}$$

where $\boldsymbol{S}^{\text{opt}}_{D_k}$ and $\bar{\boldsymbol{\omega}}^{\text{opt}}_k$ correspond to the optimal values of $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$, respectively, and the sequences of compact sets $\{\underline{\boldsymbol{S}}_{D_k}(i)\}_{i\geq 0}$ and $\{\underline{\bar{\boldsymbol{\omega}}}_k(i)\}_{i\geq 0}$ converge to the sets $\underline{\boldsymbol{S}}_{D_k,\text{opt}}$ and $\underline{\bar{\boldsymbol{\omega}}}_{k,\text{opt}}$, respectively.

The sets $\underline{\boldsymbol{S}}_{D_k,\text{opt}}$ and $\underline{\bar{\boldsymbol{\omega}}}_{k,\text{opt}}$ are not directly given, but we observe the sequence of compact sets $\{\underline{\boldsymbol{S}}_{D_k}(i)\}_{i\geq 0}$ and $\{\underline{\bar{\boldsymbol{\omega}}}_k(i)\}_{i\geq 0}$. The goal of the proposed algorithms is to find a sequence of $\boldsymbol{S}_{D_k}(i)$ and $\bar{\boldsymbol{\omega}}_k(i)$ in a distributed way such that

$$\lim_{i\to\infty} D_k\left(\boldsymbol{S}_{D_k}(i), \bar{\boldsymbol{\omega}}_k(i)\right) = D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right). \tag{39}$$

To present a set of sufficient conditions under which the proposed algorithms converge, we employ the so–called three– and four–point properties [32], [33], [42], which are used in the study of the theory of convex sets. Let us assume that there is a function $f : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ such that the following conditions are satisfied.

1) *Three–point property* $(\boldsymbol{S}^{\text{opt}}_{D_k}, \tilde{\boldsymbol{S}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k)$. For all $i \geq 1$, $\boldsymbol{S}^{\text{opt}}_{D_k} \in \underline{\boldsymbol{S}}_{D_k}(i)$, $\bar{\boldsymbol{\omega}}^{\text{opt}}_k \in \underline{\bar{\boldsymbol{\omega}}}_k(i)$ and $\tilde{\boldsymbol{S}}_{D_k} \in \arg\min_{\bar{\boldsymbol{\omega}}^{\text{opt}}_k \in \underline{\bar{\boldsymbol{\omega}}}_k(i)} D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right)$, we have

$$f\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \tilde{\boldsymbol{S}}_{D_k}\right) + D_k\left(\tilde{\boldsymbol{S}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right) \leq D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right). \tag{40}$$

2) *Four–point property* $(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k, \tilde{\boldsymbol{S}}_{D_k}, \tilde{\bar{\boldsymbol{\omega}}}_k)$. For all $i \geq 1$, $\boldsymbol{S}^{\text{opt}}_{D_k}, \tilde{\boldsymbol{S}}_{D_k} \in \underline{\boldsymbol{S}}_{D_k}(i)$, $\bar{\boldsymbol{\omega}}^{\text{opt}}_k \in \underline{\bar{\boldsymbol{\omega}}}_k(i)$ and $\tilde{\bar{\boldsymbol{\omega}}}_k \in \arg\min_{\bar{\boldsymbol{\omega}}^{\text{opt}}_k \in \underline{\bar{\boldsymbol{\omega}}}_k(i)} D_k\left(\tilde{\boldsymbol{S}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right)$, we have

$$D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \tilde{\bar{\boldsymbol{\omega}}}_k\right) \leq D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right) + f\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \tilde{\boldsymbol{S}}_{D_k}\right). \tag{41}$$

*Theorem*: Let $\{\underline{\boldsymbol{S}}_{D_k}(i)\}_{i\geq 0}$, $\{\underline{\bar{\boldsymbol{\omega}}}_k(i)\}_{i\geq 0}$, $\underline{\boldsymbol{S}}^{\text{opt}}_{D_k}$, $\underline{\bar{\boldsymbol{\omega}}}^{\text{opt}}_k$ be compact subsets of the compact metric space $(\mathcal{M}, r)$ such that

$$\underline{\boldsymbol{S}}_{D_k}(i) \overset{r_h}{\to} \underline{\boldsymbol{S}}^{\text{opt}}_{D_k} \qquad \underline{\bar{\boldsymbol{\omega}}}_k(i) \overset{r_h}{\to} \underline{\bar{\boldsymbol{\omega}}}^{\text{opt}}_k \tag{42}$$

and let $D_k : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ be a continuous function.

Now, let conditions 1) and 2) hold. Then, for the proposed algorithms, we have

$$\lim_{i\to\infty} D_k\left(\boldsymbol{S}_{D_k}(i), \bar{\boldsymbol{\omega}}_k(i)\right) = D_k\left(\boldsymbol{S}^{\text{opt}}_{D_k}, \bar{\boldsymbol{\omega}}^{\text{opt}}_k\right). \tag{43}$$

A general proof of this theorem is detailed in [32], [33].

### C. Convergence to the Optimal Low-Rank Estimator

In this section, we show that the proposed low–rank algorithm globally and exponentially converges to the optimal low-rank estimator [15], [43]. This result is applicable to least-squares type algorithms with forgetting factor $\lambda = 1$. We remark that for stochastic gradient (or LMS) algorithms and least-squares algorithms with forgetting factor $\lambda \neq 1$, there will be a misadjustment or loss in MSE due to the adaptation with the step size when an LMS algorithms is adopted, or due to the forgetting factor when an RLS algorithm is chosen. To

TABLE III
COMPUTATIONAL COMPLEXITY OF DIFFERENT ALGORITHMS

| Algorithm | Multiplications | Additions |
|---|---|---|
| DRJIO–NLMS | $2(D+1)M+(3+\lvert\mathcal{N}_k\rvert)D+5$ | $(2D+1)M+(2+\lvert\mathcal{N}_k\rvert)D-2$ |
| DRJIO–RLS | $2M^2+(3+2D)M+4D^2$ $+(9+\lvert\mathcal{N}_k\rvert)D$ | $2M^2+2DM+4D^2$ $+(2+\lvert\mathcal{N}_k\rvert)D$ |
| Distributed NLMS [2] | $(4+\lvert\mathcal{N}_k\rvert)M+1$ | $(5+\lvert\mathcal{N}_k\rvert)M-1$ |
| Distributed RLS [41] | $4M^2+(12+\lvert\mathcal{N}_k\rvert)M-1$ | $4M^2+(16+\lvert\mathcal{N}_k\rvert)M+1$ |
| Krylov Subspace NLMS [13] | $6DM^2+4M+(5+\lvert\mathcal{N}_k\rvert)D$ | $6DM^2+2M+(2+\lvert\mathcal{N}_k\rvert)D$ |
| Distributed principal subspace estimation [11] | $M^3+2(D+2)M$ $+(3+\lvert\mathcal{N}_k\rvert)D+4$ | $M^3+(D+1)M$ $+(2+\lvert\mathcal{N}_k\rvert)D-1$ |

proceed with our proof, let us rewrite the expressions in (8) and (10 for time instant zero as follows:

$$\boldsymbol{R}_k(0)\boldsymbol{S}_{D_k}(0)\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}(0)=\boldsymbol{P}_{D_k}(0)=\boldsymbol{p}_k(0)\bar{\boldsymbol{\omega}}_k^H(0)+\delta\boldsymbol{\Upsilon}, \tag{44}$$

where $\boldsymbol{\Upsilon}=\begin{bmatrix}\boldsymbol{I}_D\\ \boldsymbol{0}_{M-D\times D}\end{bmatrix}$, $\boldsymbol{\Upsilon}$ is an $M\times D$ matrix containing an identity matrix $\boldsymbol{I}_D$ with size $D$ and an $M-D\times D$ matrix with zeros $\boldsymbol{0}_{M-D\times D}$, $\delta$ is a small positive scalar used to regularize the recursion at initialization and ensure that a rank-$D$ matrix $\boldsymbol{P}_{D_k}(0)$ is obtained, and the $D$-dimensional set of normal equations that must be solved to compute $\bar{\boldsymbol{\omega}}_k(1)$ is given by

$$\begin{aligned}\bar{\boldsymbol{R}}_k(0)\bar{\boldsymbol{\omega}}_k(1)&=\boldsymbol{S}_{D_k}^H(0)\boldsymbol{R}_k(0)\boldsymbol{S}_{D_k}(0)\bar{\boldsymbol{\omega}}_k(1)\\&=\boldsymbol{S}_{D_k}(0)\boldsymbol{p}_k(0)=\bar{\boldsymbol{p}}_k(0),\end{aligned} \tag{45}$$

where $\boldsymbol{p}_k(i)=E[d_k(i)^*\boldsymbol{x}_k(i)$ is the cross-correlation vector.

Using (44), we can obtain the following relation

$$\begin{aligned}\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}(0)=&\left(\boldsymbol{S}_{D_k}^H(0)\boldsymbol{R}_k^2(0)\boldsymbol{S}_{D_k}(0)\right)^{-1}\\&\times\boldsymbol{S}_{D_k}(0)\boldsymbol{R}_k(0)\boldsymbol{P}_{D_k}(0).\end{aligned} \tag{46}$$

Substituting the aforementioned result for $\bar{\boldsymbol{R}}_{\bar{\boldsymbol{\omega}}_k}(0)$ into the expression in (44), we get a recursive expression for $\boldsymbol{S}_{D_k}(0)$ as

$$\begin{aligned}\boldsymbol{S}_{D_k}(0)=&\boldsymbol{R}_k^{-1}(0)\boldsymbol{P}_{D_k}(0)\left(\boldsymbol{S}_{D_k}^H(0)\boldsymbol{R}_k(0)\boldsymbol{P}_{D_k}(0)\right)^{-1}\\&\times\left(\boldsymbol{S}_{D_k}^H(0)\boldsymbol{R}_k^2(0)\boldsymbol{S}_{D_k}(0)\right)^{-1}.\end{aligned} \tag{47}$$

Using (44), we can express $\bar{\boldsymbol{\omega}}_k(1)$ as

$$\bar{\boldsymbol{\omega}}_k(1)=\left(\boldsymbol{S}_{D_k}^H(0)\boldsymbol{R}_k(0)\boldsymbol{S}_{D_k}(0)\right)^{-1}\boldsymbol{S}_{D_k}^H(0)\boldsymbol{p}_k(0). \tag{48}$$

For the proposed DRJIO–NLMS and DRJIO–RLS, the relation is given by

$$\boldsymbol{\omega}_k(1)=\boldsymbol{S}_{D_k}(1)\sum_{l\in\mathcal{N}_k}c_{kl}\bar{\boldsymbol{\omega}}_l(1). \tag{49}$$

Substituting $\boldsymbol{S}_{D_k}(1)$ and $\bar{\boldsymbol{\omega}}_l(1)$ into (49), we obtain

$$\begin{aligned}\boldsymbol{\omega}_k(1)=&\boldsymbol{R}_k^{-1}(1)\boldsymbol{P}_{D_k}(1)\left(\boldsymbol{S}_{D_k}^H(1)\boldsymbol{R}_k(1)\boldsymbol{P}_{D_k}(1)\right)^{-1}\\&\times\left(\boldsymbol{S}_{D_k}^H(1)\boldsymbol{R}_k^2(1)\boldsymbol{S}_{D_k}(1)\right)^{-1}\\&\times\sum_{l\in\mathcal{N}_k}c_{kl}\left(\boldsymbol{S}_{D_l}^H(0)\boldsymbol{R}_l(0)\boldsymbol{S}_{D_l}(0)\right)^{-1}\boldsymbol{S}_{D_l}^H(0)\boldsymbol{p}_l(0).\end{aligned} \tag{50}$$

More generally, we can express the proposed distributed algorithms by the following recursion:

$$\begin{aligned}\boldsymbol{\omega}_k(i)&=\boldsymbol{S}_{D_k}(i)\sum_{l\in\mathcal{N}_k}c_{kl}\bar{\boldsymbol{\omega}}_l(i)\\&=\boldsymbol{R}_k^{-1}(i)\boldsymbol{P}_{D_k}(i)\left(\boldsymbol{S}_{D_k}^H(i)\boldsymbol{R}_k(i)\boldsymbol{P}_{D_k}(i)\right)^{-1}\\&\quad\times\left(\boldsymbol{S}_{D_k}^H(i)\boldsymbol{R}_k^2(i)\boldsymbol{S}_{D_k}(i)\right)^{-1}\\&\quad\times\sum_{l\in\mathcal{N}_k}c_{kl}\left(\boldsymbol{S}_{D_l}^H(i-1)\boldsymbol{R}_l(i-1)\boldsymbol{S}_{D_l}(i-1)\right)^{-1}\\&\quad\times\boldsymbol{S}_{D_l}^H(i-1)\boldsymbol{p}_l(i-1).\end{aligned} \tag{51}$$

At this point, we resort to the assumption that the matrices $\boldsymbol{S}_{D_k}(i)$ for each node $k$ must converge to the same values, which correspond to the optimal transformation matrix. Because the optimal low-rank filter can be described by the eigenvalue decomposition of $\boldsymbol{R}_k^{-1/2}(i)\boldsymbol{p}_k(i)$ [20], [21], where $\boldsymbol{R}_k^{-1/2}(i)$ is the square root of the matrix $\boldsymbol{R}_k(i)$, and $\boldsymbol{p}_k(i)$ is the cross-correlation vector, we have

$$\boldsymbol{R}_k^{-1/2}(i)\boldsymbol{p}_k(i)=\boldsymbol{\Phi}_k\boldsymbol{\Lambda}_k\boldsymbol{\Phi}_k^H\boldsymbol{p}_k(i), \tag{52}$$

where $\boldsymbol{\Lambda}_k$ is an $M\times M$ diagonal matrix with the eigenvalues of $\boldsymbol{R}_k$, and $\boldsymbol{\Phi}_k$ is a $M\times M$ unitary matrix with the eigenvectors of $\boldsymbol{R}_k$.

Let us also assume that there exists some $\boldsymbol{\omega}_k(0)$ such that the randomly selected $\boldsymbol{S}_{D_k}(0)$ can be written as [21]

$$\boldsymbol{S}_{D_k}(0)=\boldsymbol{R}_k^{-1/2}(i)\boldsymbol{\Phi}_k\boldsymbol{\omega}_k(0), \tag{53}$$

Using (52) and (53) in (51) together with the assumption and some manipulation of the algebraic expressions, we can express (51) in a more compact way that is suitable for analysis, as given by

$$\begin{aligned}\boldsymbol{\omega}_k(i)=&\sum_{l\in\mathcal{N}_k}c_{kl}\boldsymbol{\Lambda}_l^2\boldsymbol{\omega}_l(i-1)\left(\boldsymbol{\omega}_l^H(i-1)\boldsymbol{\Lambda}_l^2\boldsymbol{\omega}_l(i-1)\right)^{-1}\\&\times\boldsymbol{\omega}_l^H(i-1)\boldsymbol{\omega}_l(i-1).\end{aligned} \tag{54}$$

The aforementioned expression can be decomposed as follows:

$$\boldsymbol{\omega}_k(i)=\sum_{l\in\mathcal{N}_k}c_{kl}\boldsymbol{Q}_l(i)\boldsymbol{Q}_l(i-1)\ldots\boldsymbol{Q}_l(1)\boldsymbol{\omega}_l(0), \tag{55}$$

where

$$\boldsymbol{Q}_l(i) = \boldsymbol{\Lambda}_l^{2i}\boldsymbol{\omega}_l(0)\big(\boldsymbol{\omega}_l^H(0)\boldsymbol{\Lambda}_l^{4i-2}\boldsymbol{\omega}_l(0)\big)^{-1}\boldsymbol{\omega}_l^H(0)\boldsymbol{\Lambda}_l^{2i-2}. \tag{56}$$

At this point, we need to establish that the norm of $\boldsymbol{S}_{D_k}(i)$, for all $i$, is both lower and upper bounded, i.e., $0 <\| \boldsymbol{S}_{D_k}(i) \|< \infty$, for all $i$, and that $\boldsymbol{\omega}_k(i) = \boldsymbol{S}_{D_k}(i)\sum_{l\in\mathcal{N}_k} c_{kl}\bar{\boldsymbol{\omega}}_l(i)$ exponentially approaches $\boldsymbol{\omega}_{k,\text{opt}}(i)$ as $i$ increases. Due to the linear mapping, the boundedness of $\boldsymbol{S}_{D_k}(i)$ is equivalent to the boundedness of $\boldsymbol{\omega}_k(i)$. Therefore, we have upon convergence when $i \to \infty$ that

$$\boldsymbol{\omega}_k^H(i)\boldsymbol{\omega}_k(i-1) = \boldsymbol{\omega}_k^H(i-1)\boldsymbol{\omega}_k(i-1). \tag{57}$$

Because $\| \boldsymbol{\omega}_k^H(i)\boldsymbol{\omega}_k(i-1) \|\leq\| \boldsymbol{\omega}_k^H(i-1) \|\| \boldsymbol{\omega}_k(i) \|$ and $\| \boldsymbol{\omega}_k^H(i-1)\boldsymbol{\omega}_k(i-1) \|=\| \boldsymbol{\omega}_k(i-1) \|^2$, the relation $\boldsymbol{\omega}_k^H(i)\boldsymbol{\omega}_k(i-1) = \boldsymbol{\omega}_k^H(i-1)\boldsymbol{\omega}_k(i-1)$ implies that $\| \boldsymbol{\omega}_k(i) \|\geq\| \boldsymbol{\omega}_k(i-1) \|$, and hence we have

$$\| \boldsymbol{\omega}_k(\infty) \|\geq\| \boldsymbol{\omega}_k(i) \|\geq\| \boldsymbol{\omega}_k(0) \| . \tag{58}$$

To show that the upper bound $\| \boldsymbol{\omega}_k(\infty) \|$ is finite, let us express the $M \times M$ matrix $\boldsymbol{Q}_k(i)$ as a function of the $M \times 1$ vector $\boldsymbol{\omega}_l(i) = \begin{bmatrix} \boldsymbol{\omega}_{l,1}(i) \\ \boldsymbol{\omega}_{l,2}(i) \end{bmatrix}$ and the $M \times M$ matrix $\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_{l,1} & \\ & \boldsymbol{\Lambda}_{l,2} \end{bmatrix}$. Substituting the previous expressions of $\boldsymbol{\omega}_l(i)$ and $\boldsymbol{\Lambda}_k l$ into $\boldsymbol{Q}_l(i)$ as given in (56), we obtain

$$\begin{aligned}\boldsymbol{Q}_l(i) = &\begin{bmatrix} \boldsymbol{\Lambda}_{l,1}^{2i}\boldsymbol{\omega}_{l,1}(0) \\ \boldsymbol{\Lambda}_{l,2}^{2i}\boldsymbol{\omega}_{l,2}(0) \end{bmatrix}\big(\boldsymbol{\omega}_{l,1}^H(0)\boldsymbol{\Lambda}_{l,1}^{4i-2}\boldsymbol{\omega}_{l,1}(0) \\ &+ \boldsymbol{\omega}_{l,2}^H(0)\boldsymbol{\Lambda}_{l,2}^{4i-2}\boldsymbol{\omega}_{l,2}(0)\big)^{-1}\begin{bmatrix} \boldsymbol{\omega}_{l,1}^H(0)\boldsymbol{\Lambda}_{l,1}^{2i-2} \\ \boldsymbol{\omega}_{l,2}^H(0)\boldsymbol{\Lambda}_{l,2}^{2i-2} \end{bmatrix}\end{aligned} \tag{59}$$

Using the matrix identity $(\boldsymbol{A}+\boldsymbol{B})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{I} + \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{A}^{-1}$ in the decomposed $\boldsymbol{Q}_l(i)$ in (59) and making $i$ large, we get

$$\boldsymbol{Q}_l(i) = \text{diag}(\underbrace{1\ldots1}_{D}\underbrace{0\ldots0}_{M-D}) + O_l\big(\epsilon_l(i)\big), \tag{60}$$

where $\epsilon_l(i) = (\sigma_{r+1}/\sigma_r)^{2i}$, in which $\sigma_{r+1}$ and $\sigma_r$ are the $(r+1)$th and the $r$th largest singular values of $\boldsymbol{R}_l^{-1/2}(i)\boldsymbol{p}_l(i)$, respectively, and $O(\cdot)$ denotes the order of the argument. Based on (60), it follows that, for some positive constant $g$, we have $\| \boldsymbol{\omega}_l(i) \|\leq 1 + g\epsilon_l(i)$. Based on (55), we obtain

$$\begin{aligned}\| \boldsymbol{\omega}_k(\infty) \| &\leq \sum_{l\in\mathcal{N}_k} c_{kl}(\| \boldsymbol{Q}_l(\infty) \| \ldots \| \boldsymbol{Q}_l(1) \|\| \boldsymbol{Q}_l(0) \|) \\ &\leq \sum_{l\in\mathcal{N}_k} c_{kl}\Bigg( \| \boldsymbol{\omega}_l(0) \| \prod_{i=0}^{\infty}\big(1+g\epsilon_l(i)\big)\Bigg) \\ &= \sum_{l\in\mathcal{N}_k} c_{kl}\Bigg( \| \boldsymbol{\omega}_l(0) \| \exp\Bigg(\sum_{i=1}^{\infty}\log\big(1+g\epsilon_l(i)\big)\Bigg)\Bigg) \\ &\leq \sum_{l\in\mathcal{N}_k} c_{kl}\Bigg( \| \boldsymbol{\omega}_l(0) \| \exp\Bigg(\sum_{i=1}^{\infty} g\epsilon_l(i)\Bigg)\Bigg) \\ &=\| \sum_{l\in\mathcal{N}_k} c_{kl}\Bigg( \| \boldsymbol{\omega}_l(0) \| \exp\Bigg(\frac{g}{1-(\sigma_{r+1}/\sigma_r)^2}\Bigg)\Bigg).\end{aligned} \tag{61}$$

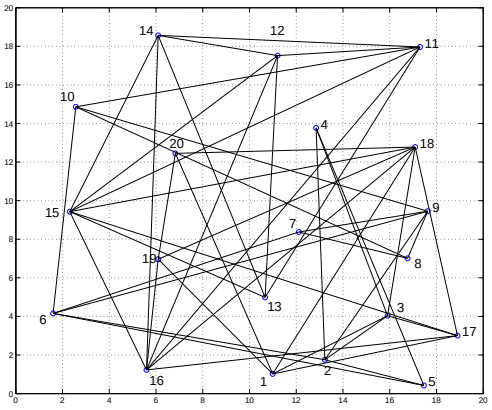


Fig. 4. Network topology with $N = 20$ nodes

With the previous development, the norm of $\boldsymbol{\omega}_k(i)$ is proven to be both lower and upper bounded. Once this case has been established, the expression in (51) converges for a sufficiently large $i$ to the low-rank Wiener filter. This condition is verified by equating the terms of (54), which yields

$$\boldsymbol{\omega}_k(i) = \sum_{l\in\mathcal{N}_k} c_{kl}\Bigg(\boldsymbol{R}_l^{-1/2}(i)\boldsymbol{\Phi}_{l,1}\boldsymbol{\Lambda}_{l,1}\boldsymbol{\Phi}_{l,1}^H\boldsymbol{p}_l(i) + O_l\big(\epsilon_l(i)\big)\Bigg) \tag{62}$$

where $\boldsymbol{\Phi}_{l,1}$ is a $M \times D$ matrix with the $D$ largest eigenvectors of $\boldsymbol{R}_l(i)$, and $\boldsymbol{\Lambda}_{l,1}$ is a $D \times D$ matrix with the largest eigenvalues of $\boldsymbol{R}_l(i)$.

## VI. Simulation results

In this section, we investigate the performance of the proposed DRJIO–NLMS and DRJIO–RLS algorithms for distributed estimation in two scenarios: wireless sensor networks and smart grids.

### A. Wireless Sensor Networks

In this subsection, we compare the proposed DRJIO–NLMS and DRJIO–RLS algorithms with the distributed NLMS algorithm (normalized version of [2]), distributed RLS algorithm [41], Krylov subspace NLMS [13] and distributed principal subspace estimation [11], based on their MSE performance.

With the network topology structure outlined in Fig. 4 with $N = 20$ nodes, we consider numerical simulations under three scenarios for the parameter vector $\omega_o$:

- Full–rank system with $M$=20
- Sparse system with $M$=20 ($D$ non-zero coefficients and $M - D$ zeros coefficients)
- Full–rank system with $M$=60

The input signal is generated as $\boldsymbol{x}_k(i) = [x_k(i) \quad x_k(i-1) \quad \ldots \quad x_k(i-M+1)]$ and $x_k(i) = u_k(i) + \alpha_k x_k(i-1)$, where $\alpha_k$ is a correlation coefficient and $u_k(i)$ is a white noise process with variance $\sigma_{u,k}^2 = 1-|\alpha_k|^2$, to ensure the variance of $\boldsymbol{x}_k(i)$ is $\sigma_{x,k}^2 = 1$. In particular, this application requires the estimation of a set of parameters that could be modeled

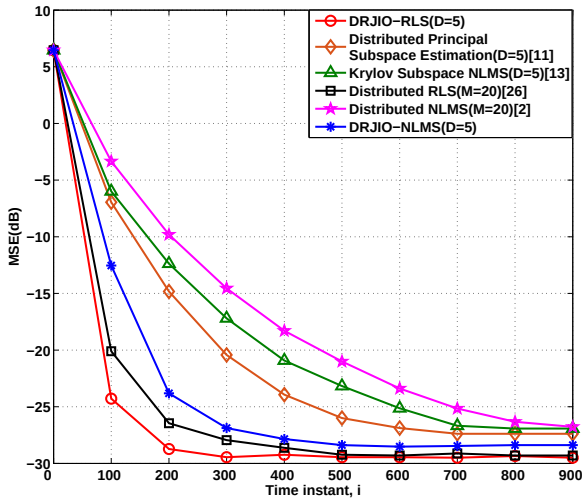


Fig. 5. Full–rank system with $M$=20

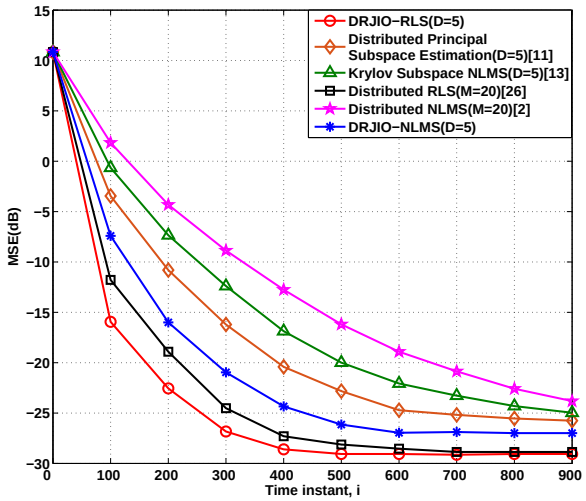


Fig. 6. Full–rank system with $M$=60

as a finite-impulse response (FIR) filter (related to a moving average (MA) model). Furthermore, the algorithms would also work with an input generated by an auto-regressive (AR) model but their performance would depend on the condition number and the rank of the correlation matrix of the input data. The noise samples are modeled as complex Gaussian noise with variance $\sigma^2_{n,k} = 0.001$. We have adopted the regularization parameters $\gamma = 0.02$ and $\delta = 0.01$ in all examples. We have also evaluated the impact of different values of regularization parameters and the results indicate that the performance of the algorithms degrades when the parameters are not well chosen. Moreover, the optimized values work very well for a wide range of scenarios and values of noise variance. We assume that the network has error–free transmission between linked nodes.

The step size $\mu_0$ for the distributed NLMS algorithm, Krylov subspace NLMS, distributed principal subspace estimation and DRJIO–NLMS is set to 0.15 and $\eta_0$ is set to 0.5. For the distributed RLS algorithm and DRJIO–RLS algorithm, the forgetting factor $\lambda$ is equal to 0.99 and $\delta$ is set to 0.11. In Fig. 5, we compare the proposed DRJIO–NLMS and DRJIO–RLS algorithms with the existing strategies using the full–rank system with $M$=20 and $D$=5. The dimensionality reduction matrix $\boldsymbol{S}_{D_k}(0)$ is initialized as $[\boldsymbol{I}_D \;\; \boldsymbol{0}_{D,M-D}]^T$.

We observe that the proposed DRJIO–RLS algorithm has the best performance when compared with other algorithms, while the proposed DRJIO–NLMS algorithm also has a better performance, which is very close to the distributed RLS algorithm. The superior performance of DRJIO-RLS can be explained by the fact that the convergence rate or learning speed of adaptive algorithms depends on the number of parameters that need to be estimated. This is well known in adaptive signal processing [6]. For instance, when we compare the DRJIO-RLS and the full-rank RLS, the difference is that the proposed DRJIO-RLS estimates the unknown parameters using a reduced dimension and retaining the most relevant features of the data. As a result, the DRJIO-RLS converges faster than the standard RLS algorithm. However, its complexity is an order of magnitude lower than those of the distributed RLS algorithm and the DRJIO–RLS algorithm.

When the full–rank system $M$ increases to 60, Fig. 6 illustrates that, the proposed DRJIO–RLS algorithm still has the best performance, while DRJIO–NLMS algorithm also shows a fast convergence rate, which is comparable to the distributed RLS algorithm. For the distributed NLMS, Krylov subspace NLMS and distributed principal subspace estimation algorithms, their convergence speed is much lower.

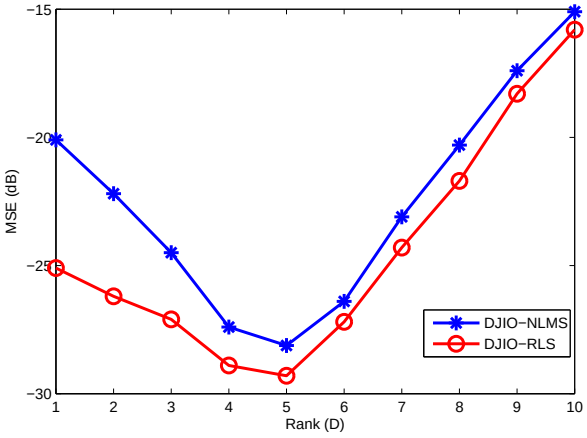


Fig. 7. MSE performance versus rank $D$ for a sparse system with $M$=100

In a sparse system scenario with $M = 100$, we first evaluate the MSE performance versus the rank $D$ and then we assess the MSE performance versus the number of iterations, as shown in Figs. 7 and 8, respectively. In particular, the curves illustrating the MSE performance versus the rank $D$ are obtained after 500 iterations for a range of $D$ between 1 and 10. The results depicted in Fig. 7 indicate that the

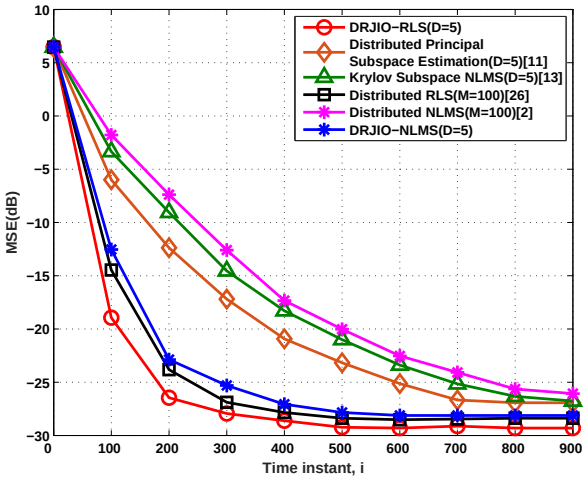


Fig. 8. Sparse system with $M$=100

best rank for both DJRIO–NLMS and DJRIO–RLS algorithms corresponds to $D = 5$ and that the MSE performance gradually degrades for other values. The rank $D$ should be carefully selected as it affects the performance of these algorithms and determines the number of parameters that should be exchanged between nodes. Moreover, we have considered $D = 5$ for assessing the MSE performance versus the number of iterations for the proposed and other existing algorithms, as shown in Fig. 8. The results indicate that the proposed DRJIO–RLS and DRJIO–NLMS algorithms have a more pronounced performance advantage over the distributed NLMS, the Krylov subspace NLMS and the distributed principal subspace estimation algorithms. Specifically, the proposed DRJIO–NLMS algorithm performs very close to the distributed RLS algorithm and outperforms the other analyzed algorithms.

In the last example on wireless sensor networks, we compare the performance between the proposed DRJIO–NLMS and the DCE scheme in [26], under different sparsity level scenarios. The step size for both algorithms is set to 0.3 and the $\eta_0$ for DRJIO–NLMS is set to 0.5. The length of the unknown parameter $\boldsymbol{\omega}_0$ is 20 and $D = 10$. For the first scenario, the number of non–zero coefficients in the unknown parameter is 3 and for the second scenario, the number of non–zero coefficients is set to 10. The comparison results are shown in Fig. 9 and 10. It is clear that in a very sparse system, the proposed DCE scheme outperforms the DRJIO–NLMS algorithm. With the decrease of the system sparsity level, the proposed DRJIO–NLMS algorithm outperforms the DCE scheme. The results of 9 and 10 indicate that the proposed DRJIO–NLMS algorithm is superior to the DCE scheme when the level of sparsity is not very high. The computational complexity of DRJIO-NLMS is much lower than the DCE scheme because the latter requires a basis pursuit algorithm to reconstruct the full-dimension estimator and DRJIO-NLMS employs a rank-$D$ approximation based on a simple matrix-vector multiplication.

### B. Smart Grids

In order to test the proposed algorithms in a possible smart grid scenario, we consider the Hierarchical IEEE 14–bus system which has been proposed in [44], where 14 is the number of substations. At every time instant $i$, each bus $k, k = 1, 2, \ldots, 14$, takes a scalar measurement $d_k(i)$ according to

$$d_k(i) = X_k\big(\boldsymbol{\omega}_0(i)\big) + n_k(i), \quad k = 1, 2, \ldots, 14, \tag{63}$$

where $\boldsymbol{\omega}_0(i)$ is the state vector of the entire interconnected system, $X_k(\boldsymbol{\omega}_0(i))$ is a nonlinear measurement function of bus $k$. The quantity $n_k(i)$ is the measurement error with mean equal to zero and which corresponds to bus $k$.

We focus on the linearized DC state estimation problem. We assume that each bus connects and measures the state of three users. As a result, for the IEEE–14 bus system, there will be 42 users in the system. The system is built with 1.0 per unit (p.u) voltage magnitudes at all users and j1.0 p.u. branch impedance. Then, the state vector $\boldsymbol{\omega}_0(i)$ is taken as the voltage phase angle vector $\boldsymbol{\omega}_0$ for all users. Initially, each bus only knows the voltage phase angle of the three users connected to it. With the help of distributed estimation algorithms, each bus is supposed to estimate the state of the voltage phase angles for all users in the system. Therefore, the nonlinear measurement model for state estimation (63) is approximated by

$$d_k(i) = \boldsymbol{x}_k^H(i)\boldsymbol{\omega}_0 + n_k(i), \quad k = 1, 2, \ldots, 14, \tag{64}$$

where $\boldsymbol{x}_k(i)$ is the measurement Jacobian vector for bus $k$. Then, the aim of the distributed estimation algorithm is to compute an estimate of $\boldsymbol{\omega}_0$, which can minimize the cost function given by

$$J_{\boldsymbol{\omega}_k(i)}(\boldsymbol{\omega}_k(i)) = \mathbb{E}|d_k(i) - \boldsymbol{x}_k^H(i)\boldsymbol{\omega}_k(i)|^2. \tag{65}$$

and the global network cost function is described by

$$J_{\omega}(\boldsymbol{\omega}) = \sum_{k=1}^{N} \mathbb{E}|d_k(i) - \boldsymbol{x}_k^H(i)\boldsymbol{\omega}|^2. \tag{66}$$

We compare the proposed algorithms with the $\mathcal{M}$–$\mathcal{CSE}$ algorithm [4], the distributed RLS algorithm [41], the distributed NLMS algorithm (normalized version of [2]) and distributed principal subspace estimation [11] in terms of MSE performance. The MSE comparison is used to determine the accuracy of the algorithms and the rate of convergence. We define the Hierarchical IEEE–14 bus system as in Fig. 11.

All buses are corrupted by additive white Gaussian noise with variance $\sigma_{n,k}^2 = 0.001$. The step size for the distributed NLMS [2] and the proposed DRJIO–NLMS algorithms is $\mu = 0.15$ and $\eta_0$ is set to 0.5. The parameter vector $\boldsymbol{\omega}_0$ is set to an all–one vector with size $42 \times 1$. For the distributed RLS, DRJIO–RLS algorithms the forgetting factor $\lambda$ is set to 0.99 and $\delta$ is equal to 0.11. The reduced dimension $D$ is set to 10 for both DRJIO–RLS and DRJIO–NLMS algorithm. The results are averaged over 100 independent runs. We simulate the proposed algorithms for smart grids under a static scenario.

From Fig. 12, it can be seen that the proposed DRJIO–RLS algorithm has the best performance, and significantly

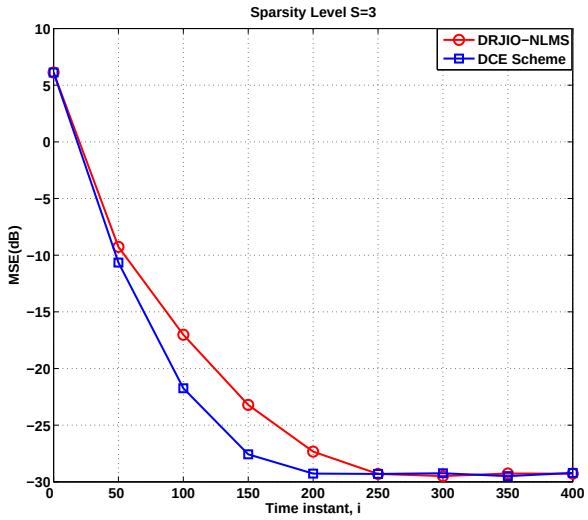


Fig. 9. DRJIO–NLMS vs DCE scheme with sparsity level S=3

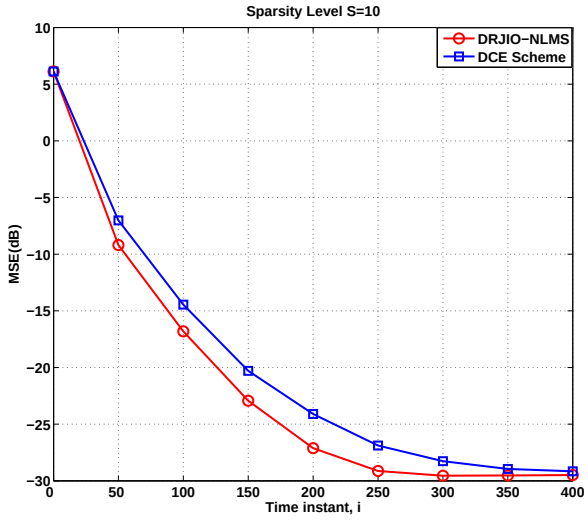


Fig. 10. DRJIO–NLMS vs DCE scheme with sparsity level S=10

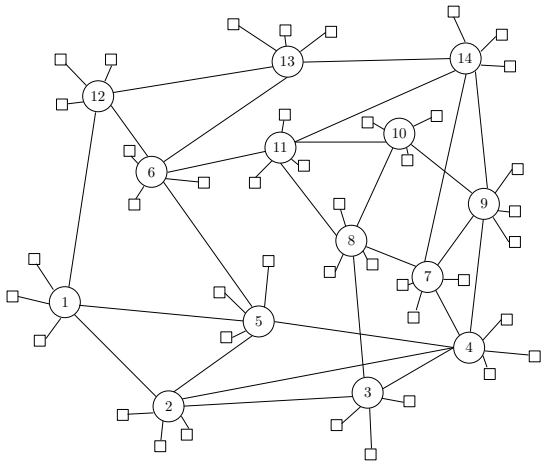

Fig. 11. Hierarchical IEEE 14–bus system

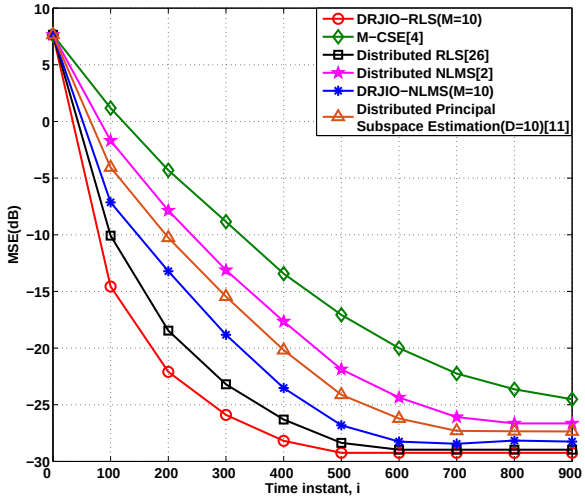


Fig. 12. MSE performance for smart grids

outperforms the distributed NLMS [2] and the $\mathcal{M}$–$\mathcal{CSE}$ [4] algorithms. The DRJIO–NLMS is slightly worse than distributed RLS algorithm [41], but better than the distributed NLMS and $\mathcal{M}$–$\mathcal{CSE}$ algorithms. In addition, the proposed DRJIO–NLMS and DRJIO–RLS algorithms can compress the data to be transmitted from each node from $M$ to $D$, resulting in reduced bandwidth requirements. These algorithms are also important tools for dealing with large sets of data which exhibit some form of redundancy, sparsity and are compressible.

## VII. Conclusions

In this paper, we have proposed a novel distributed low-rank scheme along with efficient algorithms for distributed estimation in wireless sensor networks and smart grids. Simulation results have shown that the proposed DRJIO–RLS has the best performance, while DRJIO–NLMS algorithm has a better performance and lower cost than existing algorithms in all the three scenarios considered. We have also compared the proposed algorithms with the DCE scheme, which was presented in [26], for systems with different levels of sparsity. Furthermore, the proposed scheme requires the transmission of only $D$ parameters instead of $M$, resulting in higher bandwidth efficiency than standard schemes. In addition, with the popularity of neural network based deep learning techniques, the proposed novel distributed low-rank schemes also have the potential to reduce computational complexity of such techniques.

## Appendix
## Reconstruction using a rank-$D$ approximation

In this appendix, we show how the reconstruction of the full-dimension estimator $\boldsymbol{\omega}_k(i)$ can be carried out using a rank-$D$ approximation with the low-rank estimator $\bar{\boldsymbol{\omega}}_k(i)$, i. e.,

$$\boldsymbol{\omega}_k{}^{(D)}(i) = \boldsymbol{S}_{D_k}(i)\bar{\boldsymbol{\omega}}_k(i), \tag{67}$$

In order to show the above relation, we consider the expression of the low-rank estimator given by

$$\begin{aligned}\bar{\boldsymbol{\omega}}_k(i) &= \bar{\boldsymbol{R}}_k^{-1}(i)\bar{\boldsymbol{p}}_k(i) \\ &= \Big(\boldsymbol{S}_{D_k}^H(i)\boldsymbol{R}_k(i)\boldsymbol{S}_{D_k}(i)\Big)^{-1}\boldsymbol{S}_{D_k}^H(i)\boldsymbol{p}_k(i).\end{aligned} \tag{68}$$

and the Wiener filter given by

$$\boldsymbol{\omega}_k(i) = \boldsymbol{R}_k^{-1}(i)\boldsymbol{p}_k(i). \tag{69}$$

Using the fact that the low-rank estimator converges to the low-rank Wiener filter, $\boldsymbol{S}_{D_k}(i)$ converges to a $M \times D$ matrix with the eigenvectors $\boldsymbol{\Phi}_D$ and an eigenvalue decomposition of $\boldsymbol{R}_k(i) = \boldsymbol{\Phi}_N\boldsymbol{\Lambda}_N\boldsymbol{\Phi}_N^H = \sum_{n=1}^N \lambda_n(i)\boldsymbol{\phi}_n(i)\boldsymbol{\phi}_n^H(i)$, where $\boldsymbol{\Lambda}_N$ and $\boldsymbol{\Phi}_N$ are the $N \times N$ diagonal matrix with the eigenvalues and the $N \times N$ unitary matrix with the eigenvectors of $\boldsymbol{R}_k(i)$, respectively, $\lambda_n(i)$ is the $n$th eigenvalue and $\boldsymbol{\phi}_n(i)$ is the $n$th eigenvector of $\boldsymbol{R}_k(i)$, we have

$$\begin{aligned}\bar{\boldsymbol{\omega}}_k(i) &= \Big(\boldsymbol{S}_{D_k}^H(i)\sum_{n=1}^N \lambda_n(i)\boldsymbol{\phi}_n(i)\boldsymbol{\phi}_n^H(i)\boldsymbol{S}_{D_k}(i)\Big)^{-1}\boldsymbol{S}_D^H(i)\boldsymbol{p}_k(i) \\ &= (\boldsymbol{\Phi}_D^H\boldsymbol{\Phi}_N\Lambda_N\boldsymbol{\Phi}_N^H\boldsymbol{\Phi}_D)^{-1}\boldsymbol{\Phi}_D^H\boldsymbol{p}_k(i) \\ &= \Lambda_D^{-1}\boldsymbol{\Phi}_D^H\boldsymbol{p}_k(i),\end{aligned} \tag{70}$$

then, multiplying $\boldsymbol{S}_{D_k}(i) = \boldsymbol{\Phi}_D$ on both sides, we obtain

$$\begin{aligned}\boldsymbol{S}_{D_k}(i)\bar{\boldsymbol{\omega}}_k(i) &= \boldsymbol{S}_{D_k}(i)\Lambda_D^{-1}\boldsymbol{\Phi}_D^H\boldsymbol{p}_k(i) \\ &= \boldsymbol{\Phi}_D\Lambda_D^{-1}\boldsymbol{\Phi}_D^H\boldsymbol{p}_k(i) \\ &= \boldsymbol{\omega}_k{}^{(D)}(i),\end{aligned} \tag{71}$$

where $\boldsymbol{R}_k^{(D)}(i) = \boldsymbol{\Phi}_D\Lambda_D\boldsymbol{\Phi}_D^H$ is a rank-$D$ approximation of $\boldsymbol{R}_k(i)$ and $\boldsymbol{\omega}_k{}^{(D)}(i) = \boldsymbol{\Phi}_D\Lambda_D^{-1}\boldsymbol{\Phi}_D^H\boldsymbol{p}_k(i)$ is the rank-$D$ approximation of $\boldsymbol{\omega}_k(i)$, which gives us the relation in (67). Note that when $D = M$, the $D$-rank approximation yields the full-rank Wiener filter.

## REFERENCES


[1] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 223–229, August 2007.

[2] ——, "Diffusion least–mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, July 2008.

[3] S. Xu, R. C. de Lamare, and H. V. Poor, "Adaptive link selection algorithms for distributed estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, no. 1, p. 86, 2015.

[4] L. Xie, D. H. Choi, S. Kar, and H. V. Poor, "Fully distributed state estimation for wide-area monitoring systems," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1154–1169, September 2012.

[5] S. Xu, R. C. de Lamare, and H. V. Poor, "Dynamic topology adaptation for distributed estimation in smart grids," in *Proc. IEEE 5th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, December 2013, pp. 420–423.

[6] S. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2002.

[7] A. H. Sayed, *Fundamentals of Adaptive Filtering*. Hoboken, NJ, USA: John Wiley&Sons, 2003.

[8] E. Msechu, S. Roumeliotis, A. Ribeiro, and G. Giannakis, "Decentralized quantized kalman filtering with scalable communication cost," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3727–3741, October 2008.

[9] J. J. Xiao, A. Ribeiro, Z. Q. Luo, and G. Giannakis, "Decentralized quantized Kalman filtering with scalable communication cost," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 27–41, July 2006.

[10] S. Pereira and A. Pages-Zamora, "Distributed consensus in wireless sensor networks with quantized information exchange," in *Proc. IEEE 9th Workshop on Signal Processing Advances in Wireless Communications*, July 2008, pp. 241–245.

[11] L. Li, A. Scaglione, and J. H. Manton, "Distributed principal subspace estimation in wireless sensor networks," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 725–738, August 2011.

[12] M. O. Sayin and S. Kozat, "Single bit and reduced dimension diffusion strategies over distributed networks," *IEEE Signal Process. Lett.*, vol. 20, no. 10, pp. 976–979, October 2013.

[13] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Trading off complexity with communication costs in distributed adaptive learning via Krylov subspaces for dimensionality reduction," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 257–273, April 2013.

[14] L. L. Scharf and D. W. Tufts, "Rank reduction for modeling stationary signals," *IEEE Transactions on Acoustics Speech and Signal Processing*, vol. 35, pp. 350–355, 1987.

[15] L. L. Scharf, "The SVD and reduced-rank signal processing," *Signal Processing*, vol. 25, no. 2, pp. 113–133, November 1991.

[16] M. L. Honig and J. S. Goldstein, "Adaptive reduced–rank interference suppression based on the multistage Wiener filter," *IEEE Trans. Commun.*, vol. 6, no. 50, pp. 986–994, June 2002.

[17] R. C. de Lamare and R. Sampaio-Neto, "Adaptive reduced-rank processing based on joint and iterative interpolation, decimation, and filtering," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2503–2514, 2009.

[18] ——, "Reduced–rank adaptive filtering based on joint iterative optimization of adaptive filters," *IEEE Signal Process. Lett.*, vol. 14, no. 12, pp. 980–983, December 2007.

[19] P. Clarke and R. C. de Lamare, "Low–complexity reduced-rank linear interference suppression based on set-membership joint iterative optimization for ds–cdma systems," *IEEE Trans. Veh. Technol.*, vol. 60, no. 9, pp. 4324–4337, November 2011.

[20] Y. Sun, V. Tripathi, and M. L. Honig, "Adaptive, iterative, reduced–rank (turbo) equalization," *IEEE Trans. Wireless Commun.*, vol. 4, no. 6, pp. 2789–2800, March 2005.

[21] R. C. de Lamare and R. Sampaio-Neto, "Adaptive reduced–rank equalization algorithms based on alternating optimization design techniques for MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 60, no. 6, pp. 2482–2494, July 2011.

[22] H. Qian and S. N. Batalama, "Data–record–based criteria for the selection of an auxiliary vector estimator of the MMSE/MVDR filter," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1700–1708, October 2003.

[23] S. Nuan, W. U. Alokozai, R. C. de Lamare, and M. Haardt, "Adaptive widely linear reduced–rank beamforming based on joint iterative optimization," *IEEE Signal Process. Lett.*, vol. 21, no. 3, pp. 265–269, March 2014.

[24] L. Wang and R. C. de Lamare, "Low–complexity constrained adaptive reduced-rank beamforming algorithms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 4, pp. 2114–2128, October 2013.

[25] Y. Yao, A. P. Petropulu, and H. V. Poor, "MIMO radar using compressive sampling," *IEEE J. Sel. Top. Sign. Proces.*, vol. 4, no. 1, pp. 146–163, February 2010.

[26] S. Xu, R. C. de Lamare, and H. V. Poor, "Distributed compressed estimation based on compressive sensing," *IEEE Signal Process. Lett.*, vol. 22, no. 9, pp. 1311–1315, September 2015.

[27] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary learning over distributed models," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 1001–1016, February 2015.

[28] P. Chainais and C. Richard, "Learning a common dictionary over a sensor network," in *Proceedings of IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, December 2013, pp. 1–4.

[29] J. Liang, M. Zhang, X. Zeng, and G. Yu, "Distributed dictionary learning for sparse representation in sensor networks," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 2528–2541, June 2014.

[30] S. Chouvardas, Y. Kopsinis, and S. Theodoridis, "An online algorithm for distributed dictionary learning," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, April 2015, pp. 1–5.

[31] H. Zheng, S. R. Kulkarni, and H. V. Poor, "Attribute-distributed learning: Models, limits, and algorithms," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 386–398, January 2011.

[32] I. Csiszar and G. Tusnady, "Information geometry and alternating minimization procedures," *Statist. Decision Supplement Issue.*, no. 1, pp. 205–237, 1984.

[33] U. Niesen, D. Shah, and G. W. Wornell, "Adaptive alternating minimization algorithms," *IEEE Trans. Inf. Theory.*, vol. 55, no. 3, pp. 1423–1429, March 2009.
[34] Q. Ling and A. Ribeiro, "Decentralized dynamic optimization through the alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1185–1197, March 2014.
[35] H. Bai, G. Li, S. Li, Q. Li, Q. Jiang, and L. Chang, "Alternating optimization of sensing matrix and sparsifying dictionary for compressed sensing," *IEEE Transactions on Signal Processing,*, vol. 63, no. 6, pp. 1581–1594, March 2015.
[36] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu, "Consistency-driven alternating optimization for multigraph matching: A unified approach," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 994–1009, March 2015.
[37] S. Magnusson, P. Chathuranga, M. Rabbat, and C. Fischione, "On the convergence of alternating direction lagrangian methods for nonconvex structured optimization problems," *IEEE Transactions on Control of Network Systems*, vol. PP, no. 99, pp. 1–1, 2015.
[38] A. Bertrand and M. Moonen, "Distributed adaptive node–specific signal estimation in fully connected sensor networks–part II: Simultaneous and asynchronous node updating," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5292–5306, October 2010.
[39] H. Qian and S. N. Batalama, "Data record–based criteria for the selection of an auxiliary vector estimator of the MMSE/MVDR filter," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1700–1708, October 2003.
[40] R. C. de Lamare and R. Sampaio-Neto, "Reduced–rank space–time adaptive interference suppression with joint iterative least squares algorithms for spread-spectrum systems," *IEEE Trans. Veh. Technol.*, vol. 59, no. 3, pp. 1217–1228, March 2010.
[41] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1865–1877, May 2008.
[42] M. L. J. van de Vel, *Theory of Convex Structures*. Elsevier, 1993.
[43] Y. Hua and M. Nikpour, "Computing the reduced-rank Wiener filter by iqmd," *IEEE Signal Process. Lett.*, vol. 6, no. 9, pp. 240–242, September 1999.
[44] H. Ma, Y. Yang, Y. Chen, and K. J. R. Liu, "Distributed state estimation in smart grid with communication constraints," in *Asia–Pacific Signal Information Processing Association Annual Summit and Conference*, December 2012, pp. 1–4.