Digit. Signal Process. 158 (2025) 104920

Contents lists available at ScienceDirect

Digital Signal Processing

journal homepage: www.elsevier.com/locate/dsp

Multitask diffusion adaptation over hyper-networks

Tahereh Bahraini *, Mohammad B. Shamsollahi, Fatemeh Afkhaminia

Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran

ARTICLE INFO

Keywords:
Adaptive filtering
Signal processing
Least mean squares
Adaptive network
Multichannel data

ABSTRACT

Current adaptive networks have several challenges in processing various data types, including medical signals, audio signals, telecommunication signals, etc. Researchers have introduced different methods to solve these challenges. However, these methods still show limitations in the face of certain data types. This paper aims to suggest a new framework using an adaptive filtering approach in a novel concept named hyper-network, in which, a set of hyper-nodes interact with each other, each of which can be a unique adaptive network. The proposed adaptive filtering framework can be based on the least means square (LMS) networks and lead to increased accuracy compared to existing techniques. We performed a theoretical analysis on the convergence of the mean and mean square for the proposed framework. Furthermore, we performed experiments to compare it with multiple other approaches suggested in the field, aiming to assess their effectiveness. The obtained results provide strong evidence for the effectiveness of our suggested framework and highlight its potential.

1. Introduction

Adaptive filters are extensively used in widespread applications across different domains such as system identification [1], improving speech quality [2], eliminating acoustic echoes or acoustic echo cancellation (AEC) [3], noise removal [4], channel equalization [5], etc. To achieve higher accuracy and speed in estimating the target parameter, new generations of adaptive methods were introduced that combined a set of filters in a network. In these methods, the concept of node, edge and network were introduced under the idea of a graph, which was a mechanism for the interaction of filters.

Distributed adaptive estimation poses an appealing and demanding challenge, enabling a network of interconnected nodes to carry out predetermined tasks based on streaming measurements. These tasks may include parameter estimation. While centralized strategies may leverage information gathered from the entire network, mostly, distributed strategies offer greater resilience in solving inference problems autonomously and collaboratively [6].

Recently, the focus of research on distributed estimation problems has been on situations where the networks are used to calculate the single target vector collectively [7]. Different strategies have been suggested to process data sequentially across networks, such as consensus strategies [8–15], incremental strategies [16–20], and diffusion strategies [21–23]. Diffusion strategies are particularly appealing since they offer broader stability ranges and improved adaptation performance when constant step-sizes allow continuous learning [13]. Consequently, this article concentrates on this category of strategies. These strategies aim to estimate a shared parameter vector through the minimization of a global criterion that merges neighborhood cost functions. Nodes interact locally by cooperating solely with their neighbors, without exchanging any global information. As a result, the networks leverage spatial and temporal diversity within the data, leading to powerful learning and tracking capabilities [13,24]. The adaptive networks' performance has been widely investigated in the literature across various scenarios, including imperfect communication and model non-stationarities [25,26]. Additionally, more general cost functions and data models have been considered within this framework [24,27–30], by including additional regularizers [31–33], or extending its use to other scenarios [34–37].

Previous studies on LMS propagation strategies have operated under the working hypothesis that nodes collaborate to obtain a single target vector. These types of problems are referred to as single-task problems. Although, numerous topics of interest involve multiple optimal parameter vectors that must be inferred simultaneously and jointly [38–40]. In the field of distributed estimation, that is the emphasis of this research, there are numerous scenarios where agents are exposed to measuring data derived from distinct models or sensing data that fluctuates in the spatial domain. Multi-task problems arise when multiple optimal parameter vectors need to be simultaneously estimated. These are common in a variety of fields. For example, multi-channel data, such as EEG signals, audio signals obtained from microphone arrays, and telecommunication

* Corresponding author.
E-mail addresses: tahereh.bahraini67@sharif.edu (T. Bahraini), mbshams@sharif.edu (M.B. Shamsollahi), fatemeh.afkhaminia@ee.sharif.edu (F. Afkhaminia).

https://doi.org/10.1016/j.dsp.2024.104920

Available online 10 December 2024

signals received from antenna arrays, contain data from numerous channels, sources, or sensors within the environment.

In our article, we suggest a novel cooperation framework based on the hyper-network concept, which is inspired by cooperation between networks (and not only nodes), and in this new concept, we have collaborative networks. In other words, it can be imagined that we have a network of hyper-nodes, where each hyper-node consists of an internal network. Each internal network can have its unique strategy and interact and cooperate towards the overall goal or a part of the goal of the more extensive network. This image will be more similar to the nesting system in nature. We assess the effectiveness of our suggested framework using simulation trials and offer a theoretical examination of the convergence of our approach, which can use the diffusion least mean square (DLMS) in all or part of its hyper-nodes, for example.

The key findings of this research are outlined below:

- An adaptive filter-based hyper-network framework has been developed, which offers superior efficacy in filtering multi-channel data.
- We have conducted theoretical investigations into the mean and mean square behaviors of the suggested framework.
- A variety simulations were conducted to evaluate the precision of the suggested framework on various types of multi-channel data.

In summary, the proposed algorithm is described in section two. The third part presents how well the proposed framework performs in theory when certain simplifications are taken into account. Furthermore, we provide numerical simulations in the fourth section, comparing the algorithm's performance to traditional algorithms in a filtering environment. Lastly, in the fifth section, we conclude our findings in the article.

Notations: We will utilize the following notation conventions, in this paper:
- We denote vectors and Matrices by bold letters, like $\mathbf{u}_{i,k}(n)$.
- Scalars are represented by small letters such as $d(n)$.
- The notation $(.)^T$ denotes the transpose for the matrix and vector.
- The expectation operator is denoted by the symbol E[.].

2. Proposed framework

Ensuring precise filtering of samples poses a significant challenge when dealing with various data sets. The advent of adaptive filters has revolutionized data processing by employing filters with dynamically updated coefficients over time. Moreover, integrating adaptive structures within networks has enabled the processing of larger data sets at increased speed and accuracy. In line with the evolution of this approach, we aim to introduce a concept termed a hyper-network, serving as a comprehensive framework for data filtering. Within this innovative framework, each hyper-network comprises interconnected hyper-nodes, each possessing a unique structure aligned with the specified goals for processing a given data set. These hyper-nodes can be likened to networks as described in existing literature within the field. Not only does this new hyper-network encompass all previously identified network types, but it also can generate novel designs wherein diverse networks interact with one another. The proposed adaptive framework, which can be constructed using various cooperative strategies (such as based on Least Mean Square (LMS)), provides greater accuracy compared to current techniques. In the maturity of the proposed method theory, we can use any loss function in our optimization problem, such as least mean square (LMS), correntropy, Huber, Log-cosh, etc. But in this article, only for the sake of simplicity and for a better understanding of the optimization problem, we used the LMS loss function, which led to nodes with the LMS optimization problem. Also, nodes are assumed to be dependent across the hyper-network. Based on the descriptions, it is evident that single-task networks, multi-task networks, and clustered multi-task networks [41] are all special cases of the proposed hyper-networks. If all hyper-nodes utilize the same strategy, such as DLMS, and all are single-task, with the hyper-network also being single-task, then the hyper-network will have equivalent performance to a standard single-task DLMS network. Additionally, when all hyper-nodes use the same strategy, for example, DLMS and all are single-task, but the hyper-network is multi-task, then the hyper-network can perform similarly to a standard clustered multi-task DLMS network. These two cases represent the most straightforward scenarios for the proposed hyper-network. Our proposed hyper-network has copied its nature from real-world networks among many animals, plants, parasites, bacteria, etc., where it is a place full of nested sets of collaborators. In general, these networks are nested up to several layers (even to infinity), but given the limitations of our work in theory and simulation, the goal is to simplify the problem solving. Accordingly, this nesting is considered in only two layers, which are: One layer related to the network within each hyper-node, and the other related to the hyper-network. In the following, we conducted a theoretical analysis of the mean and mean square convergence of the proposed framework. Also, we examined this framework and its operational mechanisms.

2.1. Hyper-network model

In the proposed method, it can be claimed that we are facing a hyper-network in which a set of several networks interact with each other. This interaction is of two types: an internal interaction (int-int) between the nodes of each network, and an external interaction (ext-int) between networks. For a better understanding of this issue, Fig. 1 shows an example of this hyper-network. According to this figure, a hyper-network consists of a set of hyper-nodes, each hyper-node being a unique network with internal interactions. Also, a hyper-node has external interactions with other hyper-nodes in its neighborhood. Therefore, an inner neighborhood (int-nei) and an outer neighborhood (out-nei) will be defined corresponding to these definitions. Also, for these neighborhoods, two cooperation weights are defined: inner cooperation weights (for inner interactions) and outer cooperation weights (for outer interactions), respectively. We assume that i-th hyper-node has N_i nodes, and the hyper-network has N_T as total nodes, i.e., $\sum_{i=1}^{C} N_i = N_T$, in which C is the total number of hyper-nodes. As seen in Fig. 1, for internal node k of hyper-node i, there are a unique desired signal $d_{i,k}(n)$ and an input signal $\mathbf{u}_{i,k}(n)$ in time n. $d_{i,k}(n)$ is scalar and $\mathbf{u}_{i,k}(n)$ is $M \times 1$ vector with a positive definite (PD) covariance matrix $\boldsymbol{R}_{u,i,k} = E\{\mathbf{u}_{i,k}(n)\mathbf{u}_{i,k}^T(n)\} > 0$, which is correlated with $d_{i,k}(n)$. The $M \times 1$ input vector $\boldsymbol{u}_{i,k}(n) = [u_{i,k}(n), u_{i,k}(n-1), ..., u_{i,k}(n-M+1)]^T$ consisting of the recent M samples of the signal $u_{i,k}(n)$, and $\boldsymbol{\theta}_{i,k}^* \in R^{M\times 1}$ is an $M \times 1$ optimal solution for unknown target vector of the k-th node of i-th hyper-node i.e., $\boldsymbol{\theta}_{i,k} = [\theta_{0,i,k}, \theta_{1,i,k}, ..., \theta_{M-1,i,k}]^T$. In the general case, the desired signal has the following form:

$$d_{i,k}(n) = \boldsymbol{u}_{i,k}^T(n)\boldsymbol{\theta}_{i,k}^* + \eta_{i,k}(n), \tag{1}$$

where, $\eta_{i,k}(n)$ is an additive noise with $\sigma_{\eta_{i,k}}^2$ variance and zero-mean which is independent of $\boldsymbol{u}_{i,k}(n)$. If we assume that each hyper-node is based on single-task to achieve a target parameter across its network (i.e., we have $\{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_i, ..., \boldsymbol{\theta}_C\}$ instead of $\{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{N_T}\}$), we have:

$$\boldsymbol{\theta}_{i,k}^* = \boldsymbol{\theta}_i^*, \quad \forall k \in \{1, ..., N_i\}. \tag{2}$$

According to this, although each hyper-node is single-task, each one has a different task from the others, which means that the hyper-network is multi-task and it has the signal model as $d_{i,k}(n) = \boldsymbol{u}_{i,k}^T(n)\boldsymbol{\theta}_i^* + \eta_{i,k}(n)$. Additionally, the general form of the error at the i,k-th node for (1) is defined as:

$$e_{i,k}(n) = d_{i,k}(n) - y_{i,k}(n) = d_{i,k}(n) - \boldsymbol{u}_{i,k}^T(n)\boldsymbol{\theta}_{i,k}(n) \tag{3}$$

Here, $\boldsymbol{\theta}_{i,k}(n)$ represents an estimate of the optimal value $\boldsymbol{\theta}_{i,k}^*$ at iteration n.

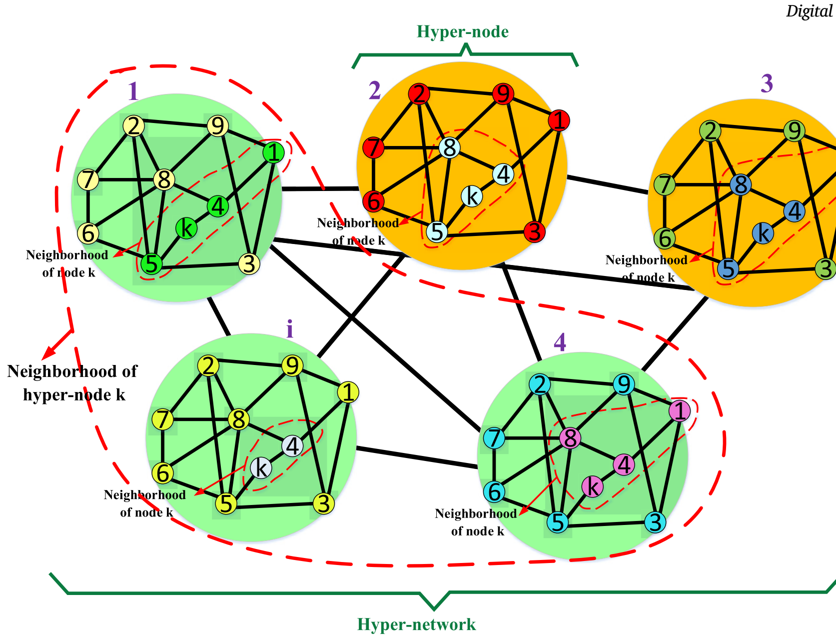


Fig. 1. The overall proposed framework.

2.2. Problem formulation

First, let's stack all the target vectors of all the nodes in the entire hyper-network and display it with a new notation like this: $\underline{\theta} = \{\theta_1, ..., \theta_{N_T}\} = \{\theta_{1,1}, ..., \theta_{1,N_1}, \theta_{2,1}, ..., \theta_{2,N_2}, ..., \theta_{C,1}, ..., \theta_{C,N_C}, \}$. Second, let us define a cost function $J^{Hyper-net}$ for hyper-network consisting of the weighted set of C global cast function $J_i^{Hyper-nod}$ corresponding to i-th hyper-node, which can be formulated as follows:

$$J^{Hyper-net}(\underline{\theta}) = \sum_{i=1}^{C} b_i J_i^{Hyper-nod} = \sum_{i=1}^{C} b_i \sum_{k=1}^{N_i} J_{i,k}(\theta_{i,k}) \tag{4}$$

where $J_i^{Hyper-nod}$ and b_i are, respectively, the cost function of the i-th hyper-node and the combination weight of $J_i^{Hyper-nod}$ with non-negative scalar values. Each of these hyper-nodes can manage their network interactions with a unique approach according to the definition of $J_{i,k}(\theta_{i,k})$, which is the cost function of the k-th node of the i-th Hyper-node. Imagine a simple hyper-network, depicted in Fig. 1, composed of a group of hyper-nodes, each supporting various interaction strategies corresponding to the number of hyper-nodes present in the hyper-network. For example, as drawn in this figure, there are five hyper-nodes, and according to this number, it can be said that a maximum of five different strategies can be used in the hyper-network (as many hyper-nodes as possible). These strategies can be different types suggested in the literature, for example, single-task adaptation-then-combination (ATC) DLMS (ATC-DLMS), single-task combination-then-adaptation (CTA) DLMS (CTA-DLMS), multi-task ATC-DLMS, multi-task CTA-DLMS, diffusion recursive least squares (DRLS), and so on. For the sake of simplicity, let's examine DLMS for every hyper-node in this context. Therefore using (1) the above equation is:

$$\begin{aligned}\{\underline{\theta}^*\} &= \arg\min_{\{\theta_{i,k}\}} J^{Hyper-net}(\underline{\theta}) \\ &= \arg\min_{\{\theta_{i,k}\}} \sum_{i=1}^{C} b_i \sum_{k=1}^{N_i} J_{i,k}(\theta_{i,k}) = \arg\min_{\{\theta_{i,k}\}} \sum_{i=1}^{C} b_i \sum_{k=1}^{N_i} E|d_{i,k}(n) - \boldsymbol{u}_{i,k}^T(n)\theta_{i,k}|^2\end{aligned} \tag{5}$$

We are trying to solve this general form of the optimization problem of the proposed framework, for which various iterative solution strategies can be proposed. In (5), we assumed for simplicity that all hyper-nodes are based on the diffusion strategy. Then, we obtained the hyper-network optimization problem using the DLMS concept. Therefore, two solution strategies can be used for this, i.e., ATC and CTA [42,43,40]. As mentioned in these references, these two iterative solution methods are similar, but in CTA, the step of aggregation is carried out before the step of adaptation. We used the ATC diffusion strategy for our problem:

$$\begin{aligned}\boldsymbol{\psi}_{i,k}(n) &= \boldsymbol{\theta}_{i,k}(n-1) - \mu_{i,k} \sum_{l \in N_{i,k}} \widehat{\nabla_{\theta_{i,l,k}^T}} J_{i,l,k}(\boldsymbol{\theta}_{i,l,k}(n-1)) \\ \boldsymbol{\theta}_{i,k}(n) &= \sum_{l \in N_{i,k}} a_{i,l,k} \boldsymbol{\psi}_{i,l}(n),\end{aligned} \tag{6}$$

where ∇ denotes the gradient operation and $\widehat{\nabla_{\theta_{i,l,k}^T}}$ is the approximation for the true gradient vector $\nabla_{\theta_{i,l,k}^T}$ with respect to $\boldsymbol{\theta}_{i,l,k}^T$. For equation (5), we obtain $\widehat{\nabla_{\theta_{i,l,k}^T}} J_{i,l,k}(\boldsymbol{\theta}_{i,l,k}(n-1)) = -2\boldsymbol{u}_{i,l,k}[d_{i,l,k}(n) - \boldsymbol{u}_{i,l,k}^T(n)\boldsymbol{\theta}_{i,l,k}(n-1)]$. In addition, $\mu_{i,k}$ is a step-size for each node within each hyper-node and can be assumed to be constant or variable. The combination coefficients $a_{i,l,k}$ have the non-negative scalar values which consists of C matrices. The conditions below are valid for each coefficient:

$$\sum_{l=1}^{N_{i,k}} a_{i,l,k} = 1, \quad and \quad a_{i,l,k} = 0 \quad if \quad l \notin N_{i,k}, \tag{7}$$

where $N_{i,k}$ is the neighborhood of the k-th node in the i-th hyper-node. $a_{i,l,k}$ can be updated at any step based on solving an optimization problem, however, we assumed here that it is a positive free variable that is chosen by the user. With the notation in (5), we can obtain the following ATC strategy to solve this problem:

$$\begin{aligned}\boldsymbol{\psi}_{i,k}(n) &= \boldsymbol{\theta}_{i,k}(n-1) + 2\mu_{i,k} \sum_{l \in N_{i,k}} \boldsymbol{u}_{i,l,k}(n) \left[d_{i,l,k}(n) - \boldsymbol{u}_{i,l,k}^T(n)\boldsymbol{\theta}_{i,l,k}(n-1)\right] \\ \boldsymbol{\theta}_{i,k}(n) &= \sum_{l \in N_{i,k}} a_{i,l,k} \boldsymbol{\psi}_{i,l}(n).\end{aligned} \tag{8}$$

Here, the ATC and CTA strategies of the proposed framework are summarized in Algorithms 1 and 2. We focused on the ATC diffusion multi-task LMS algorithm over each hyper-node and also the ATC diffusion multi-task LMS algorithm throughout the hyper-network. In the rest of this section and the following sections, this assumption is used.

Algorithm 1 Diffusion LMS for multitask hyper-networks using ATC strategy.

Input: $\boldsymbol{u}_{i,k}(n), d_{i,k}(n), \quad i = 0, ..., C, \quad k = 1, ..., N_i$
Output: $\boldsymbol{\theta}_{i,k}(n)$

Initialize $\boldsymbol{\theta}_{i,k}(0) = 0$
for each instant of time, $n \geq 0$ **do**
  for each hyper-agent $i = 1, ..., C$ **do**
    start with $\boldsymbol{\theta}_{i,k}(0) = 0$ for all k over hyper-node i and **repeat**

$$\begin{cases} \boldsymbol{\psi}_{i,k}(n) = \boldsymbol{\theta}_{i,k}(n-1) + 2\mu_{i,k} \sum_{l \in N_{i,k}} \boldsymbol{u}_{i,l,k}(n) \left[d_{i,l,k}(n) - \boldsymbol{u}_{i,l,k}^T(n) \boldsymbol{\theta}_{i,l,k}(n-1) \right] \\ \boldsymbol{\theta}_{i,k}(n) = \sum_{l \in N_{i,k}} a_{i,l,k} \boldsymbol{\psi}_{i,l}(n) \end{cases}$$

  end
end

Algorithm 2 Diffusion LMS for multitask hyper-networks using CTA strategy.

Input: $\boldsymbol{u}_{i,k}(n), d_{i,k}(n), \quad i = 0, ..., C, \quad k = 1, ..., N_i$
Output: $\boldsymbol{\theta}_{i,k}(n)$

Initialize $\boldsymbol{\theta}_{i,k}(0) = 0$
for each instant of time, $n \geq 0$ **do**
  for each hyper-agent $i = 1, ..., C$ **do**
    start with $\boldsymbol{\theta}_{i,k}(0) = 0$ for all k over hyper-node i and **repeat**

$$\begin{cases} \boldsymbol{\psi}_{i,k}(n-1) = \sum_{l \in N_{i,k}} a_{i,l,k} \boldsymbol{\theta}_{i,k}(n-1) \\ \boldsymbol{\theta}_{i,k}(n) = \boldsymbol{\psi}_{i,k}(n-1) + 2\mu_{i,k} \sum_{l \in N_{i,k}} \boldsymbol{u}_{i,l,k}(n) \left[d_{i,l,k}(n) - \boldsymbol{u}_{i,l,k}^T(n) \boldsymbol{\psi}_{i,l,k}(n-1) \right] \end{cases}$$

  end
end

2.3. Performance analysis

This section discusses the evaluation of the suggested method in relation to mean and mean square errors. The efficiency analysis of the proposed algorithm for the ATC model is addressed, and a similar analysis for the CTA strategy can also be performed, albeit it is not discussed here. Also, in this section and the experiments section, we focus on the signal model presented in (1), i.e., $d_{i,k}(n) = \boldsymbol{u}_{i,k}^T(n)\boldsymbol{\theta}_{i,k}^* + \eta_{i,k}(n)$. Before we commence, let's establish the following two assumptions:

Assumption 1. All input signals $\mathbf{u}_{i,k}(n)$ are temporally stationary, white, real-valued scalars, and zero-mean with the covariance matrix of $\boldsymbol{R}_{u_{i,k}} = E\{\mathbf{u}_{i,k}(n)\mathbf{u}_{i,k}^T(n)\}$.

Assumption 2. All input signals $\mathbf{u}_{i,k}(n)$ are generated by spatially and temporally independent Gaussian sources, also are independent of additive noise $\eta_{i,k}(n) \in R$ with variance $\sigma^2_{\eta_{i,k}}$.

As nodes and hyper-nodes share data, their current update is influenced using the weighted average obtained for previous estimates. Thus, studying the performance of the entire network would be beneficial in calculating this inter-hyper-node and inter-node dependency. To progress toward this objective, the introduction of new variables is necessary. The proposed algorithm for the k-th node in the i-th hyper-node in the ATC strategy can be articulated as follows:

$$k \in [1, ..., N_i] : \begin{cases} \boldsymbol{\psi}_{i,k}(n) = \boldsymbol{\theta}_{i,k}(n-1) + \mu'_{i,k} \sum_{l \in N_{i,k}} \boldsymbol{u}_{i,l,k}(n) \left[e_{i,l,k}(n) \right] \\ \boldsymbol{\theta}_{i,k}(n) = \sum_{l \in N_{i,k}} a_{i,l,k} \boldsymbol{\psi}_{i,l}(n) \end{cases} \tag{9}$$

Also, in this equation, a new step-size is defined, which is equal to $\mu'_{i,k} = 2\mu_{i,k}$. It is necessary to define some new variables in which local forms are rewritten as global variables. This operation is done by stacking local vectors and also by using block matrices as follows:

$$\boldsymbol{\Theta}_n = col\{\boldsymbol{\theta}_1(n), ..., \boldsymbol{\theta}_{N_T}(n)\}_{MN_T \times 1}, \tag{10}$$

$$\boldsymbol{\Phi}_n = col\{\boldsymbol{\psi}_1(n), ..., \boldsymbol{\psi}_{N_T}(n)\}_{MN_T \times 1}, \tag{11}$$

$$\boldsymbol{U}_n = diag\{\boldsymbol{u}_1(n), ..., \boldsymbol{u}_{N_T}(n)\}_{N_T \times MN_T}, \tag{12}$$

$$\boldsymbol{\Upsilon} = diag\{\mu'_1, ..., \mu'_{N_T}\}_{MN_T \times MN_T}, \tag{13}$$

$$\boldsymbol{D}_n = col\{d_1(n), ..., d_{N_T}(n)\}_{N_T \times 1}, \tag{14}$$

$$\boldsymbol{V}_n = col\{\eta_1(n), ..., \eta_{N_T}(n)\}_{N_T \times 1}, \tag{15}$$

where $col\{.\}$ operator stacks its vector arguments. Also, $diag\{.\}$ is an operator that arranges its input argument in the form of a block diagonal matrix. According to the definitions provided above, a new set of equations is formulated to represent the entire hyper-network. To start with, the relationship between the measurements is as follows:

$$\boldsymbol{D}_n = \boldsymbol{U}_n \boldsymbol{\Theta}^* + \boldsymbol{V}_n, \tag{16}$$

where, $\boldsymbol{\Theta}^* = \boldsymbol{I}\boldsymbol{\theta}^*$ and $\boldsymbol{I}$ is a matrix in form of $\boldsymbol{I} = col\{\boldsymbol{I}_M, ..., \boldsymbol{I}_M\}_{MN_T \times M}$. Therefore, $\boldsymbol{\Theta}^*$ is one vector with $MN_T \times 1$ dimension denoted the optimal value of $\boldsymbol{\Theta}$. Then, the update equations for the entire hyper-network are rewritten as follows:

$$j \in [1, ..., N_T] : \begin{cases} \boldsymbol{\Phi}_n = \boldsymbol{\Theta}_{n-1} + \boldsymbol{\Upsilon}\boldsymbol{U}_n^T(\boldsymbol{D}_n - \boldsymbol{U}_n\boldsymbol{\Theta}_{n-1}) \\ \boldsymbol{\Upsilon} = \mu_{i,k}\boldsymbol{\Omega} \\ \boldsymbol{\Omega} = [2\boldsymbol{I}_M, ..., 2\boldsymbol{I}_M] \\ \boldsymbol{\Theta}_n = \boldsymbol{B}\boldsymbol{\Phi}_n \end{cases}, \tag{17}$$

where, $\boldsymbol{B} = \boldsymbol{A} \otimes \boldsymbol{I}_m$ is one matrix with $MN_T \times MN_T$ dimension and $\otimes$ is Kronecker product. Also, $\boldsymbol{A}$ is the weighted matrix with $N_T \times N_T$ dimension where $\{\boldsymbol{A}\}_{\ell,k} = a_{\ell,k}$. Mean and mean square analysis of the proposed method are performed using the above equation (17). First, the target parameter error vector for node k in hyper-node i is defined as follows:

$$\tilde{\boldsymbol{\theta}}_{i,k}(n) = \boldsymbol{\theta}_{i,k}^* - \boldsymbol{\theta}_{i,k}(n) \tag{18}$$

Note 1. In mean analysis, we try to obtain a bound for the step-size, which guarantees convergence in the average, i.e., $0 < \mu_{i,k} < UB$, where UB denote the upper bound.

The target parameter error vector for the entire hyper-network is equal to:

$$\tilde{\boldsymbol{\Theta}}_n = \boldsymbol{\Theta}^* - \boldsymbol{\Theta}_n \tag{19}$$

Since $\boldsymbol{B}\boldsymbol{\Theta}^* = \boldsymbol{\Theta}^*$, the target parameter error vector for the entire hyper-network can be rewritten in terms of other matrices:

$$
\begin{aligned}
j \in [1,\dots,N_T]: \tilde{\boldsymbol{\Theta}}_n &= \boldsymbol{\Theta}^* - \boldsymbol{\Theta}_n = \boldsymbol{\Theta}^* - \boldsymbol{B}\boldsymbol{\Phi}_n \\
&= \boldsymbol{\Theta}^* - \boldsymbol{B}\left[\boldsymbol{\Theta}_{n-1} + \boldsymbol{\Upsilon}\boldsymbol{U}_n^T(\boldsymbol{D}_n - \boldsymbol{U}_n\boldsymbol{\Theta}_{n-1})\right] \\
&= \boldsymbol{B}\tilde{\boldsymbol{\Theta}}_{n-1} - \boldsymbol{B}\left[\boldsymbol{\Upsilon}\boldsymbol{U}_n^T(\boldsymbol{D}_n - \boldsymbol{U}_n\boldsymbol{\Theta}_{n-1})\right] \\
&= \boldsymbol{B}\tilde{\boldsymbol{\Theta}}_{n-1} - \boldsymbol{B}\left[\boldsymbol{\Upsilon}\boldsymbol{U}_n^T(\boldsymbol{U}_n\boldsymbol{\Theta}^* + \boldsymbol{V}_n - \boldsymbol{U}_n\boldsymbol{\Theta}_{n-1})\right] \\
&= \boldsymbol{B}\tilde{\boldsymbol{\Theta}}_{n-1} - \boldsymbol{B}\left[\boldsymbol{\Upsilon}\boldsymbol{U}_n^T(\boldsymbol{U}_n\tilde{\boldsymbol{\Theta}}_{n-1} + \boldsymbol{V}_n)\right] \\
&= \boldsymbol{B}\left[\boldsymbol{I}_{MN_T} - \boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{U}_n\right]\tilde{\boldsymbol{\Theta}}_{n-1} - \boldsymbol{B}\boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{V}_n
\end{aligned} \tag{20}
$$

In the following, this equation in (20) is used for mean and mean square analysis.

2.3.1. Mean behavior

To check the stability and convergence for the mean of the global error vector in the proposed method, it is necessary to obtain the mathematical expectation of the global error vector in relation (20).

Note 2. In the mean square analysis, an expression of the transient state and steady state for mean square deviation (MSD) is obtained.

The MSD for node k, hyper-node i, as well as for the whole hyper-network is as follows:

$$MSD_{i,k} = E\left[\|\tilde{\theta}_{i,k}(n)\|^2\right] = E\left[\|\theta_i^* - \theta_{i,k}(n)\|^2\right], \tag{21}$$

$$MSD_i = \frac{1}{N_i}\sum_{k=1}^{N_i} MSD_k; \tag{22}$$

$$MSD_{Hy-net} = \frac{1}{C}\sum_{i=1}^{C} MSD_i. \tag{23}$$

Note 3. Matrix $\boldsymbol{\Upsilon}$ is assumed to be independent of the input signal matrix $\boldsymbol{U}_n$.

As a result, we will have:

$$j \in [1,\dots,N_T]: \quad E\left[\boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{U}_n\right] \cong E[\boldsymbol{\Upsilon}]\,E\left[\boldsymbol{U}_n^T\boldsymbol{U}_n\right] = E[\boldsymbol{\Upsilon}]\,\boldsymbol{R}_U \tag{24}$$

where $\boldsymbol{R}_U = E\left[\boldsymbol{U}_n^T\boldsymbol{U}_n\right]$ is the auto-correlation matrix for the input signal matrix $\boldsymbol{U}_n$.

By taking $E[.]$ from the sides of (20), we will have:

$$E\left[\tilde{\boldsymbol{\Theta}}_n\right] = \boldsymbol{B}\left[\boldsymbol{I}_{MN_T} - E[\boldsymbol{\Upsilon}]\,\boldsymbol{R}_u\right]E\left[\tilde{\boldsymbol{\Theta}}_{n-1}\right] - \boldsymbol{B}E[\boldsymbol{\Upsilon}]\,E\left[\boldsymbol{U}_n^T\right]E\left[\boldsymbol{V}_n\right] \tag{25}$$

Using assumption 2, the mathematical expectation of the second term on the right side in the above relation (25) is zero. Therefore, we have:

$$E\left[\tilde{\boldsymbol{\Theta}}_n\right] = \boldsymbol{B}\left[\boldsymbol{I}_{mN} - E[\boldsymbol{\Upsilon}]\,\boldsymbol{R}_u\right]E\left[\tilde{\boldsymbol{\Theta}}_{n-1}\right] \tag{26}$$

In the above equation, to ensure the convergence of the mean error vector, the following must be satisfied:

$$\left|\lambda_{\max}\left(\boldsymbol{B}\left[\boldsymbol{I}_{MN_T} - E[\boldsymbol{\Upsilon}]\,\boldsymbol{R}_u\right]\right)\right| = \left|\lambda_{\max}(\boldsymbol{B}\boldsymbol{Z})\right| < 1 \tag{27}$$

where $\boldsymbol{Z} = \left[\boldsymbol{I}_{MN_T} - E[\boldsymbol{\Upsilon}]\,\boldsymbol{R}_u\right]$. Also, $\lambda_{\max}(.)$ is the maximum eigenvalue operation. That means the spectra of $\boldsymbol{B}\boldsymbol{Z}$ matrix must be inside the unit circle. This issue originates from Lyapunov stability, which states that a matrix is stable if all its eigenvalues lie within the unit circle. Using the inequality $\|\boldsymbol{B}\boldsymbol{Z}\|^2 \le \|\boldsymbol{B}\|^2\|\boldsymbol{Z}\|^2$, the fact that $\boldsymbol{Z}$ is a Hermitian matrix, and $\boldsymbol{B} = \boldsymbol{A} \otimes \boldsymbol{I}_M$, the above equation (27) can be rewritten as follows:

$$\left|\lambda_{\max}(\boldsymbol{B}\boldsymbol{Z})\right| \le \|\boldsymbol{A}\|^2\left|\lambda_{\max}(\boldsymbol{Z})\right| \tag{28}$$

Since $\|\boldsymbol{A}\|^2 = 1$, for non-cooperative designs we have $\boldsymbol{B} = \boldsymbol{I}_{MN_T}$. So based on this, the recent equations are rewritten as follows:

$$\left|\lambda_{\max}(\boldsymbol{B}\boldsymbol{Z})\right| \le \left|\lambda_{\max}(\boldsymbol{Z})\right| \tag{29}$$

Based on this, it can be said that cooperation plans can improve the stability of the system. Therefore, the algorithm is stable if:

$$\lim_{N_T\to\infty}\prod_{n=0}^{N_T}\left[\boldsymbol{I}_{MN_T} - E\left[\mu_{i,k}\right]\boldsymbol{R}_{u_{i,k}}\right] \to 0 \tag{30}$$

This is established if the average step-size is in the following bound:

$$0 < E\left[\mu_{i,k}\right] < \frac{2}{\lambda_{\max}\left(\boldsymbol{R}_{u_{i,k}}\right)} \tag{31}$$

To obtain step-size bands, we will continue:

$$0 < \mu_{i,k} < \frac{2}{\lambda_{\max}\left(\boldsymbol{R}_{u_{i,k}}\right)} \tag{32}$$

So, the proposed method is stable when the step-size for each node is in the bands (32).

Note 4. The proposed algorithm with ATC strategy has convergence condition $0 < \mu_{i,k} < \frac{2}{\lambda_{\max}\left(\boldsymbol{R}_{u_{i,k}}\right)}$.

2.3.2. Mean square behavior

Here, the performance of the mean square for the proposed method is investigated. For this purpose, it is necessary first to obtain the weighted norm of equation (20) and then calculate its expectation. In other words, to analyze the mean square efficiency, $E\left[\|\tilde{\boldsymbol{\Theta}}_n\|_{\boldsymbol{\Sigma}^2}\right]$ should be evaluated using the equation (20), where $\boldsymbol{\Sigma}$ is a positive definite Hermitian matrix. In this section, the equations are thoroughly investigated for the entire hyper-network. So we will have:

$$
\begin{aligned}
j \in [1,\dots,N_T]: \quad & E\left[\|\tilde{\boldsymbol{\Theta}}_n\|_{\Sigma}^2\right] \\
&= E\left[\left\|\boldsymbol{B}\left[\boldsymbol{I}_{MN_T} - \boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{U}_n\right]\tilde{\boldsymbol{\Theta}}_{n-1} - \boldsymbol{B}\boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{V}_n\right\|_{\boldsymbol{\Sigma}^2}\right] \\
&= E\left[\|\tilde{\boldsymbol{\Theta}}_{n-1}\|^2_{\boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B}}\right] - E\left[\|\tilde{\boldsymbol{\Theta}}_{n-1}\|^2_{\boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{U}_n}\right] - E\left[\|\tilde{\boldsymbol{\Theta}}_{n-1}\|^2_{\boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{B}}\right] \\
&+ E\left[\|\tilde{\boldsymbol{\Theta}}_{n-1}\|^2_{\boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{U}_n}\right] + E\left[\boldsymbol{V}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{V}_n\right] \\
&= E\left[\|\tilde{\boldsymbol{\Theta}}_{n-1}\|^2_{\tilde{\boldsymbol{\Sigma}}}\right] + E\left[\boldsymbol{V}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{V}_n\right]
\end{aligned} \tag{33}
$$

where in (33), $\boldsymbol{\Gamma}_n = \boldsymbol{B}\boldsymbol{\Upsilon}\boldsymbol{U}_n^T$. Also, in this, we have:

$$
\begin{aligned}
\tilde{\boldsymbol{\Sigma}} &= \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} - \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B}\boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{U}_n - \boldsymbol{U}_n^T\boldsymbol{U}_n\boldsymbol{\Upsilon}^T\boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} \\
&+ \boldsymbol{U}_n^T\boldsymbol{U}_n\boldsymbol{\Upsilon}^T\boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B}\boldsymbol{\Upsilon}\boldsymbol{U}_n^T\boldsymbol{U}_n \\
&= \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} - \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{U}_n - \boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{B} + \boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\boldsymbol{\Sigma}\boldsymbol{\Gamma}_n\boldsymbol{U}_n
\end{aligned} \tag{34}
$$

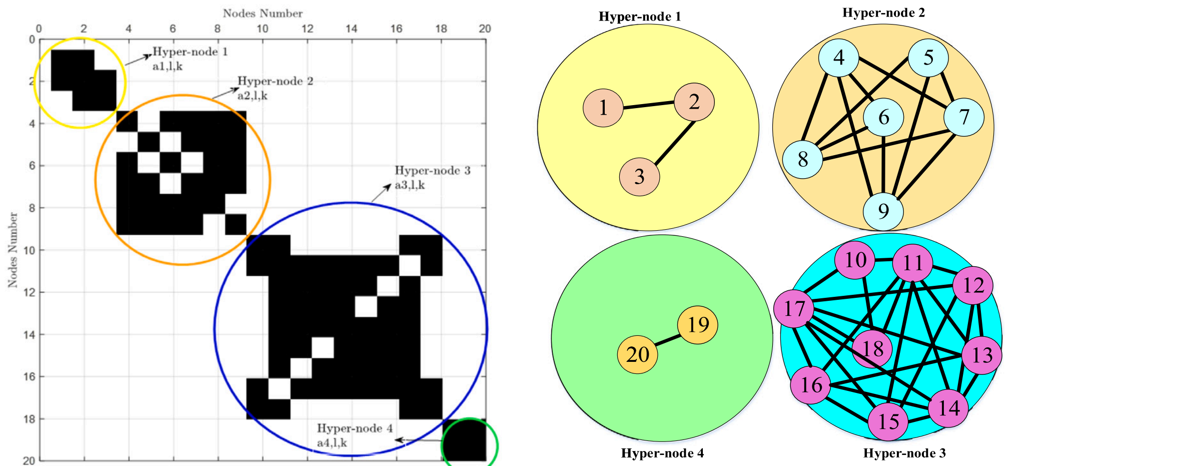
which, according to the assumption of independence of data and applying the $E[.]$ operator, we obtained:

$$
\begin{aligned}
E\left[\tilde{\boldsymbol{\Sigma}}\right] &= \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} - \boldsymbol{B}^T\boldsymbol{\Sigma}E\left[\boldsymbol{\Gamma}_n\boldsymbol{U}_n\right] - E\left[\boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\right]\boldsymbol{\Sigma}\boldsymbol{B} \\
&+ E\left[\boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\right]\boldsymbol{\Sigma}E\left[\boldsymbol{\Gamma}_n\boldsymbol{U}_n\right] \\
&= \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} - \boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B}E[\boldsymbol{\Upsilon}]\,E\left[\boldsymbol{U}_n^T\boldsymbol{U}_n\right] - E\left[\boldsymbol{U}_n^T\boldsymbol{U}_n\right]E[\boldsymbol{\Upsilon}]\,\boldsymbol{B}^T\boldsymbol{\Sigma}\boldsymbol{B} \\
&+ E\left[\boldsymbol{U}_n^T\boldsymbol{\Gamma}_n^T\right]\boldsymbol{\Sigma}E\left[\boldsymbol{\Gamma}_n\boldsymbol{U}_n\right]
\end{aligned} \tag{35}
$$

To simplify the equation, we will use $E\left[\tilde{\boldsymbol{\Sigma}}\right] = \boldsymbol{\Sigma}'$ in the following. According to assumption 1, the auto-covariance matrix can be decomposed as follows:

$$\boldsymbol{R}_U = E\left[\boldsymbol{U}_n\boldsymbol{U}_n^T\right] = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \tag{36}$$

where the diagonal matrix $\boldsymbol{\Lambda}$ is containing the eigenvalues of the whole hyper-network, and $\boldsymbol{Q}$ is a matrix including the eigenvectors associated with these eigenvalues. By employing this decomposition, we can specify: $\bar{\boldsymbol{\Theta}}_n = \boldsymbol{Q}^T\tilde{\boldsymbol{\Theta}}_n$, $\bar{\boldsymbol{U}}_n = \boldsymbol{U}_n\boldsymbol{Q}$, $\bar{\boldsymbol{B}} = \boldsymbol{Q}^T\boldsymbol{B}\boldsymbol{Q}$, $\bar{\boldsymbol{\Sigma}} = \boldsymbol{Q}^T\boldsymbol{\Sigma}\boldsymbol{Q}$, $\bar{\boldsymbol{\Sigma}}' = \boldsymbol{Q}^T\boldsymbol{\Sigma}'\boldsymbol{Q}$, and $\bar{\boldsymbol{\Upsilon}} = \boldsymbol{Q}^T\boldsymbol{\Upsilon}\boldsymbol{Q} = \boldsymbol{\Upsilon}$. It is also assumed that the input signals at each node



a) Connection matrix **b) Hyper-network Topology**

Fig. 2. First scenario all hyper-nodes are non-cooperative: a) Connection matrix and b) Hyper-network topology used in simulations of proposed method.

are independent of each other and the step-size matrix $\boldsymbol{\Upsilon}$ is diagonal. Thus, it doesn't change because $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$. Using these, we can write:

$$E\left[\|\tilde{\boldsymbol\Theta}_n\|^2_{\bar{\boldsymbol\Sigma}}\right] = E\left[\|\tilde{\boldsymbol\Theta}_{n-1}\|^2_{\bar{\boldsymbol\Sigma}'}\right] + E\left[\boldsymbol V_n^T\bar{\boldsymbol\Gamma}_n^T\boldsymbol\Sigma\bar{\boldsymbol\Gamma}_n\boldsymbol V_n\right] \tag{37}$$

where,

$$\begin{aligned}\bar{\boldsymbol\Sigma}' &= \bar{\boldsymbol B}^T\bar{\boldsymbol\Sigma}\bar{\boldsymbol B} - \bar{\boldsymbol B}^T\bar{\boldsymbol\Sigma}\bar{\boldsymbol B}E\left[\boldsymbol\Upsilon\right]E\left[\bar{\boldsymbol U}_n^T\bar{\boldsymbol U}_n\right] - E\left[\bar{\boldsymbol U}_n^T\bar{\boldsymbol U}_n\right]E\left[\boldsymbol\Upsilon\right]\bar{\boldsymbol B}^T\bar{\boldsymbol\Sigma}\bar{\boldsymbol B}\\ &+E\left[\bar{\boldsymbol U}_n^T\bar{\boldsymbol\Gamma}_n^T\right]\boldsymbol\Sigma E\left[\bar{\boldsymbol\Gamma}_n\bar{\boldsymbol U}_n\right]\end{aligned} \tag{38}$$

which, we have $\bar{\boldsymbol\Gamma}_n = \bar{\boldsymbol B}\boldsymbol\Upsilon\bar{\boldsymbol U}_n^T$. It is obvious that $E\left[\bar{\boldsymbol U}_n^T\bar{\boldsymbol U}_n\right] = \boldsymbol\Lambda$. Utilizing the $bvec\{.\}$ operator, we can define $\bar{\boldsymbol\zeta} = bvec\left\{\bar{\boldsymbol\Sigma}\right\}$, which $bvec\{.\}$ organizes the matrix into smaller block components and then uses the $vec\{.\}$ operator to each of these parts. The covariance matrix of noise for the whole hyper-network, i.e., $\boldsymbol R_V = \boldsymbol\Lambda_V \odot \boldsymbol I_M$, is a block-diagonal matrix, $\odot$ represents the block Kroniker multiplication, and $\boldsymbol\Lambda_V$ is a diagonal matrix related to the noise variance for the entire hyper-network. Therefore, the second term from the right side of the (37) is equal to:

$$E\left[\boldsymbol V_n^T\bar{\boldsymbol\Gamma}_n^T\boldsymbol\Sigma\bar{\boldsymbol\Gamma}_n\boldsymbol V_n\right] = \boldsymbol\aleph_n^T\bar{\boldsymbol\zeta} \tag{39}$$

where $\boldsymbol\aleph_n^T = bvec\{\boldsymbol R_v E\left[\boldsymbol\Upsilon^2\right]\boldsymbol\Lambda\}$. From (38), the fourth-order moment, i.e., term $E\left[\bar{\boldsymbol U}_n^T\bar{\boldsymbol\Gamma}_n^T\right]\boldsymbol\Sigma E\left[\bar{\boldsymbol\Gamma}_n\bar{\boldsymbol U}_n\right]$, remains to be calculated. By the assumption that the step-size and the operator $\odot$ are independent, we have:

$$bvec\{E[\bar{\boldsymbol U}_n^T\bar{\boldsymbol\Gamma}_n^T\boldsymbol\Sigma\bar{\boldsymbol U}_n\bar{\boldsymbol\Gamma}_n]\} = E[\boldsymbol\Upsilon_n\odot\boldsymbol\Upsilon_n]\boldsymbol S(\boldsymbol B^T\odot\boldsymbol B^T)\bar{\boldsymbol\zeta} \tag{40}$$

where,

$$\begin{aligned}&\boldsymbol S = diag\left\{\boldsymbol S_1,...,\boldsymbol S_j,...,\boldsymbol S_{N_T}\right\},\\ &\Rightarrow \boldsymbol S_j = diag\left\{\boldsymbol\Lambda_1\otimes\boldsymbol\Lambda_j,...,\lambda_j\lambda_j^T + 2\boldsymbol\Lambda_j\otimes\boldsymbol\Lambda_j,...,\boldsymbol\Lambda_{N_T}\otimes\boldsymbol\Lambda_j\right\}\end{aligned} \tag{41}$$

where $\boldsymbol\Lambda_j$ describes the diagonal matrix of eigenvalues and λ_j is the vector of eigenvalues for node j (in the hyper-network with total node N_T where these nodes are the sum of the nodes in all hyper-nodes). In (40) $E[\boldsymbol\Upsilon_n\odot\boldsymbol\Upsilon_n]$ is obtained as:

$$\begin{aligned}&(E[\boldsymbol\Upsilon_n\odot\boldsymbol\Upsilon_n])_{jj}\\ &= E\left[diag\left\{\mu_j\boldsymbol I_M\otimes\mu_1\boldsymbol I_{\boldsymbol M},...,\mu_j\boldsymbol I_M\otimes\mu_j\boldsymbol I_M,...,\mu_j\boldsymbol I_M\otimes\mu_{N_T}\boldsymbol I_M\right\}\right]\end{aligned} \tag{42}$$

$$\begin{aligned}&= E\left[diag\left\{\mu_j\mu_1\boldsymbol I_{M^2},...,{\mu_j}^2\boldsymbol I_{M^2},...,\mu_j\mu_{N_T}\boldsymbol I_{M^2}\right\}\right]\\ &= diag\left\{E\left[\mu_j\right]E\left[\mu_1\right]\boldsymbol I_{M^2},...,E\left[\mu_j\right]E\left[\mu_{N_T}\right]\boldsymbol I_{M^2}\right\}\end{aligned}$$

Then, by applying the operator $bvec\{.\}$ to the weighted matrix $\bar{\boldsymbol\Sigma}'$ using the relation $\bar{\boldsymbol\zeta} = bvec\left\{\bar{\boldsymbol\Sigma}'\right\}$, we can get the original $\bar{\boldsymbol\Sigma}'$ (using $bvec\left\{\bar{\boldsymbol\zeta}\right\} = \bar{\boldsymbol\Sigma}'$).

$$\begin{aligned}&bvec\left\{\bar{\boldsymbol\Sigma}'\right\} = \bar{\boldsymbol\zeta}\\ &= [\boldsymbol I_{M^2N_T^2} - (\boldsymbol I_{MN_T}\odot\boldsymbol\Lambda E[\boldsymbol\Upsilon]) - (\boldsymbol\Lambda E[\boldsymbol\Upsilon]\odot\boldsymbol I_{MN_T})]\\ &+ E[\boldsymbol\Upsilon\odot\boldsymbol\Upsilon]\times\boldsymbol S(\boldsymbol B^T\odot\boldsymbol B^T)\times\bar{\boldsymbol\zeta} = \boldsymbol F_n\bar{\boldsymbol\zeta}\end{aligned} \tag{43}$$

where,

$$\begin{aligned}\boldsymbol F_n &= [\boldsymbol I_{M^2N_T^2} - (\boldsymbol I_{MN_T}\odot\boldsymbol\Lambda E[\boldsymbol\Upsilon]) - (\boldsymbol\Lambda E[\boldsymbol\Upsilon]\odot\boldsymbol I_{MN_T})]\\ &+ E[\boldsymbol\Upsilon\odot\boldsymbol\Upsilon]\times\boldsymbol S(\boldsymbol B^T\odot\boldsymbol B^T)\end{aligned} \tag{44}$$

Then, (37) is rewritten as follows:

$$E\left[\|\tilde{\boldsymbol\Theta}_n\|^2_{\bar{\zeta}}\right] = E\left[\|\tilde{\boldsymbol\Theta}_{n-1}\|^2_{\boldsymbol F_n\bar{\zeta}}\right] + \boldsymbol\aleph_n^T\bar{\boldsymbol\zeta} \tag{45}$$

This equation in (45) defines the transient behavior of the proposed hyper-network. Although this does not clearly describe the performance of the proposed framework, in fact, according to this equation, it can be said that the stability of the proposed framework depends on the stability of the weighting matrix $\boldsymbol F_n$, which is different in each iteration. However, according to equation (44), the stability is also dependent on the stability of the diagonal matrix of the motion step $\boldsymbol\Upsilon$. According to the mean square stability analysis in the previous subsection, the stability of (45) is guaranteed when (32) holds. Therefore, the relation (32) is sufficient for the convergence of the mean and mean square in the whole hyper-network.

3. Simulation results

This section presents findings from simulations that confirm the effectiveness of the new framework and the theoretical findings. Initially, all nodes had their parameter vectors $\boldsymbol\theta_{i,k}(0)$ set to zero. The simulated curves resulted from averaging data over 1000 runs, which validated consistency with the theoretical results.

3.1. Model specification

As shown in Figs. 2 and 3, we designed two particular scenarios: the first scenario is a hyper-network with non-collaborating hyper-nodes, the second scenario is a hyper-network with cooperating hyper-nodes, which are respectively seen in these two figures. This issue is realized by defining the matrix of combination coefficients according to what is

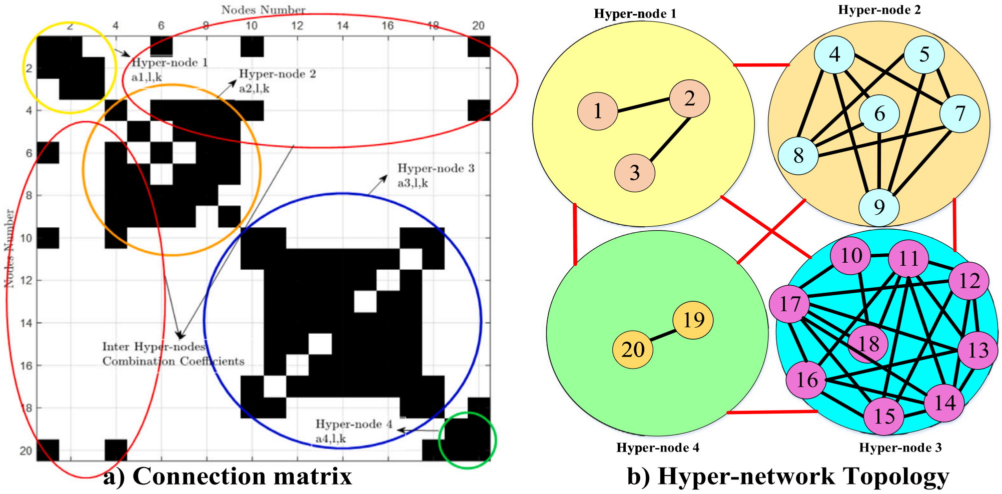


Fig. 3. Second scenario the hyper-nodes are cooperative: a) Connection matrix and b) Hyper-network topology used in simulations of proposed method.

Table 1
The considered values for the combination coefficient matrix consist of $a_{i,l,k}$.

Values	Scenario 1	Scenario 2
Num.	1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
1	1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1
2	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3	0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4	0 0 0 1 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1
5	0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0
6	0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0	1 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0
7	0 0 0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0
8	0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
9	0 0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0
10	0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0	1 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0
11	0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 0 0	0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 0 0
12	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0
13	0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 0 0
14	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 0
15	0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 0 0
16	0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0
17	0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1 0 0	0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1 0 0
18	0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0	0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 1
19	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
20	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1

shown in part a) of Figs. 2 and 3 and also in Table 1. Specifically, in these scenarios, we have four hyper-nodes, where nodes 1 to 3 form the first hyper-node, nodes 4 to 9 constitute the second hyper-node, nodes 10 to 18 make up the third hyper-node, and nodes 19 and 20 belong to the fourth hyper-node. In other words, for $C = 4$, we can say that:

$$N_i = \begin{cases} N_1 = 3, & k \in \{1 : 3\} \\ N_2 = 6, & k \in \{4 : 9\} \\ N_3 = 9, & k \in \{10 : 18\} \\ N_4 = 2, & k \in \{19 : 20\} \end{cases} \tag{46}$$

The synthetic input signal for regression denoted as $\boldsymbol{u}_{i,k}(n)$, comprised random $M \times 1$ vectors with a mean of zero in which M is assumed to be 2 and 4 (whose value is mentioned in each simulation). In Fig. 4, for the case where $M = 2$, a 2×1 vector of the input signal $\boldsymbol{u}_{i,k}(n)$ is generated for 1000 time iterations. For better display, it is drawn as a 2000×1 vector for all nodes in the hyper-network. In this figure, you can see four sub-figures, which correspond to the input signals of hyper-nodes 1 to 4, respectively. In addition, the desired signals $d_{i,k}(n)$ of these hyper-nodes are plotted in Fig. 5. The proposed hyper-network updates the weights in such a way that after filtering the input signal, the similarity between the desired and input signals is maximized. These vectors were generated using a Gaussian distribution with covariance matrices $\boldsymbol{R}_{u,i,k} = \sigma^2_{u,i,k} \boldsymbol{I}_M$. The background noises, $\eta_{i,k}(n)$, were independent and identically distributed Gaussian random variables with zero mean and were not correlated with any other signals. The variances, $\sigma^2_{u,i,k}$ and $\sigma^2_{\eta_{i,k}}$, are shown in Fig. 6. In one part of our simulations for $M = 2$, the optimal values of target parameter vectors θ^*_i to be estimated are set as:

$$\boldsymbol{\theta}^*_i = \begin{cases} \boldsymbol{\theta}^*_1 = [0.25 \quad -0.30]^T \\ \boldsymbol{\theta}^*_2 = [0.50 \quad -0.40]^T \\ \boldsymbol{\theta}^*_3 = [0.30 \quad -0.35]^T \\ \boldsymbol{\theta}^*_4 = [-0.55 \quad 0.45]^T \end{cases} \tag{47}$$

where $\boldsymbol{\theta}^*_i$ is the target of all the nodes inside the i-th hyper-node. In another part of the simulations, we gave M different values; for example, we assumed $M = 8$, and the corresponding weight vector has the following values:

$$\boldsymbol{\theta}^*_i = \begin{cases} \boldsymbol{\theta}^*_1 = [0.25 \quad -0.40 \quad 0.10 \quad -0.30 \quad -0.35 \quad -0.45 \quad -0.10 \quad -0.30]^T \\ \boldsymbol{\theta}^*_2 = [0.50 \quad 0.70 \quad -0.10 \quad -0.40 \quad 0.70 \quad 0.10 \quad 0.60 \quad -0.20]^T \\ \boldsymbol{\theta}^*_3 = [0.30 \quad 0.10 \quad -0.70 \quad -0.35 \quad 0.30 \quad -0.10 \quad 0.70 \quad -0.40]^T \\ \boldsymbol{\theta}^*_4 = [-0.55 \quad -0.90 \quad -0.10 \quad 0.45 \quad -0.55 \quad -0.90 \quad -0.10 \quad 0.45]^T \end{cases} \tag{48}$$

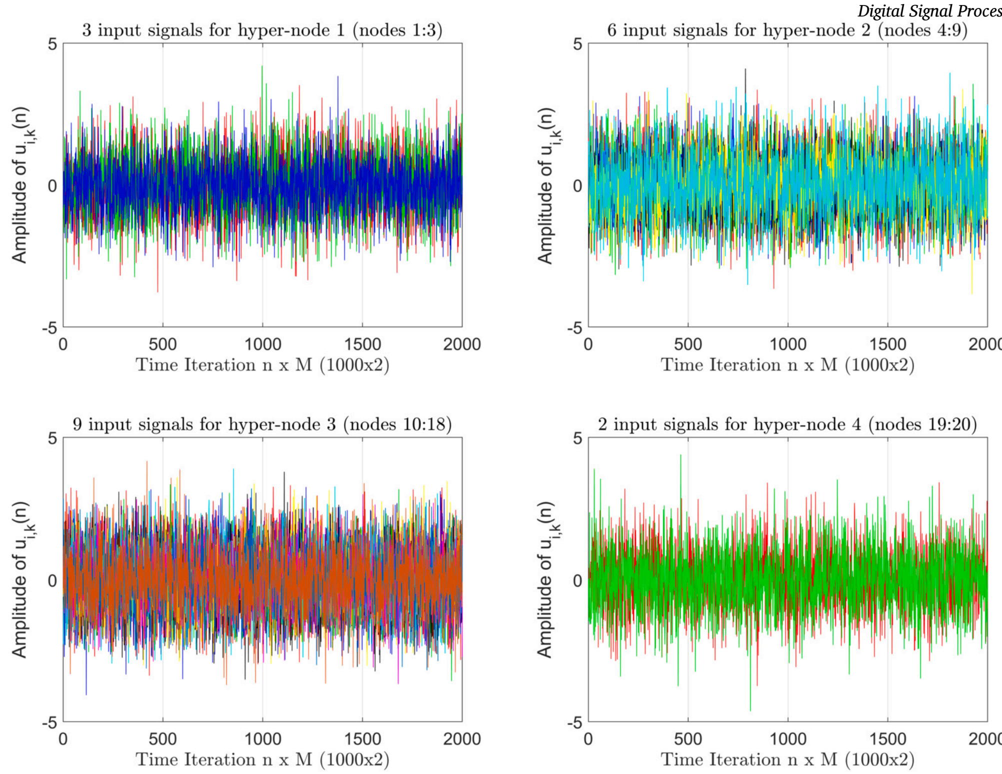


Fig. 4. The generated input signals $u_{i,k}(n)$ for all nodes displayed for 4 hyper-nodes individually.

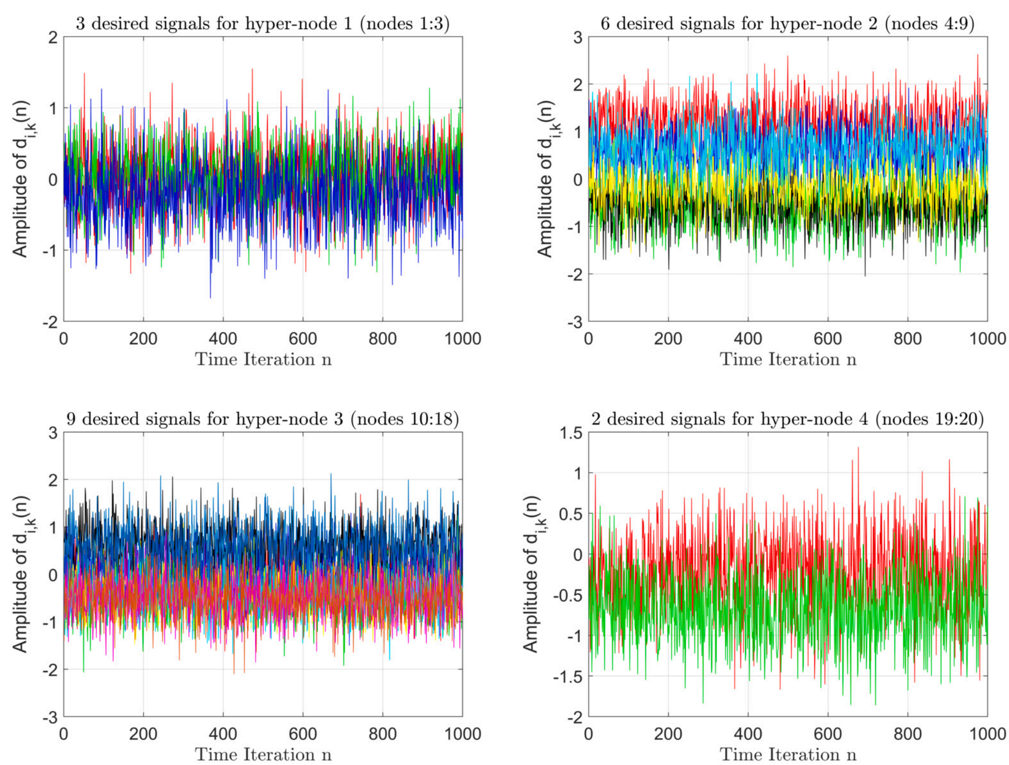


Fig. 5. The generated desired signals $d_{i,k}(n)$ for all nodes displayed for 4 hyper-nodes individually.

The two parameter vector estimation algorithms were taken into account to compare with our proposed method: 1) non-cooperative LMS algorithm, 2) conventional DLMS with a fixed combination matrix $\boldsymbol{A}$ that is constant in time, and 3) the proposed hyper-network. To assess the algorithms' performance, we randomly generated a set of sample signals $\{\boldsymbol{u}_{i,k}(n), d_{i,k}(n)\}$ and created the dataset, and carried out a preliminary evaluation. The algorithms were programmed in Matlab and executed on a computer with an Intel Core i7-12700H CPU. The step

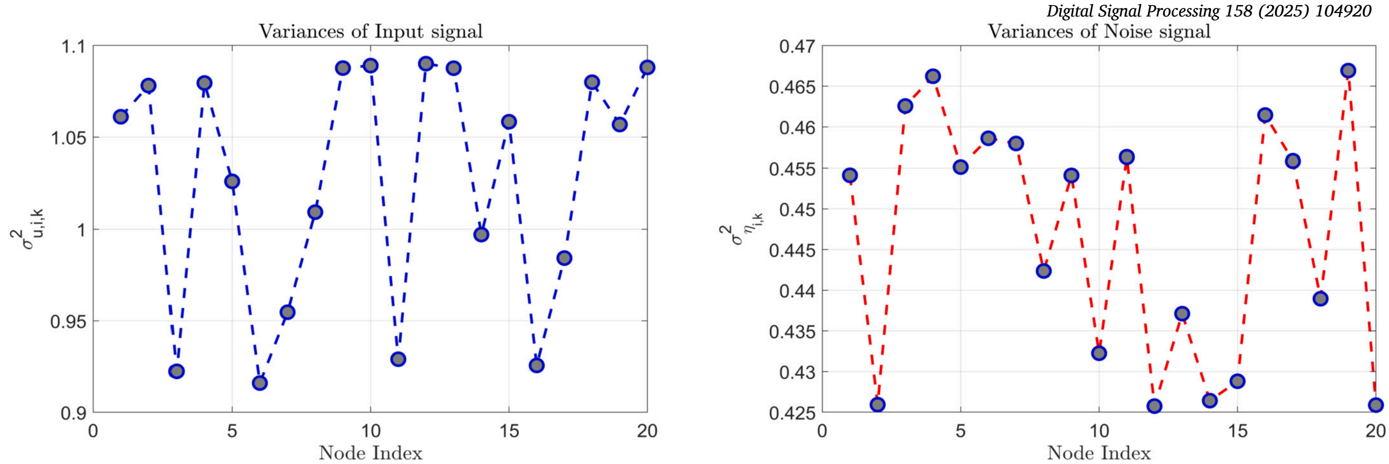


Fig. 6. The variances of the input signal and noise for each node are assumed to remain constant over time but can vary between different nodes.

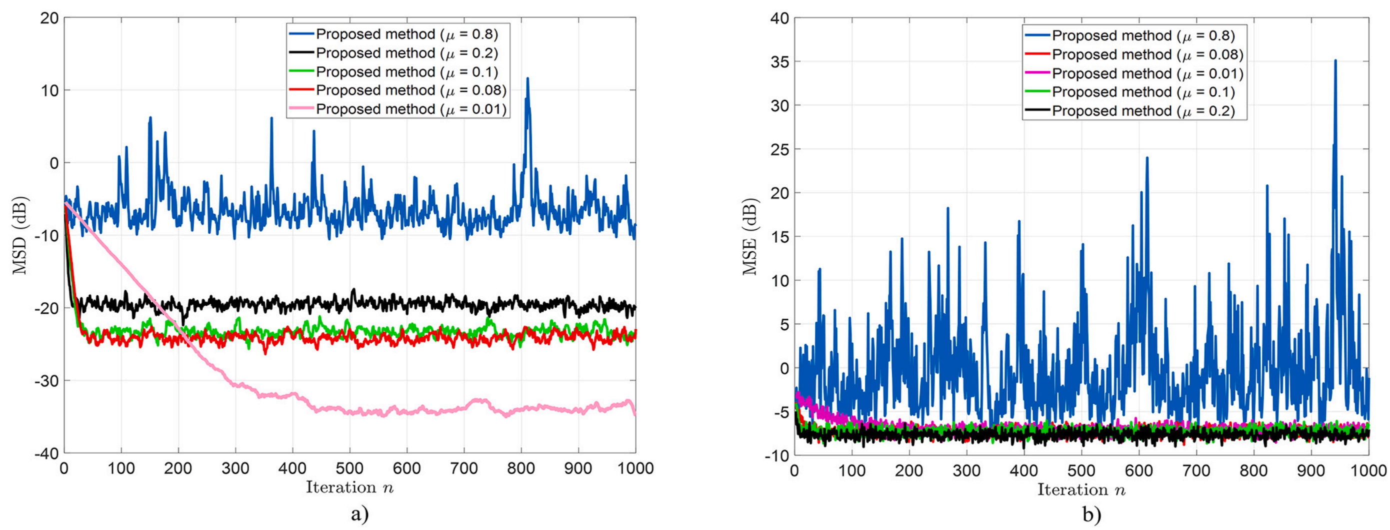


Fig. 7. The effect of the step-size values $\mu_{i,k}$ over convergence in terms of a) MSD and b) MSE curves for the proposed method.

size $\mu_{i,k}$ was fixed at 0.01 for every node. Also, in our simulations, we used two criteria, i.e., mean square deviation (MSD) and mean square error (MSE), which are:

$$MSD^{Hyper-net} = \frac{1}{N_T}\sum_{k=1}^{N_T} MSD_k \tag{49}$$

where $MSD_k = E||\boldsymbol{\theta}_{i,k} - \boldsymbol{\theta}_i^*||^2$ is according to k-th node. And for next, we have:

$$MSE^{Hyper-net} = \frac{1}{N_T}\sum_{k=1}^{N_T} MSE_k \tag{50}$$

where $MSE_k = E||d_{i,k}(n) - \boldsymbol{u}_{i,k}(n)\boldsymbol{\theta}_{i,k}||^2$.

To assess how the parameters influence the performance of the proposed framework, we designed two experiments, the outputs of which are shown in Figs. 7 and 8. At first, we examined the effect of step-size $\mu_{i,k}$ on the speed of convergence in terms of MSD and MSE. As can be seen in Fig. 7 a), in the case where the value of this parameter is considered the most significant value (i.e., 0.8), the MSD converges faster; however, it can be seen that this convergence is accompanied by an error and leads to a specified number of convergence has not occurred. As the value of this parameter decreased, although the convergence occurred slower, the MSD curves became stable. We also checked the values less than 0.01 for $\mu_{i,k}$, but because the performance of the suggested framework was not good in those values, we considered them unnecessary. Based on these evaluations, the best MSD convergence occurs at 0.01. Based on this, we set the step-size in this value in the continuation of the tests.

Also, as shown in Fig. 7 b), the larger values of step-size destroy the convergence condition, and the MSE diagram of other values considered converge to almost the same values. In the next test, we investigated the effect of the target parameter vector size on the convergence of the proposed framework in terms of MSD and MSE. In this experiment, we evaluated three vectors with dimensions 2×1, 3×1, and 8×1. We can see that when the size of the parameter vector is smaller, convergence occurs sooner, and the graph converges with a smaller amount of error for both MSD and MSE. For this purpose, in other tests, we considered the vector size of the target parameter to be 2×1.

We assess how well our method works by measuring its MSD. We set the value of M to 2 for a hyper-network consisting of 20 nodes distributed across four hyper-nodes. The parameters of other compared algorithms remain unchanged from those used in our simulations. The convergence curves illustrating the MSD are presented in Fig. 9. It is evident that our proposed method performs favorably in this assessment compared with non-cooperative LMS, conventional DLMS, Multitask DLMS [44], and Group DLMS [45].

Fig. 10 shows the effect of different noise powers on the convergence speed of the new framework, which is displayed in terms of MSD and MSE criteria. To obtain these graphs, the noise power was considered equal to $\sigma^2_{\eta_{i,k}} = [0, 0.01, 0.05, 0.1, 0.2, 0.5, 1]$. In this simulation, the target

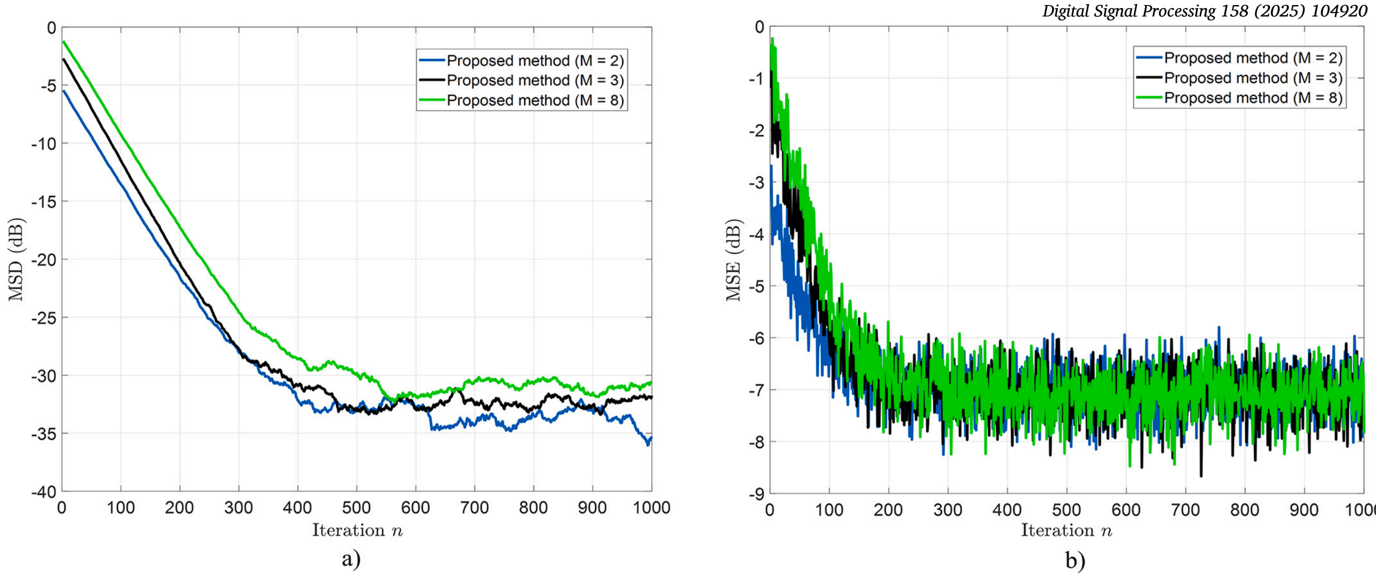


Fig. 8. The effect of the M values (the size of parameter vector) over convergence in terms of a) MSD and b) MSE curves for the proposed method.

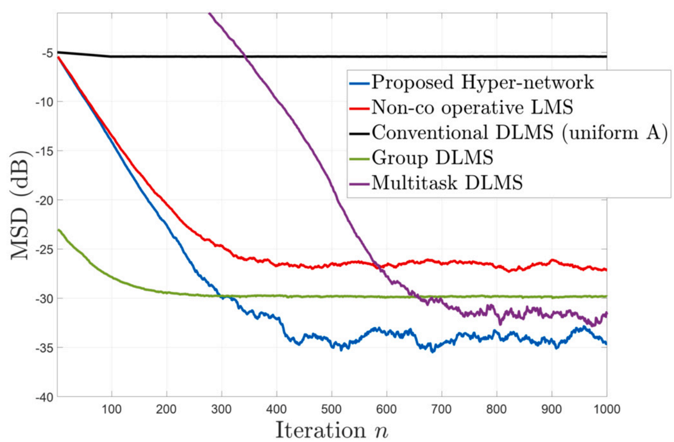


Fig. 9. Convergence curves in terms of MSD for the proposed method and other compared methods.

parameter vector $\boldsymbol{\theta}_i^*$ is considered equal to what is given in equation (48). Also, the movement step is set to $\mu_{i,k} = 0.01$. In both sets of curves, it can be seen that with the increase of noise variance, the speed of convergence decreases and also the curves converge to higher values of MSD and MSE. Also, the noise effect of the values obtained for $\boldsymbol{\theta}_i$ in (48), in different iterations, is shown in Fig. 11. Each row of this figure corresponds to one hyper-node and each column corresponds to one variance of the noise. In this figure, the effect of 4 different levels of noise power $\sigma^2_{\eta_{i,k}} = [0, 0.05, 0.5, 1]$ on the values obtained for $\boldsymbol{\theta}_i$ over time and its convergence to optimal values is shown. As can be seen, for example, in hyper-node 1, with the increase of noise power, the values obtained for $\boldsymbol{\theta}_i$ are associated with distortions compared to the optimal value $\boldsymbol{\theta}_i^*$. However, over time, these distortions decrease, and the curve converges to the optimal value $\boldsymbol{\theta}_i^*$. Optimum values are displayed in each figure with parallel black lines.

We then compared the computational complexity of the proposed method with the LMS and DLMS methods and summarized the results in Table 2. It can be seen that the proposed method for each internal node of each hyper node has $M(|N_{i,k}|)$ multiplications and $M(|N_{i,k}| - 1)$ additions.

4. Conclusion

Accurately adaptive processing is a significant challenge when dealing with various types of data, such as medical signals for diagnosing diseases, audio signals for categorizing audio sources, and telecommunication signals for separating transmitters. Different approaches have been proposed by researchers to tackle this issue, but these adaptive networks still have their limitations in accurately processing certain types of data. This study proposes a new framework that uses an adaptive nature based on the new hyper-network concept. The proposed framework takes a network-based approach using filters, resulting in higher accuracy than existing techniques. Additionally, the mean and mean square convergence of the proposed framework have been analyzed theoretically. To assess the performance of the proposed framework, experiments were conducted and compared to several compared methods in the field. The obtained results affirm the effectiveness of the proposed framework and highlight its potential when dealing with various types of data. One of the future applications for the proposed hyper-network is the processing of multi-channel signals such as multi-channel EEG data, audio signal obtained from a set of microphones, etc.

Table 2
Computational complexity for the proposed algorithms and other conventional algorithms.

Methods	Adders number	Multipliers number
LMS	$2M$	$2M$
DLMS	$M(\|N_k\| - 1)$	$M(\|N_k\|)$
Proposed Method	$M(\|N_{i,k}\| - 1)$	$M(\|N_{i,k}\|)$

CRediT authorship contribution statement

Tahereh Bahraini: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Mohammad B. Shamsollahi:** Conceptualization, Investigation, Methodology, Supervision, Validation, Writing – review & editing. **Fatemeh Afkhaminia:** Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Tahereh Bahraini reports financial support was provided by Iran National Science Foundation, with the grant number: 4020598. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

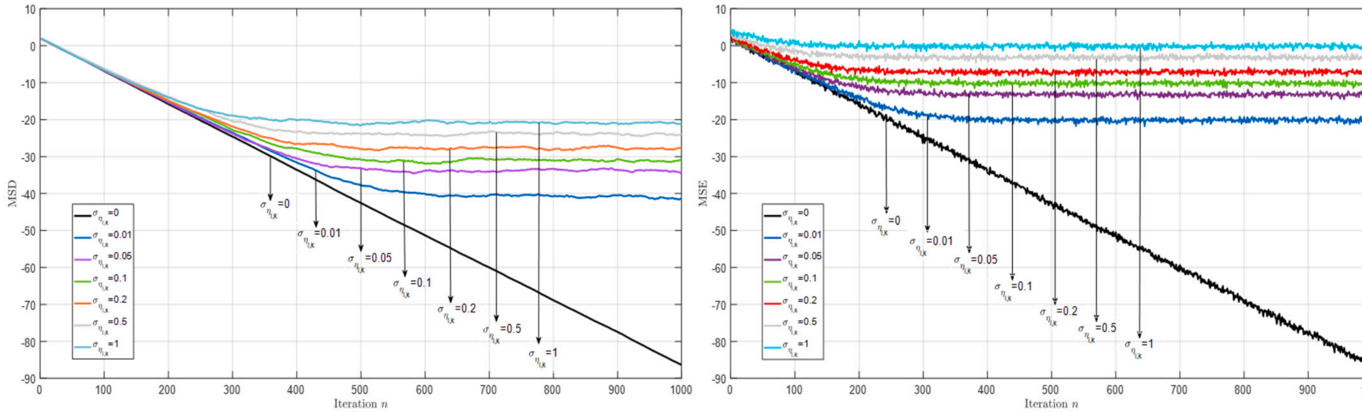


Fig. 10. The impact of varying noise power on the convergence of the proposed method, based on the MSD and MSE.

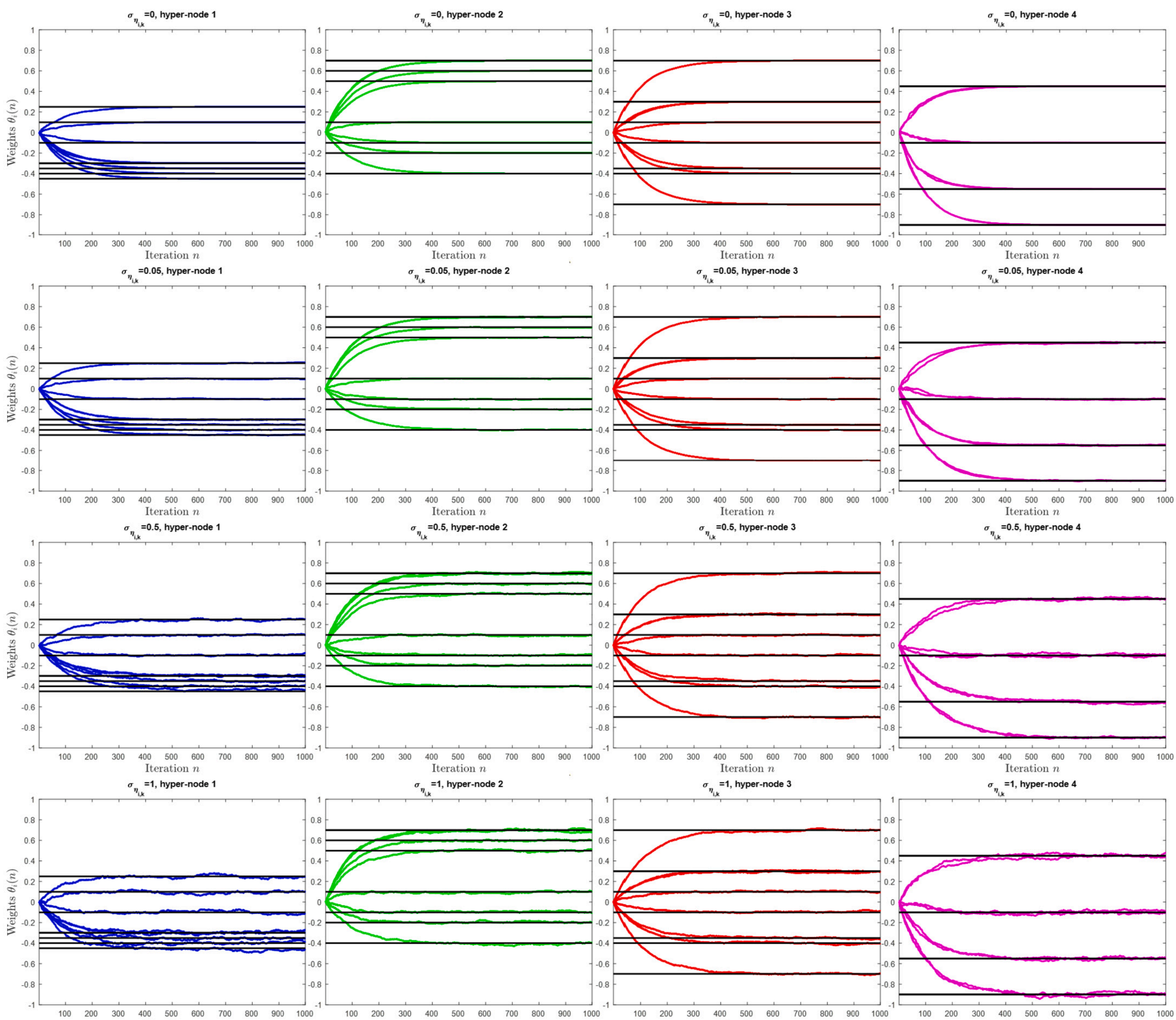


Fig. 11. The effect of different power of noise on the θ_i for the new framework.

Acknowledgments

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

First author reports financial support was provided by Iran National Science Foundation, with the grant number: 4020598 and administrative support was provided by Sharif University of Technology. Other authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] K.A. Lee, W.S. Gan, Inherent decorrelating and least perturbation properties of the normalized subband adaptive filter, IEEE Trans. Signal Process. 54 (11) (2006) 4475–4480.
[2] H. Wang, D. Wang, Neural cascade architecture with triple-domain loss for speech enhancement, IEEE/ACM Trans. Audio Speech Lang. Process. 30 (2021) 734–743.
[3] J. Arenas-García, A.R. Figueiras-Vidal, Adaptive combination of proportionate filters for sparse echo cancellation, IEEE Trans. Audio Speech Lang. Process. 17 (6) (2009) 1087–1098.
[4] R.M. Pimenta, M.R. Petraglia, D.B. Haddad, Stability analysis of the bias compensated lms algorithm, Digit. Signal Process. 147 (2024) 104395.
[5] H.-N. Lee, G.J. Pottie, Fast adaptive equalization/diversity combining for time-varying dispersive channels, IEEE Trans. Commun. 46 (9) (1998) 1146–1162.
[6] A.H. Sayed, Adaptive networks, Proc. IEEE 102 (4) (2014) 460–497.
[7] T. Bahraini, A.N. Sadigh, Proposing a robust rls based subband adaptive filtering for audio noise cancellation, Appl. Acoust. 216 (2024) 109755.
[8] J. Tsitsiklis, M. Athans, Convergence and asymptotic agreement in distributed decision problems, IEEE Trans. Autom. Control 29 (1) (1984) 42–50.
[9] L. Xiao, S. Boyd, Fast linear iterations for distributed averaging, Syst. Control Lett. 53 (1) (2004) 65–78.
[10] P. Braca, S. Marano, V. Matta, Enforcing consensus while monitoring the environment in wireless sensor networks, IEEE Trans. Signal Process. 56 (7) (2008) 3375–3380.
[11] A. Nedic, A. Ozdaglar, Distributed subgradient methods for multi-agent optimization, IEEE Trans. Autom. Control 54 (1) (2009) 48–61.
[12] S. Kar, J.M. Moura, Distributed consensus algorithms in sensor networks: link failures and channel noise, arXiv preprint arXiv:0711.3915.
[13] S.-Y. Tu, A.H. Sayed, Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks, IEEE Trans. Signal Process. 60 (12) (2012) 6217–6234.
[14] A.G. Dimakis, S. Kar, J.M. Moura, M.G. Rabbat, A. Scaglione, Gossip algorithms for distributed signal processing, Proc. IEEE 98 (11) (2010) 1847–1864.
[15] K. Srivastava, A. Nedic, Distributed asynchronous constrained stochastic optimization, IEEE J. Sel. Top. Signal Process. 5 (4) (2011) 772–790.
[16] D.P. Bertsekas, A new class of incremental gradient methods for least squares problems, SIAM J. Optim. 7 (4) (1997) 913–926.
[17] A. Nedic, D.P. Bertsekas, Incremental subgradient methods for nondifferentiable optimization, SIAM J. Optim. 12 (1) (2001) 109–138.
[18] M.G. Rabbat, R.D. Nowak, Quantized incremental algorithms for distributed optimization, IEEE J. Sel. Areas Commun. 23 (4) (2005) 798–808.
[19] D. Blatt, A.O. Hero, H. Gauchman, A convergent incremental gradient method with a constant step size, SIAM J. Optim. 18 (1) (2007) 29–51.
[20] C.G. Lopes, A.H. Sayed, Incremental adaptive strategies over distributed networks, IEEE Trans. Signal Process. 55 (8) (2007) 4064–4077.
[21] C.G. Lopes, A.H. Sayed, Diffusion least-mean squares over adaptive networks: formulation and performance analysis, IEEE Trans. Signal Process. 56 (7) (2008) 3122–3136.
[22] F.S. Cattivelli, A.H. Sayed, Diffusion lms strategies for distributed estimation, IEEE Trans. Signal Process. 58 (3) (2009) 1035–1048.
[23] D.G. Tiglea, R. Candido, M.T. Silva, An adaptive algorithm for sampling over diffusion networks with dynamic parameter tuning and change detection mechanisms, Digit. Signal Process. 127 (2022) 103587.
[24] J. Chen, A.H. Sayed, Diffusion adaptation strategies for distributed optimization and learning over networks, IEEE Trans. Signal Process. 60 (8) (2012) 4289–4305.
[25] X. Zhao, S.-Y. Tu, A.H. Sayed, Diffusion adaptation over networks under imperfect information exchange and non-stationary data, IEEE Trans. Signal Process. 60 (7) (2012) 3460–3475.
[26] A. Khalili, M.A. Tinati, A. Rastegarnia, J.A. Chambers, Steady-state analysis of diffusion lms adaptive networks with noisy links, IEEE Trans. Signal Process. 60 (2) (2011) 974–979.
[27] J. Chen, A.H. Sayed, Distributed Pareto optimization via diffusion strategies, IEEE J. Sel. Top. Signal Process. 7 (2) (2013) 205–220.
[28] O.N. Gharehshiran, V. Krishnamurthy, G. Yin, Distributed energy-aware diffusion least mean squares: game-theoretic learning, IEEE J. Sel. Top. Signal Process. 7 (5) (2013) 821–836.
[29] S. Zandi, M. Korki, Differential privacy for diffusion lms algorithm over a distributed network, Digit. Signal Process. (2024) 104634.
[30] A.H. Sayed, et al., Adaptation, learning, and optimization over networks, Found. Trends Mach. Learn. 7 (4–5) (2014) 311–801.
[31] Y. Liu, C. Li, Z. Zhang, Diffusion sparse least-mean squares over networks, IEEE Trans. Signal Process. 60 (8) (2012) 4480–4485.
[32] S. Chouvardas, K. Slavakis, Y. Kopsinis, S. Theodoridis, A sparsity promoting adaptive algorithm for distributed learning, IEEE Trans. Signal Process. 60 (10) (2012) 5412–5425.
[33] P. Di Lorenzo, A.H. Sayed, Sparse distributed learning based on diffusion adaptation, IEEE Trans. Signal Process. 61 (6) (2012) 1419–1433.
[34] P. Chainais, C. Richard, Distributed dictionary learning over a sensor network, arXiv preprint arXiv:1304.3568.
[35] P. Chainais, C. Richard, Learning a common dictionary over a sensor network, in: 2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), IEEE, 2013, pp. 133–136.
[36] J.B. Predd, S.B. Kulkarni, H.V. Poor, Distributed learning in wireless sensor networks, IEEE Signal Process. Mag. 23 (4) (2006) 56–69.
[37] P. Honeine, M. Essoloh, C. Richard, H. Snoussi, Distributed regression in sensor networks with a reduced-order kernel model, in: IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference, IEEE, 2008, pp. 1–5.
[38] J. Chen, L. Tang, J. Liu, J. Ye, A convex formulation for learning shared structures from multiple tasks, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 137–144.
[39] O. Chapelle, P. Shivaswamy, S. Vadrevu, K. Weinberger, Y. Zhang, B. Tseng, Multi-task learning for boosting with application to web search ranking, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2010, pp. 1189–1198.
[40] J. Zhou, L. Yuan, J. Liu, J. Ye, A multi-task learning formulation for predicting disease progression, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 814–822.
[41] J. Chen, C. Richard, A.H. Sayed, Multitask diffusion adaptation over networks, IEEE Trans. Signal Process. 62 (16) (2014) 4129–4144.
[42] A.H. Sayed, Diffusion adaptation over networks, in: Academic Press Library in Signal Processing, vol. 3, Elsevier, 2014, pp. 323–453.
[43] F.S. Cattivelli, A.H. Sayed, Diffusion lms strategies for distributed estimation, IEEE Trans. Signal Process. 58 (3) (2009) 1035–1048.
[44] J. Chen, C. Richard, A.H. Sayed, Diffusion lms over multitask networks, IEEE Trans. Signal Process. 63 (11) (2015) 2733–2748.
[45] J. Chen, S.K. Ting, C. Richard, A.H. Sayed, Group diffusion lms, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2016, pp. 4925–4929.

Tahereh Bahraini received the PhD degree in Ferdowsi University of Mashhad (FUM), Mashhad, Iran, in 2023. She is currently a postdoctoral researcher at Sharif University of Technology. Her research interests include signal processing, pattern recognition, adaptive networks, statistical modeling, and optimization.

Mohammad Bagher Shamsollahi received M.Sc. degree in electrical engineering, telecommunications, from the Sharif University of Technology, Tehran, in 1991, and the Ph.D. degree in electrical engineering, biomedical signal processing, from the University of Rennes 1, Rennes, France, in 1997. He is currently a Professor with the Department of Electrical Engineering, Sharif University of Technology. His research interests include: biomedical signal processing, brain–computer interface, and time-scale and time–frequency signal processing.

Fatemeh Afkhaminia is currently a PhD student in Sharif University of Technology, Tehran, Iran, in the field of electrical engineering. She received MS. Degree in University of Technology Tabriz, Iran. Her research interest includes: antenna array, direction of arrival estimation, adaptive networks, statistical modeling, EEG signal processing, and optimization.