Talebi, Sayed Pouria; Werner, Stefan; Mandic, Danilo

Distributed Adaptive Filtering of α-Stable Signals

Published in:
IEEE Signal Processing Letters

DOI:
10.1109/LSP.2018.2862639

Published: 01/10/2018

Document Version
Peer reviewed version

Please cite the original version:

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.
Distributed Adaptive Filtering of $\alpha$-Stable Signals

Sayed Pouria Talebi, Stefan Werner, and Danilo P. Mandic

Abstract—A cost-effective framework for distributed adaptive filtering of $\alpha$-stable signals over sensor networks is proposed. First, the filtering paradigm of $\alpha$-stable signals through multiple observations made over a network of sensors is revisited and an optimal solution is formulated. Then, an adaptive gradient descent based algorithm for distributed real-time filtering of $\alpha$-stable signals via multi-agent networks is derived. This not only provides an approximation of the formulated optimal solution, but also a cost-effective algorithm which scales with the size of the network. Moreover, performance of the derived algorithm is analyzed and convergence conditions are established.

Index Terms—Sensor networks, distributed adaptive filtering, consensus fusion, fractional differential, $\alpha$-stable random signals.

I. INTRODUCTION

Most signal processing and machine learning approaches, assume a Gaussian model for the signal, as this often leads to mathematically tractable and computationally efficient solutions. However, in an increasing number of modern applications, the encountered signals exhibit sharp spikes, resulting in distributions that decay slower than the Gaussian case [1]–[8]. For modeling such signals, $\alpha$-stable random processes have proven to be advantageous [3,6,7].

A general closed-form expression for the probability distribution function of the entire class of $\alpha$-stable random vectors does not exist. However, the class of real-valued $\alpha$-stable random vectors with elliptically symmetric distributions, referred to as SoS, admit characteristic function of the form [1,9]

$$\Phi_Z(s) = \mathbb{E}\left\{ e^{is^Tz} \right\} = e^{is^T\xi - (\frac{1}{2 \alpha}^T \Gamma_\alpha s)^\frac{2}{\alpha}}$$

(1)

where $i^2 = -1$, while $(\cdot)^T$ and $\mathbb{E}\{\cdot\}$ denote the transpose and statistical expectation operators, with $\Phi_Z(\cdot)$ denoting the characteristic function of the random vector $Z$, and the positive semi-definite covariance matrix $\Gamma_\alpha$ determining the elliptical shape of the distribution of $Z$ which is centered at $\xi$. The characteristic exponent $\alpha \in (0, 2]$ in (1) specifies tail heaviness of the density function [1,3]. A small value of $\alpha$ indicates severe impulsiveness, resulting in heavier tails, while a value close to 2 indicates a more Gaussian type behavior. Indeed when $\alpha = 2$, random vector $Z$ has a Gaussian distribution with covariance matrix $\Gamma_\alpha$ and mean vector $\xi$.

In addition to their trade-mark stability property [6], SoS random processes also admit a generalized version of the central limit theorem [2], which makes them attractive for modeling a large variety of signals. However, with the exception of the Gaussian case, SoS random processes only have finite statistical moments of orders strictly less than $\alpha$ [2]–[4]. Therefore, in filtering applications it is implicitly implied that $\alpha \in (1, 2]$, to ensure that finite conditional expectations do exist. Indeed, we limit our study to real-valued SoS processes with $\alpha \in (1, 2]$, where the mean vector corresponds to $\xi$ in (1).

Since out of the entire class of SoS random variables only the Gaussian case has well-defined variance and higher-order statistical moments, filtering techniques based on minimizing the second-order moment of an error measure, such as the least mean square (LMS), do not perform well when applied to the generality of SoS random processes [10]–[13]. The distributed particle filter in [13] addresses this issue; however, the solution is too cumbersome for many filtering applications. Although a number of gradient based techniques for filtering SoS signals have been proposed [11,14,15], a distributed approach suitable for implementation over a large-scale sensor network accompanied with a rigorous convergence and stability analysis is still lacking.

In recent years, fractional-order calculus is experiencing a renaissance, finding applications in statistics and control [16]–[18], where they have been used to derive fractional-order statistical moments and describe complex physical systems. However, the potential of fractional-order calculus in adaptive learning remains largely untapped.

In light of the advantages that distributed learning techniques offer [19]–[25], this work introduces a class of practical distributed adaptive filtering solutions for SoS signals. To this end, based on the characteristic function in (1), an optimal solution to such a problem is formulated. However, calculation of such a solution requires knowledge of the signal statistics and is impractical for implementation over large-scale networks. Therefore, an adaptive filtering algorithm, where agents of the network collectively minimize the fractional square of an error measure in a gradient descent manner using fractional-order differentials is derived. Moreover, performance of the derived algorithm is analyzed and convergence criteria are established. Finally, the introduced concept is validated through a simulation example.

Mathematical Notations: Scalars, column vectors, and matrices are denoted respectively by lowercase, bold lowercase, and bold uppercase letters. Multi-variate random processes are represented by uppercase bold italic letters. The transpose and statistical expectation operators are denoted by $(\cdot)^T$ and $\mathbb{E}\{\cdot\}$, while $\otimes$ denotes the Kronecker product. Finally, $(\cdot)^{(\ast)}$ denotes the element wise implementation of the function $f(z) = |z|^\tau \text{sign}(z)$, with $\text{sign}(\cdot)$ denoting the sign function.
II. PROBLEM FORMULATION

We consider the classical distributed filtering problem [23, 26]–[29] from the more general perspective of SoS random signals. This includes Gaussian random signals as a special case, for which $\alpha = 2$.

A. The Network

The sensor network is modeled as a connected graph $G = (\mathcal{N}, \mathcal{E})$, where the node set $\mathcal{N}$ represents agents of the sensor network, while the edge set $\mathcal{E}$ represents bidirectional communication links between agents of the sensor network. The neighborhood of a node, $l$, is denoted by $\mathcal{N}_l$ and is defined as the set of nodes that are connected to node $l$, including itself. The number of nodes in the set $\mathcal{N}_l$, i.e., its cardinality, is denoted by $|\mathcal{N}_l|$, with $|\mathcal{N}|$ denoting the total number of nodes in the network.

B. The Filtering Problem

The aim is collaborative estimation of a parameter matrix, $\mathbf{H}$, through observations made via sensors of the network. The observations and desired parameter vector are interrelated as

$$\forall l \in \mathcal{N}: \mathbf{y}_{l,n} = \mathbf{H}\mathbf{x}_{l,n} + \mathbf{w}_{l,n}$$

where, for node $l$ and time instant $n$, $\mathbf{y}_{l,n}$ and $\mathbf{w}_{l,n}$ denote the observation and background noise vectors, with $\mathbf{x}_{l,n}$ representing the regression vector used to identify the system. The regression vectors $\{\mathbf{x}_{l,n} : \forall l \in \mathcal{N}\}$ and background noise vectors $\{\mathbf{w}_{l,n} : \forall l \in \mathcal{N}\}$ are assumed to be temporally and spatially independent white zero-mean SoS random sequences.\(^1\)

III. OPTIMAL SOLUTION

In order to derive an optimal solution to the proposed filtering problem, the parameter matrix, $\mathbf{H}$, is first formulated in terms of the regression and observation vector statistics. To this end, the observation, regression, and background noise vectors are organized into the column vectors

$$\mathbf{y}_n = [\mathbf{y}_{1,n}^\top, \cdots, \mathbf{y}_{|\mathcal{N}|,n}]^\top$$
$$\mathbf{x}_n = [\mathbf{x}_{1,n}^\top, \cdots, \mathbf{x}_{|\mathcal{N}|,n}]^\top$$
$$\mathbf{w}_n = [\mathbf{w}_{1,n}^\top, \cdots, \mathbf{w}_{|\mathcal{N}|,n}]^\top$$

which allows the expression in (2) to be formulated from a network-wide perspective as

$$\mathbf{y}_n = (\mathbf{I} \otimes \mathbf{H}) \mathbf{x}_n + \mathbf{w}_n$$

where $\mathbf{I}$ is an $|\mathcal{N}| \times |\mathcal{N}|$ identity matrix.

Now, from (1) the joint characteristic function of $\mathbf{y}_n$ and $\mathbf{x}_n$ can be expressed as

$$\Phi_{\mathbf{Y},\mathbf{X}}(\mathbf{y}_n, \mathbf{x}_n) = \mathbb{E}\left\{e^{i\mathbf{s}_n^\top \mathbf{y}_n + \mathbf{s}_x^\top \mathbf{x}_n}\right\}$$

$$= \mathbb{E}\left\{e^{i\mathbf{s}_n^\top (\mathbf{I} \otimes \mathbf{H}) \mathbf{x}_n + \mathbf{s}_x^\top \mathbf{w}_n}\right\}$$

$$= \Phi_{\mathbf{X}}((\mathbf{I} \otimes \mathbf{H}) \mathbf{s}_x + \mathbf{s}_x) \Phi_{\mathbf{W}}(\mathbf{s}_y).$$

On the other hand, considering the general characteristic function of SoS random processes in (1) we have

$$\Phi_{\mathbf{X}}((\mathbf{I} \otimes \mathbf{H}) \mathbf{s}_x + \mathbf{s}_x) = e^{-\frac{1}{2}(\mathbf{G}_{\mathbf{x}} \mathbf{s}_x)}$$

where

$$\Theta_{\mathbf{Y}}(\mathbf{y}_n, \mathbf{s}_x) = \mathbf{s}_y^\top (\mathbf{I} \otimes \mathbf{H}) \Gamma_x (\mathbf{I} \otimes \mathbf{H})^\top \mathbf{s}_y + \mathbf{s}_x^\top \Gamma_x \mathbf{s}_x$$

$$+ \mathbf{s}_y^\top (\mathbf{I} \otimes \mathbf{H}) \Gamma_x \mathbf{s}_x + \mathbf{s}_x^\top \Gamma_x (\mathbf{I} \otimes \mathbf{H})^\top \mathbf{s}_y.$$  

In a similar fashion, we have

$$\Phi_{\mathbf{W}}(\mathbf{s}_y) = e^{-\frac{1}{2}(\mathbf{G}_{\mathbf{w}} \mathbf{s}_y)}$$

with $\Theta_{\mathbf{w}}(\mathbf{s}_y) = \mathbf{s}_y^\top \Gamma_{\mathbf{w}} \mathbf{s}_y$. (8)

From the expressions in (5)-(8), $\Phi_{\mathbf{Y},\mathbf{X}}(\mathbf{y}_n, \mathbf{s}_x)$ can be formulated as

$$\Phi_{\mathbf{Y},\mathbf{X}}(\mathbf{y}_n, \mathbf{s}_x) = e^{-\frac{1}{2}(\mathbf{G}_{\mathbf{y},\mathbf{w}}(\mathbf{s}_y, \mathbf{s}_x))}$$

with $\Theta_{\mathbf{y},\mathbf{x}}(\mathbf{y}_n, \mathbf{s}_x) = \Theta_{\mathbf{x}}^2(\mathbf{s}_y, \mathbf{s}_x) + \Theta_{\mathbf{w}}^2(\mathbf{s}_y)$ and

$$\Theta_{\mathbf{x}}(\mathbf{y}_n, \mathbf{s}_x) = \left[\mathbf{s}_y^\top \mathbf{s}_x^\top \Gamma_y \Gamma_{\mathbf{y},\mathbf{x}} \Gamma_x \right] \left[\mathbf{s}_y \mathbf{s}_x\right].$$

where $\Gamma_y = (\mathbf{I} \otimes \mathbf{H}) \Gamma_x (\mathbf{I} \otimes \mathbf{H})^\top$, $\Gamma_{\mathbf{y},\mathbf{x}} = \Gamma_x (\mathbf{I} \otimes \mathbf{H})^\top$, and $\Gamma_x = (\mathbf{I} \otimes \mathbf{H})\Gamma_x$. Therefore, it can be shown that

$$(\mathbf{I} \otimes \mathbf{H}) = \Gamma_{\mathbf{y},\mathbf{x}} \Gamma_x^{-1}.\)  

The temporal and spatial independent assumption of the regression and background noise sequences leads to $\Gamma_{\mathbf{y},\mathbf{x}}$ and $\Gamma_x$ being block diagonal matrices so that

$$\Gamma_{\mathbf{y},\mathbf{x}} = \text{block-diag}\{\Gamma_{\mathbf{y},\mathbf{x}} : \forall l \in \mathcal{N}\}$$

$$\Gamma_x = \text{block-diag}\{\Gamma_x : \forall l \in \mathcal{N}\}.$$

Finally, upon replacing (11) into (10) and summing the block diagonal elements we have

$$\mathbf{H} = \frac{1}{|\mathcal{N}|} \sum_{l \in \mathcal{N}} \Gamma_{\mathbf{y},\mathbf{x}} \Gamma_x^{-1}.$$

Akin to what was performed in [13], sufficient statistics for calculating the solution in (12) can be found from empirical estimates of characteristic functions of the regression and observation vectors. Thus the expression in (12) is considered as the optimal solution to the defined filtering problem. However, this approach is neither cost-effective nor is it suitable for implementation over a network. To this end, an adaptive gradient descent based technique is next developed.

IV. DISTRIBUTED ADAPTIVE APPROACH

Consider $\hat{\mathbf{y}}_{l,n}$, the estimate of $\mathbf{y}_{l,n}$, obtained through the strictly linear filter

$$\hat{\mathbf{y}}_{l,n} = \tilde{\mathbf{H}}_n \mathbf{x}_{l,n}$$

where $\tilde{\mathbf{H}}_n$ is the estimate of desired parameter matrix $\mathbf{H}$ at time instant $n$. The estimates $\{\tilde{\mathbf{H}}_n : n = 1, 2, \ldots\}$ are selected to iteratively minimize expected value of the cost function

$$J_n = \frac{1}{|\mathcal{N}|} \sum_{l \in \mathcal{N}} \epsilon_{l,n}^T \epsilon_{l,n}$$

with $\epsilon_{l,n} = \mathbf{y}_{l,n} - \hat{\mathbf{y}}_{l,n}$, where $1 < \alpha' < \alpha$.\(^2\)
Remark 1. The fractional order of “$\alpha’-1$” ensures that the cost function, $J_n$, retains a convex shape. This also ensures that the terms $\forall l \in \mathcal{N} : \epsilon_{l,n}^{(\alpha’-1)}$ have finite statistical expectations.

At each time instant, the estimate of the parameter matrix is updated in a steepest descent manner expressed as

$$\hat{H}_{n+1} = \hat{H}_n - \eta \nabla^{\alpha’-1} J_n$$

where $\eta$ is a positive real-valued gain and $\nabla^{\alpha’}$ denotes the $(\alpha’-1)$-order gradient operator.² Adopting the framework introduced in [30,31] for calculating fractional differentials, the update term in (15) yields

$$\hat{H}_{n+1} = \hat{H}_n + \eta \sum_{\forall l \in \mathcal{N}} \epsilon_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T = \frac{1}{|\mathcal{N}|} \sum_{\forall l \in \mathcal{N}} \Phi_{l,n}$$

(16)

where constant multiplicative terms are absorbed into $\eta$, while

$$\Phi_{l,n} = \hat{H}_n + \mu \epsilon_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T$$

represents the intermediate estimate of $\mathbf{H}$ at time instant $n$ at node $l$, whereas $\mu = \eta/|\mathcal{N}|$ is an adaptation gain.

The average in (16) can be calculated in a distributed manner using an average consensus filter (ACF) [32,33]. In the context of this work, the state of the ACF after $\kappa$ iterations is expressed as

$$\mathbf{F}_{i,(\kappa)} = \mathbf{F}_{i,(\kappa-1)} + \sum_{\forall j \in \mathcal{N}_i} m_{i,j} \left( \mathbf{F}_{j,(\kappa-1)} - \mathbf{F}_{i,(\kappa-1)} \right)$$

(18)

where $\mathbf{F}_{i,(\kappa)}$ is the state of the filter at node $i$ after $\kappa$ iterations and $m_{i,j}$ denotes a positive real-valued weight. This can be expressed in a more mathematically convenient fashion as

$$\mathbf{F}_{(\kappa)} = (\mathbf{M} \otimes \mathbf{I}) \mathbf{F}_{(\kappa-1)} = (\mathbf{M}^\kappa \otimes \mathbf{I}) \mathbf{F}_{(0)}$$

(19)

where $\mathbf{F}_{(\kappa)} = [\mathbf{F}_{1,(\kappa)}, \ldots, \mathbf{F}_{|\mathcal{N}|,(\kappa)}]^T$ whereas $\mathbf{I}$ is an identity matrix of appropriate size, while the element on the $i^{th}$ row and $j^{th}$ column of $\mathbf{M}$ is selected so that

$$\mathbf{M}_{i,j} = \left\{ \begin{array}{ll} 1 + m_{i,i} & \text{if } i = j, \\ m_{i,j} & \text{if } i \in \mathcal{N}_j \setminus j, \\ 0 & \text{otherwise}. \end{array} \right.$$ 

Furthermore, assuming the weights are selected so that $\mathbf{M}$ is also doubly stochastic and meets the conditions in [32], from the work in [32,33], we have

$$\lim_{\kappa \to \infty} \mathbf{F}_{i,(\kappa)} = \frac{1}{|\mathcal{N}|} \sum_{\forall \mathbf{F}_{j,(0)}} \mathbf{F}_{j,(0)}$$

(20)

which is the ACF required to calculate the averages in (16). The operation of the ACF at node $i$ after $\kappa$ iterations is represented via the following schematic

$$\mathbf{F}_{i,(\kappa)} \leftarrow \text{ACF} \leftarrow \{ \forall j \in \mathcal{N} : \mathbf{F}_{j,(0)} \}. $$

Algorithm 1. Distributed Fractional Least Mean Square

For nodes $l = \{1, \ldots, |\mathcal{N}|\}$:

**Estimate Filter Output & Error Term**

$$\hat{y}_{l,n} = \mathbf{H}_{l,n} \mathbf{x}_{l,n}$$

$$\epsilon_{l,n} = y_{l,n} - \hat{y}_{l,n}$$

**Perform Local Update**

$$\Phi_{l,n} = \mathbf{H}_{l,n} + \mu \epsilon_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T$$

(23)

**Perform Consensus Update**

$$\mathbf{H}_{l,n+1} \leftarrow \text{ACF} \leftarrow \{ \forall i \in \mathcal{N} : \Phi_{i,n} \}$$

(24)

V. CONVERGENCE AND STABILITY ANALYSIS

From the error term defined in (14) we have

$$\epsilon_{l,n} = y_{l,n} - \hat{y}_{l,n} = \left( \mathbf{H} - \hat{\mathbf{H}}_{l,n} \right) \mathbf{x}_{l,n} + \mathbf{w}_{l,n}.$$ 

(25)

Now, substitution of $\epsilon_{l,n}$ from (25) into (23) yields

$$\Phi_{l,n} = \mathbf{H}_{l,n} + \mu \epsilon_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T = \mathbf{H}_{l,n} + \mu \epsilon_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T.$$ 

(26)

Subsequently, considering $\Phi_{l,n}$ in the formulation in (26) gives

$$\mathbf{H} - \Phi_{l,n} = \mathbf{Y}_{l,n} \left( \mathbf{I} - \mu \mathbf{x}_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T \right) - \mu \mathbf{w}_{l,n} \left( x_{l,n}^{(\alpha’-1)} \right)^T.$$ 

(27)

From (24) and (27), the error of the estimated parameter matrix can be expressed in a regressive fashion as

$$\mathbf{Y}_{l,n+1} \leftarrow \text{ACF} \leftarrow \{ \forall i \in \mathcal{N} : \mathbf{H} - \Phi_{i,n} \}.$$ 

(28)

Replacing the expression in (19) as the ACF in (28) yields

$$\mathbf{E}_{n+1} = \mathbf{A}_n \mathbf{E}_n - \mu (\mathbf{M}^\kappa \otimes \mathbf{I}) \mathbf{Q}_n$$

(29)

where $\mathbf{A}_n = (\mathbf{M}^\kappa \otimes \mathbf{I}) (\mathbf{I} \otimes \mathbf{I} - \mu \mathbf{X}_n)$, while

$$\mathbf{E}_n = [\mathbf{T}_{1,n,n}, \ldots, \mathbf{T}_{|\mathcal{N}|,n,n}]^T$$

$$\mathbf{X}_n = \text{block-diag}\{ x_{l,n}^{(\alpha’-1)} \mathbf{x}_{l,n}^T : \forall l \in \mathcal{N} \}$$

$$\mathbf{Q}_n = [Q^T_{1,n}, \ldots, Q^T_{|\mathcal{N}|,n}]^T$$

with $Q_{l,n} = x_{l,n}^{(\alpha’-1)} \mathbf{w}_{l,n}^T$.

Alternatively, (29) can be formulated as

$$\text{vec}(\mathbf{E}_{n+1}) = (\mathbf{I} \otimes \mathbf{A}_n) \text{vec}(\mathbf{E}_n) - \mu \text{vec}( (\mathbf{M}^\kappa \otimes \mathbf{I}) \mathbf{Q}_n )$$

(30)

where vec($\cdot$) transforms a matrix into a column vector by stacking its columns.³

³The effect of network topology and selection of ACF parameters, i.e., number of iterations and weights, on the estimation error dynamics are indicated by $\mathbf{M}^\kappa$. 
Considering the expressions in (29) and (30), for the case of $1 < p < \alpha$, $\|\text{vec}(\mathbf{E})\|_p^p$ will converge if the eigenvalues of $\mathbf{A}_n$ lie within the unit circle. In statistical terms, $\mathbb{E}\left\{\|\text{vec}(\mathbf{E}_n)\|_p^p\right\}$ will converge to a stabilizing solution if the eigenvalues of $\mathbb{E}\{\mathbf{A}_n\}$ lie within the unit circle. Given that

$$\mathbb{E}\{\mathbf{A}_n\} = (\mathbf{M}^\kappa \otimes \mathbf{I}) (\mathbf{I} \otimes \mathbf{I} - \mu \mathbb{E}\{\mathbf{X}_n\})$$

and $\mathbf{M}$ was assumed doubly stochastic, resulting in the eigenvalues of $\mathbf{M}^\kappa$ lying on or within the unit circle; then, the sufficient condition for convergence is given by

$$\mu < \frac{1}{\lambda_{\text{max}}(\mathbb{E}\{\mathbf{X}_n\})}$$

(31)

where $\lambda_{\text{max}}(\mathbb{E}\{\mathbf{X}_n\})$ is the largest eigenvalue of $\mathbb{E}\{\mathbf{X}_n\}$.

**Remark 2.** The criterion $1 < p < \alpha$ guarantees that all statistical expectations exist and are finite.

**Remark 3.** If statistics of the regression vectors are not available, a straightforward technique for guaranteeing convergence would be to normalize the adaptation step-size at each time instant, that is, to replace (23) from Algorithm 1 with

$$\Phi_{l,n} = \hat{\mathbf{H}}_{l,n} + \mu \epsilon_{l,n} (\mathbf{x}_{l,n}' \mathbf{x}_{l,n}^{-1})^T (\mathbf{x}_{l,n}' \mathbf{x}_{l,n}^{-1})^{-1}$$

(32)

where $\mu < 1$ will guarantee convergence.

**Remark 4.** Taking into account that the background noise and regression processes were assumed to be temporally and spatially independent white zero-mean SoS sequences, taking the statistical expectation of (29) yields

$$\mathbb{E}\{\mathbf{E}_{n+1}\} = (\mathbf{M}^\kappa \otimes \mathbf{I}) (\mathbf{I} \otimes \mathbf{I} - \mu \mathbb{E}\{\mathbf{X}_n\}) \mathbb{E}\{\mathbf{E}_n\}.$$  

(33)

Therefore, if the condition in (31) is satisfied, any misadjustment in initialization values decreases exponentially fast.

**VI. Performance Evaluation**

In order to demonstrate the performance of the proposed algorithm, the filtering problem defined in Section II-B was considered, where $\alpha = 1.7$, whereas the parameter matrix was

$$\mathbf{H} = \begin{bmatrix} 1 & 0.5 & -1.5 \\ 0.6 & 1 & -0.6 \end{bmatrix}$$

while the covariation matrices of the regression and background noise sequences were

$$\mathbf{\Gamma}_\mathbf{r} = \mathbf{I} \otimes \begin{bmatrix} 1 & 0.2 & 0.2 \\ 0.2 & 2 & 0.5 \\ 0.2 & 0.5 & 1 \end{bmatrix}$$

and $\mathbf{\Gamma}_w = 10^{-2} \times \mathbf{I} \otimes \begin{bmatrix} 1 & 0.1 \\ 0.1 & 1 \end{bmatrix}$

with the adaptation gain set to $\mu = 0.01$ and $\alpha' = 1.6$. The network with $|\mathcal{N}| = 20$ and the topology shown in Figure 1 was used to estimate the parameter matrix.

Two performance indicators were considered: i) the mean absolute error (MAE) term at each node, that is, $\mathbb{E}\{|\epsilon_{l,n}\}|$; ii) the mean absolute deviation (MAD) term at each node, that is, $\mathbb{E}\{\mathbf{H} - \hat{\mathbf{H}}_{l,n}\}$. The mean-values were calculated empirically via averaging of the results obtained from $10^3$ independent realizations of the experiment. In addition, at each time instant the ACF was iterated $\kappa = 4$ times to approximate the required averages and the initial value for the parameter matrix estimate was $\forall l \in \mathcal{N} : \hat{\mathbf{H}}_{l,0} = 0$.

In Figure 2, the performance of the developed DFLMS is compared to that of the traditional DLMS in [35]. Observe that the proposed DFLMS, both in its formulation in Algorithm 1 and with the normalized step-size given in (32), performed well in a 1,7-stable environment and achieved lower MAE and MAD than the DLMS. In addition, the DFLMS framework outperformed the DLMS in terms of convergence rate. Finally, notice that the DLMS exhibited large jitters (sharp spikes) in its MAE (cf. MAD) behavior as a result of regression and background noise sequences that were heavy-tailed, whereas such behavior was greatly attenuated in the developed DFLMS framework. To enable a fair performance comparison, for DFLMS with normalized step-size, adaptation gain was chosen as $\mu = 0.08$ to achieve a similar convergence rate to that of the DFLMS in Algorithm 1.

![Figure 1. The sensor network with 20 nodes and 48 edges used in simulations.](image1)

![Figure 2. The MAE (top) and MAD (bottom) performance of the DFLMS and DLMS. Performance of the DFLMS across all nodes lies within the region in green, while performance of the DFLMS with normalized step-size lies within the region in red, whereas performance of the DLMS lies within the region in blue.](image2)

**VII. Conclusion**

Adaptive filtering of real-valued SoS signals over sensor networks has been considered where, based on the characteristic function of such signals, an optimal filtering solution has been derived. In addition, an algorithm for filtering SoS signals in a distributed fashion, based on minimizing real-time fractional powers of an error measure using differentials of order “$\alpha' - 1$”, where $1 < \alpha' < \alpha$, has been proposed. The performance of the proposed distributed filtering algorithm has been analyzed and convergence bound on the adaptation gain has been established.
REFERENCES


