Topology-Independent Distributed Adaptive Node-Specific Signal Estimation in Wireless Sensor Networks

Joseph Szurley, Alexander Bertrand, *Senior Member, IEEE,* and Marc Moonen, *Fellow, IEEE*

Abstract—**A topology-independent distributed adaptive node-specific signal estimation (TI-DANSE) algorithm is presented where each node of a wireless sensor network (WSN) is tasked with estimating a node-specific desired signal. To reduce the amount of data exchange, each node applies a linear compression to its sensors signal observations, and only transmits the compressed observations to its neighbors. The TI-DANSE algorithm is shown to converge to the same optimal node-specific signal estimates as if each node were to transmit its raw (uncompressed) sensor signal observations to every other node in the WSN. The TI-DANSE algorithm is first introduced in a fully connected WSN and then shown, in fact, to have the same convergence properties in any topology. When implemented in other topologies, the nodes rely on an in-network summation of the transmitted compressed observations that can be accomplished by various means. We propose a method for this in-network summation via a data-driven signal flow that takes place on a tree, where the topology of the tree may change in each iteration. This makes the algorithm less sensitive to link failures and applicable to WSNs with dynamic topologies.**

Index Terms—**Distributed signal estimation, wireless sensor networks, Wiener filtering, ad-hoc topologies**

I. Introduction

WIRELESS sensor networks (WSNs) typically consist of a set of sensor nodes, that are deployed throughout a sensing environment to detect or estimate a signal or parameter of interest. WSNs typically accomplish this estimation in one of two ways. Either all the information is aggregated and processed at a central location or each node takes part in the processing of the data thereby distributing the computation.

In [1]–[4] the WSNs are tasked with estimating a signal or parameter vector of interest. The sensor observations in these WSNs are compressed and then transmitted to a fusion center under a dimensionality constraint which is a by-product of a bandwidth constraint. The fusion center then uses these compressed observations to estimate the parameter vector or signal of interest.

These type estimation of 'compress-and-centralize' WSNs have been extended to WSNs without a fusion center, in which each node acts as a data sink and aims to estimate a parameter vector or signal based on local sensor data and data obtained from its neighboring nodes. A large part of the literature considers the problem where all of the nodes are interested in estimating a single parameter vector through iterative techniques in which nodes share intermediate estimates of the parameter vector with their neighbors. Well-known examples of such algorithms are, e.g., diffusion [5]–[11], consensus [12], [13], gossip [14], [15], or primal-dual decompositions [16]. Instead of estimating the same parameter vector across the WSN, in [17]–[19] it is assumed that there are multiple parameter vectors of interest and each node is only interested in a subset, possibly overlapping, parameter vector of interests, thereby making the problem *node-specific*.

The estimation problem envisaged in this work is inherently different than the previously mentioned literature pertaining to parameter estimation. In the latter, the estimation variable is a parameter vector of fixed dimension, which is assumed to be static over time or at most slowly time varying. This allows each node to perform iterative refinements of the parameter vector estimate while sharing these intermediate estimates with its neighbors, until all local estimates have converged to a steady state. In this work, we are estimating signal samples, and hence a new estimate variable is introduced at each time instant. Initiating a new distributed iterative parameter estimation algorithm, each time, for every individual sample of a rapidly sampled signal, such as, e.g., an audio signal, is then extremely expensive in terms of communication cost. Instead, the nodes apply local compression or fusion rules on their observed signals, before broadcasting them to their neighbors.

It is assumed that the nodes wish to estimate their so-called desired signal observations which are linear mixtures of a set of unknown independent source signals in the environment. It

This research work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Research Council CoE PFV/10/002 'Optimization in Engineering' (OPTEC), Research Project FWO nr. G.0763.12 'Wireless acoustic sensor networks for extended auditory communication', Research Project FWO nr. G.0931.14 'Design of distributed signal processing algorithms and scalable hardware platforms for energy-vs-performance adaptive wireless acoustic sensor networks, Project BOF/STG-14-005 Research Fund KU Leuven, IWT O&O Project nr. 110722 'Signal processing and automatic fitting for next generation cochlear implants' and HANDiCAMS. The project HANDiCAMS acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 323944. The scientific responsibility is assumed by its authors.

J. Szurley is with Robert Bosch LLC Research and Technology Center (RTC) North America 2555 Smallman st. STE 3 Pittsburgh PA 15222, USA (e-mail: joseph.szurley@us.bosch.com)

A. Bertrand and M. Moonen are with KU Leuven, Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics (A. Bertrand is also with iMinds Medical IT) Kasteelpark Arenberg 10, box 2446, 3001 Leuven, Belgium (e-mail: alexander.bertrand, marc.moonen @esat.kuleuven.be).

is noted that this is different than inverse problems and blind source separation which aim to estimate the signal path and unmix the original source signals [20]–[22].

To estimate their desired signal observations, each node makes a weighted linear combination of their local sensor signals and the compressed signals obtained from their neighbors, where the weights are regularly updated based on a linear minimum mean squared error (LMMSE) formulation, assuming some quasi-stationarity conditions on the signals[1]. This compression and estimation step is done only once per sample at each node. Rather than iterating over the estimation variables themselves (i.e., the signal samples), as would be the case in parameter estimation, the LMMSE combination weights and the fusion rules that the nodes use to compress their signals are then recursively updated based on previously observed inputs. This means that the algorithms iterate over these fusion rules rather than the estimation variables.

A. Previous Work and Motivation

The main focus of this paper is to perform distributed *signal* estimation where each node is tasked with estimating its own *node-specific* desired signal without the availability of a fusion center. Such distributed signal estimation techniques have been applied in various contexts such as speech enhancement or direction of arrival estimation in wireless acoustic sensor networks and binaural hearing aids [23]–[25], and artifact removal in wireless EEG sensor networks [26].

This type of node-specific signal estimation has been explored in fully connected, tree-topologies and combinations thereof (mixed-topology) and has led to the introduction of a host of distributed adaptive node-specific signal estimation (DANSE) algorithms [27]–[29]. It has been shown that, even though nodes only transmit a linearly compressed version of their sensor signal observations, each node converges to its optimal node-specific LMMSE signal estimate as if each node were to transmit its raw uncompressed sensor signal observations to every other node in the WSN. The compression used is essentially lossless for the LMMSE task at hand, i.e., we obtain the so-called *centralized* LMMSE solution (as if each node has access to all other uncompressed sensor signal observations), independent of the noise correlation structure, which is not possible when first compressing the data with subspace techniques as in [15].

However, the local LMMSE filter coefficients, signal compression and subsequent signal estimates of the previous DANSE algorithm are neighbor-specific. This entails that the nodes must communicate with the same neighbors during the entire estimation procedure, i.e., the topology must remain static. In the case where the topology would change, e.g. due to a link failure, the DANSE algorithm would then have to reconverge to a new set of filter coefficients to again obtain the optimal node-specific LMMSE signal estimates within the new network topology. However, besides link failure, there are any number of reasons the links may change between nodes, e.g., using a minimum broadcast energy with mobile nodes. In [29], a way to overcome this re-convergence was explored for a mixed (or tree) topology. However, it required the nodes to retain network-wide routing tables along with an increased information exchange to transform the affected nodes' filter coefficients.

Also when nodes are added to the WSN, the computational complexity of the DANSE algorithm in a fully connected and to a smaller degree in a tree topology, at the nodes increases. This increase in computational complexity therefore affects the overall scalability of the algorithms as it can become prohibitively expensive for the nodes to calculate their local LMMSE filter coefficients.

B. Contributions

In this work, the topology-independent distributed adaptive node-specific signal estimation (TI-DANSE) algorithm is presented which overcomes the aforementioned problems of changing topologies and scalability. The aim is to let the nodes converge to a new set of estimator and compression parameters, which always yield the node-specific LMMSE signal estimates, independent of the underlying topology. Furthermore, the topology may even change in between iterations, without the need to let the algorithm re-converge to a new set of estimator and compression parameters for the new topology. In fact, as long as the WSN remains connected, i.e., there exists a path between any two nodes, the TI-DANSE algorithm is robust against changes in topology that can occur due to mobile nodes, link failure, etc. It can be shown that the convergence speed of the TI-DANSE algorithm is independent of the topology or changes therein.

The TI-DANSE algorithm accomplishes this by letting each node compress its signal observations based on a linear compression rule and applying a linear transformation to the sum of the compressed signal observations of the other nodes. This usage of the sum of the compressed signal observations of the other nodes not only results in a complexity reduction in the per-node estimation problems but also makes the algorithm completely scalable when compared to the previous versions of the DANSE algorithm, which is shown in Section III-C. This then means that there is no increase in the per-node computational complexity when nodes are added to the WSN.

To converge to the optimal node-specific signal estimates, the nodes in the TI-DANSE must have access to the sum of all of the compressed signal observations that are shared by the other nodes in the WSN. There are various means to calculate an in-network sum, e.g., relying on gossip or consensus based algorithms [12], [30]–[32]. Although these methods are useful for the summation of fixed or slowly varying parameters, they become impractical for the summation of signal observations that are collected at high sampling rates. Indeed, these methods typically need many iterations to converge to the solution, as well as multiple (re)-broadcasts of the intermediary summed variables. We therefore propose a method to calculate this in-network sum which relies on a tree topology that is formed from the set of available links. The method can be described in a completely data-driven way, i.e., no upper layer coordination is needed between nodes. Since a tree topology is used, the

[1]This quasi-stationarity assumption is needed to accurately collect the relevant statistics of the desired signals and noise.

in-network sum can be accomplished in a maximum of 2 transmissions per node. The tree used for the data-driven signal flow can be chosen randomly at every iteration which differs from the original tree-DANSE (T-DANSE) algorithm [28] where the tree must remain static during the entire estimation.

C. Paper Organization

The structure of the paper is as follows. In Section II the data model is introduced as well as a centralized LMMSE filtering process where it is assumed that all nodes have access to all sensor signal observations in the WSN. Although this contradicts its aims, in Section III the TI-DANSE algorithm is first described in a fully connected topology for the sake of an easy exposition along with a proof of convergence. In Section IV it is explained how the specific nature of the TI-DANSE algorithm allows it to be applied in any topology, relying on an in-network summation of compressed signal observations. A method for this in-network summation is proposed that, at every iteration, partitions the WSN into a tree followed by a data-driven signal flow. Numerical simulations are performed in Section V showing the convergence of the TI-DANSE algorithm compared to previous realizations of the DANSE algorithm. Finally conclusions are given in Section VI.

Notation: Throughout this paper we use the following notation. Lowercase letters denote scalars and boldface lowercase letters denote column vectors. Boldface uppercase letters denote matrices. $(\cdot)^T$, $(\cdot)^H$ denote the transpose and conjugate transpose respectively. The expectation of a random variable is denoted as $\mathcal{E}\{\cdot\}$, the cardinality of a set S is denoted as $|S|$, $\|\cdot\|_2$ and $\|\cdot\|_F$ represent the l^2 and Frobenius norm respectively.

II. Problem setup

We assume a WSN with K nodes, where the set of nodes is denoted as $\mathcal{K} = \{1, \ldots, K\}$, and where each node $k \in \mathcal{K}$ has access to M_k sensor signals. Each sensor signal is modeled as a combination of a node-specific desired signal component and additive noise, i.e., the sensor signal for the m^{th} sensor of node k is

$$y_{k,m} = d_{k,m} + n_{k,m} \tag{1}$$

where $d_{k,m}$ and $n_{k,m}$ are the desired signal component and additive noise, respectively. It is noted that the noise is not assumed to be spatially white, i.e., the noise can be correlated across the different nodes. The sensor signals of node k are placed in a stacked vector of length M_k of the form:

$$\mathbf{y}_k = [y_{k,1}, \ldots, y_{k,M_k}]^T \tag{2}$$

where $\mathbf{d}_k$ and $\mathbf{n}_k$ are defined similarly so that

$$\mathbf{y}_k = \mathbf{d}_k + \mathbf{n}_k \ . \tag{3}$$

Similar to [27], we assume that the desired signal components of each node share the same latent Q-dimensional signal subspace which is given as

$$\mathbf{d}_k = \mathbf{\Psi}_k \mathbf{s} \tag{4}$$

where $\mathbf{s}$ is a Q-dimensional vector that contains the source signals and $\mathbf{\Psi}_k$ is an unknown $M_k \times Q$ steering matrix, which contains the transfer function between the source and sensors and is assumed to be unique for each node. We will also assume that $M_k > Q, \forall\, k$ in the sequel, as this will make the exposition of the algorithm easier. The case where $M_k < Q$ will be briefly addressed in Section III-A.

Each node, k, is tasked with estimating a node-specific desired signal $\overline{\mathbf{d}}_k$, which is a subset of the desired signal components of (4), i.e.,

$$\overline{\mathbf{d}}_k = \overline{\mathbf{\Psi}}_k \mathbf{s} \tag{5}$$

where $\overline{\mathbf{\Psi}}_k$ is again an unknown matrix that contains a subset of the rows of $\mathbf{\Psi}_k$ in (4). Note that none of the elements in (5) are known, except for the fact that $\overline{\mathbf{d}}_k$ contains the desired signals as observed in a known subset of the sensors at node k. This means that it is only assumed that node k knows *which* of its sensor signals it is trying to estimate. An example of this data model is multi-microphone hearing aids which are typically interested in estimating a desired signal as it impinges on one of the microphones of the device (typically the front facing microphone) and uses the other microphone signals to aid in a noise reduction algorithm [25].

For the ease of exposition but without loss of generality, the number of channels in $\overline{\mathbf{d}}_k$ is assumed to be equal to Q in the sequel, such that $\overline{\mathbf{\Psi}}_k$ is a $Q \times Q$ square matrix which will also be assumed to be of full rank in the sequel. The node-specific desired signals can therefore be related to one another by (4) as

$$\overline{\mathbf{d}}_k = \overline{\mathbf{\Psi}}_k (\overline{\mathbf{\Psi}}_q)^{-1} \overline{\mathbf{d}}_q \ . \tag{6}$$

Note that the aim for each node is to perform sensor signal denoising, i.e., a node k aims to estimate the desired signal, $\overline{\mathbf{d}}_k$, as it is observed by its local sensor(s). Although the desired signal components can be the result of a mixing process (see (5)), we do not aim to unmix them.

We first assume that each node has access to every sensor signal in the WSN, that is, each node broadcasts observations of its M_k sensor signals to all other nodes in the WSN, which we refer to as a centralized filtering process. In Section III we discuss how this filtering can be performed in a distributed fashion, where each node only has access to the compressed sensor signal observations of every other node and can only iteratively update a portion of its node-specific estimator.

The sensor signals of the entire WSN are represented as a stacked vector $\mathbf{y}$, of length $M = \sum_{q=1}^{K} M_q$, i.e.,

$$\mathbf{y} = [\mathbf{y}_1^T \cdots \mathbf{y}_K^T]^T \tag{7}$$

and the stacked vectors $\mathbf{d}$ and $\mathbf{n}$ are defined similarly, so that again we have

$$\mathbf{y} = \mathbf{d} + \mathbf{n} \ . \tag{8}$$

An estimate, $\check{\mathbf{d}}_k$, of the node-specific desired signal, $\overline{\mathbf{d}}_k$, of node k is defined by applying a linear filter-and-sum operation on all the sensor signals in the WSN, i.e., $\check{\mathbf{d}}_k = \mathbf{W}_k^H \mathbf{y}$. Note that we consider the case where the signals are complex-valued, which allows to, e.g., apply the algorithm in each frequency bin in the (short-time) Fourier transform domain to also capture time-domain convolutions. The filter $\mathbf{W}_k$ can be thought of as a stacked set of filters, i.e.,

$$\mathbf{W}_k = \begin{bmatrix} \mathbf{W}_{k1} \\ \vdots \\ \mathbf{W}_{kK} \end{bmatrix} \tag{9}$$

where $\mathbf{W}_{kq} \in \mathbb{C}^{M_q \times Q}$ represents the filter that node k applies to the received sensor signal of node q, $\mathbf{y}_q$, and $\mathbf{W}_{kk} \in \mathbb{C}^{M_k \times Q}$ represents the filter that node k applies to its own sensor signals. The node-specific filter for node k, $\mathbf{W}_k$, is found as the linear minimum mean squared error (LMMSE) estimator of the node-specific desired signal $\overline{\mathbf{d}}_k$, i.e.,

$$\widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{k1} \\ \vdots \\ \widehat{\mathbf{W}}_{kK} \end{bmatrix} = \underset{\mathbf{W}_{k1},\ldots,\mathbf{W}_{kK}}{\arg\min} \ \mathcal{E}\left\{ \left\| \overline{\mathbf{d}}_k - \sum_{q=1}^{K} \mathbf{W}_{kq}^H \mathbf{y}_q \right\|_2^2 \right\} \tag{10}$$

$$= \underset{\mathbf{W}_{k1},\ldots,\mathbf{W}_{kK}}{\arg\min} \ J_k(\mathbf{W}_k) \tag{11}$$

where $J_k(\mathbf{W}_k)$ is the cost function of node k.

The solution to (10) is given as the multi-channel Wiener filter (MWF) [33] which has the form

$$\widehat{\mathbf{W}}_k = \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{R}_{\mathbf{y}\overline{\mathbf{d}}_k} \tag{12}$$

where $\mathbf{R}_{\mathbf{yy}} = \mathcal{E}\{\mathbf{y}\mathbf{y}^H\}$ and $\mathbf{R}_{\mathbf{y}\overline{\mathbf{d}}_k} = \mathcal{E}\{\mathbf{y}\overline{\mathbf{d}}_k^H\}$.

Under the assumption of (short-term) stationarity of the signals, the estimation of $\mathbf{R}_{\mathbf{yy}}$ is accomplished straightforwardly by averaging over observations of $\mathbf{y}\mathbf{y}^H$. As the desired signal and the additive noise are assumed to be uncorrelated, $\mathbf{R}_{\mathbf{y}\overline{\mathbf{d}}_k}$ may be represented as

$$\mathbf{R}_{\mathbf{y}\overline{\mathbf{d}}_k} = \mathcal{E}\{\mathbf{y}\overline{\mathbf{d}}_k^H\} \tag{13}$$

$$= \mathcal{E}\{\mathbf{d}\overline{\mathbf{d}}_k^H\} \tag{14}$$

$$= \mathcal{E}\{\mathbf{d}\mathbf{d}^H\}\mathbf{E}_k \tag{15}$$

$$= \mathbf{R}_{\mathbf{dd}}\mathbf{E}_k \tag{16}$$

where $\mathbf{E}_k$ is an $M \times Q$ selection matrix containing only zeros, except for a single entry equal to one in each column to select the desired columns of $\mathbf{R}_{\mathbf{dd}}$ corresponding to the sensors that define the desired signals in $\overline{\mathbf{d}}_k$ in (5). While it is assumed each node knows the dimension of the latent Q-dimensional signal subspace and which local sensor signal observations $\overline{\mathbf{d}}_k$ they wish to estimate, this can be supplanted using subspace tracking techniques.

We note that while $\mathbf{R}_{\mathbf{dd}}$ is unobservable, it can be estimated indirectly from $\mathbf{R}_{\mathbf{yy}}$ by exploiting on/off behavior or some prior knowledge of the signals [23]–[26], [33]. If it is assumed that the desired signals exhibit on/off behavior, $\mathbf{R}_{\mathbf{yy}}$ can be estimated, at a time t, using some type of short-term averaging by means of a forgetting factor α

$$\mathbf{R}_{\mathbf{yy}}[\mathrm{t}] = \alpha \mathbf{R}_{\mathbf{yy}}[\mathrm{t}-1] + (1-\alpha)\mathbf{y}\mathbf{y}^H \tag{17}$$

when the desired signals are present. Likewise when only noise is present, a noise only matrix $\mathbf{R}_{\mathbf{nn}}[\mathrm{t}]$ can be estimated in a similar fashion. This then allows for $\mathbf{R}_{\mathbf{dd}}$ to be estimated as $\mathbf{R}_{\mathbf{dd}} = \mathbf{R}_{\mathbf{yy}} - \mathbf{R}_{\mathbf{nn}}$.

While it is implicitly assumed that the desired signal and noise statistics are sufficiently stationary to be collected via short-term averaging there exist other methods to collect the relevant statistics in non-stationary environments by various means such as extracting quasi-stationary segments [34], assigning a signal-presence probability [35], or using the generalized eigenvalue decomposition [36].

By combining (6) and (12) we see that the columns of the MWF at each node k all span the same Q-dimensional subspace, i.e.,

$$\widehat{\mathbf{W}}_k = \widehat{\mathbf{W}}_q \overline{\mathbf{\Psi}}_{kq} \ , \forall k, q \in K \tag{18}$$

with $\overline{\mathbf{\Psi}}_{kq} = \left(\overline{\mathbf{\Psi}}_q\right)^{-H} \overline{\mathbf{\Psi}}_k^H$.

III. TI-DANSE in a fully connected WSN

In the previous section, it was assumed that each node transmits observations of all of its sensor signals to every other node in the WSN such that each node can compute (12). We now look, instead, to the case where each node only transmits observations of a compressed version of its sensor signals by means of the TI-DANSE algorithm. For the ease of exposition, we first describe the TI-DANSE algorithm in a fully connected WSN, where each node is able to directly communicate with every other node in the WSN. In Section IV the TI-DANSE algorithm is described in WSNs with any topology, where it will be shown that the same local compression that is introduced in this section can also be applied.

A. The TI-DANSE algorithm

We envisage a fully connected WSN, where each node transmits observations of a, yet to be defined (see (28)), linearly compressed version of its sensor signals denoted as $\mathbf{z}_k$. Since the compression that generates $\mathbf{z}_k$ changes in each iteration of the TI-DANSE algorithm, the signal statistics of $\mathbf{z}_k$ will change over time. Therefore, we will add an iteration index i as a superscript to $\mathbf{z}_k$. Each iteration corresponds to a single update of the local compression rule and LMMSE estimator at one node (this will be defined in (23)). In between two iterations, nodes continuously share and fuse compressed sensor observations with their neighbors through the signals $\mathbf{z}_k^i$ until sufficient observations are available to the updating node to accurately estimate the second order statistics required to perform the update (see (23)). Each node then collects observations of its own sensor signals, $\mathbf{y}_k$, and observations of $K-1$ linear compressed signals from the other nodes, $\mathbf{z}_q^i$, $\forall q \in \mathcal{K}\backslash\{k\}$.

Each node now sums the linearly compressed signals from the other nodes. For example, the sum at node k is denoted as $\boldsymbol{\eta}_{-k}^i$, where the subscript $-k$ indicates that there is no contribution from node k and is given as

$$\boldsymbol{\eta}_{-k}^i = \sum_{\forall q \in \mathcal{K}\backslash\{k\}} \mathbf{z}_q^i \ . \tag{19}$$

Node k places $\boldsymbol{\eta}_{-k}^i$ in a stacked vector with its own sensor signals represented as

$$\widetilde{\mathbf{y}}_k^i = \begin{bmatrix} \mathbf{y}_k \\ \boldsymbol{\eta}_{-k}^i \end{bmatrix} \ . \tag{20}$$

Since node k only has access to its own sensor signals $\mathbf{y}_k$ and sums the $\mathbf{z}_q^i$ signals of the other nodes, it can only control a specific part of its node-specific estimator $\mathbf{W}_k$, namely, $\mathbf{W}_{kk}$ (see (9)) and apply a transformation $\mathbf{G}_k \in \mathbb{C}^{Q\times Q}$ to $\boldsymbol{\eta}_{-k}^i$. The node finds its LMMSE estimator with respect to (20) at iteration i by solving

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_k}{\arg\min}\ \mathcal{E}\left\{\left|\left|\overline{\mathbf{d}}_k - \begin{bmatrix} \mathbf{W}_{kk} & \mathbf{G}_k \end{bmatrix}^H \widetilde{\mathbf{y}}_k^i\right|\right|_2^2\right\} \tag{21}$$

$$\mathbf{W}_k^{i+1} = \underset{\mathbf{W}_k}{\arg\min}\ J_k(\mathbf{W}_k) \tag{22}$$

where $J_k(\mathbf{W}_k^i)$ is the cost function of node k at iteration i. The solution to (21) is similar to (12) which using (20) is given as

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \mathbf{R}_{\widetilde{\mathbf{y}}_k^i\widetilde{\mathbf{y}}_k^i}^{-1}\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k} \tag{23}$$

where $\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\widetilde{\mathbf{y}}_k^i} = \mathcal{E}\{\widetilde{\mathbf{y}}_k^i\widetilde{\mathbf{y}}_k^{iH}\}$ and $\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k} = \mathcal{E}\{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k^H\}$.

The local estimation of $\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\widetilde{\mathbf{y}}_k^i}$ at node k, can be accomplished in a similar manner as given in Section II exploiting the on-off behavior of the signal. If we define $\widetilde{\mathbf{d}}_k^i$ similarly to (20), which contains only the desired signals components at node k, then $\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k}$ can be estimated in a similar fashion to (16) as

$$\begin{aligned} \mathbf{R}_{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k} &= \mathcal{E}\{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k^H\} && (24)\\ &= \mathcal{E}\{\widetilde{\mathbf{d}}_k^i\overline{\mathbf{d}}_k^H\} && (25)\\ &= \mathcal{E}\{\widetilde{\mathbf{d}}_k^i\widetilde{\mathbf{d}}_k^{i\,H}\}\widetilde{\mathbf{E}}_k && (26)\\ &= \mathbf{R}_{\widetilde{\mathbf{d}}_k^i\widetilde{\mathbf{d}}_k^i}\widetilde{\mathbf{E}}_k && (27) \end{aligned}$$

where $\widetilde{\mathbf{E}}_k$ is an $(M_k + Q) \times Q$ matrix that has the same functionality as $\mathbf{E}_k$ in (16).

After node k has updated its node-specific LMMSE estimator, $\mathbf{W}_{kk}$ and $\mathbf{G}_k$, it updates the compression used to generate its own broadcast signal, $\mathbf{z}_k^{i+1}$. In particular, $\mathbf{z}_k^{i+1}$ is formed by first linearly combining the sensor signals $\mathbf{y}_k$ by means of the corresponding part of the estimator, $\mathbf{W}_{kk}^{i+1}$, and then transforming this result by the inverse of the other part of the estimator, $\left(\mathbf{G}_k^{i+1}\right)^{-1}$, i.e.,

$$\begin{aligned} \mathbf{z}_k^{i+1} &= \left(\mathbf{G}_k^{i+1}\right)^{-H}\mathbf{W}_k^{i+1\,H}\mathbf{y}_k && (28)\\ &= \mathbf{P}_k^{i+1\,H}\mathbf{y}_k && (29)\\ \mathbf{P}_k^{i+1} &\triangleq \mathbf{W}_{kk}^{i+1}\left(\mathbf{G}_k^{i+1}\right)^{-1}\ . && (30) \end{aligned}$$

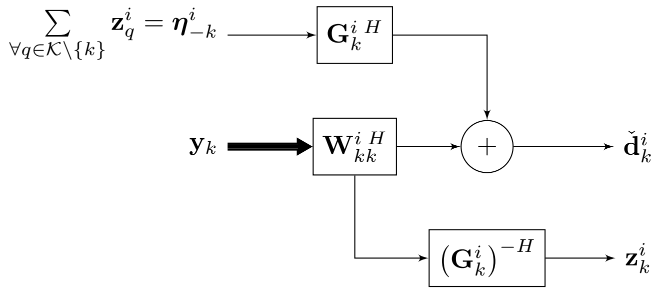
The matrix $\mathbf{P}_k^{i+1}$ is an $M_k \times Q$ matrix, and hence yields a compression ratio of $\frac{M_k}{Q}$. This process is outlined in Table I where it is assumed that the nodes update in a round-robin fashion and is depicted in Figure 1 for node k.

The estimate of $\overline{\mathbf{d}}_k$, denoted as $\check{\mathbf{d}}_k^i$, at any node k and at any point in the iterative process is given as

$$\check{\mathbf{d}}_k^{i+1} = \mathbf{W}_{kk}^{i+1\,H}\mathbf{y}_k + \mathbf{G}_k^{i+1\,H}\boldsymbol{\eta}_{-k}^i\ . \tag{31}$$

Although $\mathbf{z}_k^i$ contains an iteration index, node k does not re-transmit the observations of $\mathbf{z}_k^i$ each time it performs an update. An update at node k only means that *future* observations of $\mathbf{y}_k$ will be compressed into observations of $\mathbf{z}_k^{i+1}$ instead of $\mathbf{z}_k^i$ (using the new $\mathbf{P}_k^{i+1}$). The iterations of TI-DANSE are only performed on the estimator $\mathbf{W}_{kk}$, $\mathbf{G}_k$ and the compressor $\mathbf{P}_k$, but not on the signal observations. In practice, the TI-DANSE algorithm is implemented in a block-adaptive fashion, i.e., the first block of L samples is estimated as $\check{\mathbf{d}}_k^1[n]$ (with $n = 0, ..., L-1$), the second block of L samples is estimated as $\check{\mathbf{d}}_k^2[n]$ (with $n = L, ..., 2L-1$), etc. This means that the initial samples are not well estimated, as the TI-DANSE algorithm has not yet converged.

We note in the case there is a node k with $M_k < Q$, it should merely broadcast its raw sensor signal observations to another node q who will then incorporate these in its own set of M_q sensor signal observations (node k is then excluded as a node in TI-DANSE, as its function will be taken over by node q).



$\sum_{\forall q\in\mathcal{K}\backslash\{k\}} \mathbf{z}_q^i = \boldsymbol{\eta}_{-k}^i$
$\mathbf{G}_k^{i\,H}$
$\mathbf{y}_k$
$\mathbf{W}_{kk}^{i\,H}$
$+$
$\check{\mathbf{d}}_k^i$
$\left(\mathbf{G}_k^i\right)^{-H}$
$\mathbf{z}_k^i$

Fig. 1: A depiction of the filtering and compression scheme for the TI-DANSE algorithm for node k.

TABLE I: TI-DANSE in a fully connected WSN.

1) Initialize $i \leftarrow 0$, $k \leftarrow 1$
2) Initialize $\mathbf{W}_{qq}^0$, $\mathbf{P}_q^0$ and $\mathbf{G}_q^0$ randomly, $\forall q \in \mathcal{K}$
3) Each node transmits its compressed signal observations, $\mathbf{z}_k^i$.
4) Node k updates its node-specific local parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_k$, by minimizing its LMMSE criterion based on its own sensor signals and the summed signals transmitted from the other $K-1$ nodes

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_k}{\arg\min}\ \mathcal{E}\left\{\left|\left|\overline{\mathbf{d}}_k - \begin{bmatrix} \mathbf{W}_{kk} & \mathbf{G}_k \end{bmatrix}^H \widetilde{\mathbf{y}}_k^i\right|\right|_2^2\right\} \tag{32}$$

for which the solution is given by (23) and repeated here for convenience as,

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \mathbf{R}_{\widetilde{\mathbf{y}}_k^i\widetilde{\mathbf{y}}_k^i}^{-1}\mathbf{R}_{\widetilde{\mathbf{y}}_k^i\overline{\mathbf{d}}_k}\ . \tag{33}$$

The compression matrix is then updated as

$$\mathbf{P}_k^{i+1} = \mathbf{W}_{kk}^{i+1}\left(\mathbf{G}_k^{i+1}\right)^{-1}\ . \tag{34}$$

The other nodes do not update their node-specific local parameters:

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i, \mathbf{G}_q^{i+1} = \mathbf{G}_q^i \Rightarrow \mathbf{P}_q^{i+1} = \mathbf{P}_q^i\ . \tag{35}$$

5) $i \leftarrow i+1$
6) $k \leftarrow (i \bmod K) + 1$
7) return to 3

B. Convergence and Optimality

When node k solves its node-specific LMMSE estimation problem, it essentially finds a parameterized version of the node-specific filter given in (9), i.e.,

$$\mathbf{W}_k = \begin{bmatrix} \mathbf{P}_1\mathbf{G}_k \\ \vdots \\ \mathbf{W}_{kk} \\ \vdots \\ \mathbf{P}_K\mathbf{G}_k \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{11}\mathbf{G}_1^{-1}\mathbf{G}_k \\ \vdots \\ \mathbf{W}_{kk} \\ \vdots \\ \mathbf{W}_{KK}\mathbf{G}_K^{-1}\mathbf{G}_k \end{bmatrix} \tag{36}$$

where it can only manipulate the entries in $\mathbf{W}_{kk}$ and $\mathbf{G}_k$. The parameterization in (36) defines a solution space $\mathbf{W}_k, \forall k \in \mathcal{K}$ simultaneously. The next theorem shows that this solution space contains the MWFs given in (12).

Theorem 1. *If (5) holds then the MWFs given in (12) lie in the solution space defined by the parameterization in (36).*

Proof. Setting $\mathbf{G}_k = \overline{\mathbf{\Psi}}_k^H$, $\forall k$, then

$$\mathbf{G}_q^{-1}\mathbf{G}_k = \left(\overline{\mathbf{\Psi}}_q\right)^{-H}\overline{\mathbf{\Psi}}_k^H \tag{37}$$

$$= \overline{\mathbf{\Psi}}_{kq} \; . \tag{38}$$

Substituting (38) into (36) and setting $\mathbf{W}_{kk} = \widehat{\mathbf{W}}_{kk}$, $\forall k \in \mathcal{K}$ yields

$$\forall k \in K \; : \; \widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{11}\mathbf{G}_1^{-1}\mathbf{G}_k \\ \vdots \\ \widehat{\mathbf{W}}_{kk} \\ \vdots \\ \widehat{\mathbf{W}}_{KK}\mathbf{G}_K^{-1}\mathbf{G}_k \; . \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{11}\overline{\mathbf{\Psi}}_{k1} \\ \vdots \\ \widehat{\mathbf{W}}_{kk} \\ \vdots \\ \widehat{\mathbf{W}}_{KK}\overline{\mathbf{\Psi}}_{kK} \end{bmatrix} \tag{39}$$

which, when comparing with (18), shows that the solution space defined by the parameterization in (36) contains the MWFs given in (12). □

Theorem 2. *Consider a WSN with a fully connected topology. If (5) holds, then for any initialization in steps 1 and 2 in Table I, the TI-DANSE algorithm obtains the node-specific LMMSE signal estimates corresponding to the MWFs given in (12) for every node $k \in \mathcal{K}$.*

Proof. See Appendix □

Remark 1. We note that the given parameterization is non-unique, i.e.,

$$\{\widehat{\mathbf{W}}_{11}, \widehat{\mathbf{W}}_{22}, \ldots, \widehat{\mathbf{W}}_{KK}, \mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_K\} \tag{40}$$

and

$$\{\widehat{\mathbf{W}}_{11}, \widehat{\mathbf{W}}_{22}, \ldots, \widehat{\mathbf{W}}_{KK}, \mathbf{T}\mathbf{G}_1, \mathbf{T}\mathbf{G}_2, \ldots, \mathbf{T}\mathbf{G}_K\} \tag{41}$$

will result in the same estimator $\widehat{\mathbf{W}}_k$ for any invertible $\mathbf{T}$. As a consequence, the $\mathbf{G}_k$ matrices computed in the algorithm described in Table I may not converge in the strict sense (we refer to Remark 2 in the Appendix for further details). Nevertheless, since the optimal estimator $\widehat{\mathbf{W}}_k$ itself is unique, the non-uniqueness of its parameterization does not have an impact on the convergence and optimality of the actual signal estimates that are produced by the algorithm.

TABLE II: LMMSE computational complexity at each iteration assuming the inversion of an $M \times M$ matrix requires $O(M^3)$ operations

Algorithm	LMMSE computational complexity at each iteration i
DANSE	$O((M_k + Q(K-1))^3)$
T-DANSE	$O((M_k + Q\lvert\mathcal{T}_k\rvert)^3)$
TI-DANSE	$O((M_k + Q)^3)$

C. Scalability of the TI-DANSE algorithm

We now look to compare the computational complexity of calculating the solution to LMMSE for the TI-DANSE algorithm (23) compared to the previous versions of the DANSE algorithm, specifically fully connected and tree topologies. While the algorithms will not be covered extensively, some background must be given to understand the calculations. For the DANSE algorithm, every node finds its LMMSE solution in a similar manner to (23), which relies on the inverse of a $(M_k + Q(K-1)) \times (M_k + Q(K-1))$ matrix. Likewise, for the T-DANSE algorithm, every node finds its LMMSE solution in a similar manner to (23), which relies on the inverse of a $(M_k + Q|\mathcal{T}_k|) \times (M_k + Q|\mathcal{T}_k|)$ matrix, where $\mathcal{T}_k$ is the set of neighbors of node k in the tree excluding node k itself.

Using these matrix dimensions and assuming a worst-case calculation scenario where the matrix inverse of an $M \times M$ matrix is assumed to require $O(M^3)$ operations, the computation of the LMMSE for a node k at iteration i is given in Table II.

We see that the computational complexity, and therefore scalability, of the DANSE algorithm is impacted the greatest when nodes are added to the WSN. The T-DANSE algorithm is impacted to a lesser extent, as only the nodes who have an added neighbor will have an increase in computational complexity. Finally, since the nodes in the TI-DANSE algorithm rely on the sum of the compressed signal observations of the other nodes, there is no computational increase in the LMMSE estimate of each node when nodes are added to the WSN making it completely scalable.

In the centralized scenario, where no compression takes place, the communication data rate of a node k can be given as $M_k f_s$ samples per second, where f_s is the sampling rate of the sensors. This communication data rate is reduced with the DANSE and TI-DANSE algorithms in place down to Qf_s samples per second per node. With the T-DANSE algorithm in place, however, the communication data rate of a node is dependent on its location in the tree. In fact, for the T-DANSE algorithm, the main compression is not due to the reduction in observations of M_k sensor signals to observations of a Q-channel signal per node, but due to the fact that there is in-network fusion of all the signals. In a tree without in-network data fusion (i.e., with centralized data collection), multi-hop data routing is in place, and hence the amount of data that each node has to transfer will be at least $M_k f_s$ samples per second for nodes with a single connection, and much higher for the other nodes.

IV. TI-DANSE WITHOUT TOPOLOGY CONSTRAINTS

We now describe how the TI-DANSE algorithm presented in Section III, can be implemented in a WSN without topology constraints. This is accomplished with a slight modification to the transmitted signal of a node (28), which now fuses its compressed signal, $\mathbf{z}_k$, with the compressed signal from its neighbors. This will allow for the transmitted signals of the nodes to disperse through the WSN by means of an in-network summation. However, the TI-DANSE in a fully connected WSN as presented in Section III will be shown to be a special case of the proposed modification.

In using the TI-DANSE algorithm we see that, to converge to the optimal solution, each node needs to compute $\boldsymbol{\eta}^i_{-k}$, which can be found from a summed version of all $\mathbf{z}^i_q$'s as defined in (29), i.e., (19) can instead be given as

$$\boldsymbol{\eta}^i_{-k} = \sum_{\forall q \in \mathcal{K}} \mathbf{z}^i_q - \mathbf{z}^i_k \tag{42}$$

$$= \boldsymbol{\eta}^i - \mathbf{z}^i_k \tag{43}$$

where $\boldsymbol{\eta}^i = \sum_{\forall q \in \mathcal{K}} \mathbf{z}^i_q$.

We propose a method to perform this in-network summation via a data-driven signal flow that takes place in a tree topology. While, at first this may seem like a restriction on the topology, it is merely presented as a method to aggregate information in a data-driven way. It is not a requirement of the algorithm itself and the tree can be formed in any manner and the designer is free to choose how it is built, and how frequently it is built (the algorithm is fully generic and requires no assumptions on these aspects). The only true requirement that the TI-DANSE algorithm has is that the nodes have access to $\boldsymbol{\eta}^i_{-k}$.

The node-specific estimation at each node can be performed independent of the actual topology, i.e., equivalently to Table I, such that the algorithm becomes robust to link failures or dynamic topologies. If one of the branches of the tree would get disconnected, a new tree can be grown without the need to let the TI-DANSE algorithm reconverge (as would be the case in the original T-DANSE algorithm [28]). Also the sequential updating order does not need to follow a path though the WSN as in the case of the T-DANSE algorithm. We note that a similar approach to an in-network summation has been presented in [37] to provide a summed output signal to all nodes performing a distributed generalized sidelobe canceler technique.

A. Data-driven signal flow

We assume that when new sensor signal observations become available at the nodes, a tree is pruned from the ad-hoc topology, using any tree formation algorithm[2], in which these signal observations are then fused and disseminated. We denote $\mathcal{N}_k$ as the set of neighbors of node k in the ad-hoc topology with node k excluded. We assume that after the tree is formed, the nodes only communicate with their neighbors in the tree again represented as $\mathcal{T}_k$, which is a subset of the total number of neighbors of the ad-hoc topology, i.e., $\mathcal{T}_k \subseteq \mathcal{N}_k$. Every time a tree is formed, one arbitrary node is assigned as the root node, e.g., if the nodes update in the assumed round-robin fashion as in the fully connected case, the updating node can be chosen as the root node. The following data-driven signal flow is executed for each new block of sensor signal observations collected at the nodes:

1) Any leaf node, i.e., a non-root node k which has only a single neighbor, can immediately fire and transmit a block of compressed observations to its single neighbor (toward the root node) based on (28) which is repeated here for convenience,

$$\mathbf{z}^i_k = \mathbf{P}^{i\,H}_k \mathbf{y}_k \ . \tag{44}$$

Any non-root node k with more than a single neighbor waits until it has received the compressed observations of all its neighbors except for a single neighbor that has yet to fire, say node q, and then computes the block of compressed observations

$$\mathbf{z}^i_k = \mathbf{P}^{i\,H}_k \mathbf{y}_k + \sum_{l \in \mathcal{T}_k \setminus \{q\}} \mathbf{z}^i_l \tag{45}$$

and transmits this to node q (toward the root node). This process repeats at every node in the tree until the root node of the tree is reached.

2) Once the data-driven signal flow has reached the root node, say node r, it generates a block of observations of, $\boldsymbol{\eta}^i$, based on (45)

$$\boldsymbol{\eta}^i = \mathbf{P}^{i\,H}_r \mathbf{y}_r + \sum_{q \in \mathcal{T}_k} \mathbf{z}^i_q \tag{46}$$

which contains all of the compressed sensor signal observations in the WSN. This signal is now flooded through the WSN (away from the root node) so that it reaches every node, where the nodes simply act as relays to pass $\boldsymbol{\eta}^i$ further through the tree.

Based on the data-driven signal flow, we see that any leaf node will transmit only a single block of compressed observations based on (44) during an iteration. Any non-leaf node will transmit a maximum of two blocks of compressed observations during an iteration, first toward the root node based on (45) and away from the root node based on (46).

B. Equivalence to fully connected topology and topology independence

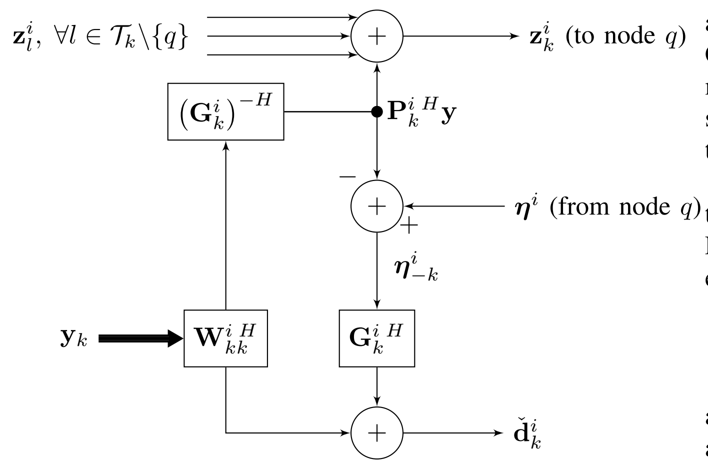
Now that the nodes have access to (46), they subtract out their own compressed $\mathbf{P}^{i\,H}_k \mathbf{y}_k$ signal, i.e.,

$$\boldsymbol{\eta}^i - \mathbf{P}^{i\,H}_k \mathbf{y}_k = \sum_{\forall q \in \mathcal{K} \setminus \{k\}} \mathbf{z}^i_q \tag{47}$$

$$= \boldsymbol{\eta}^i_{-k} \tag{48}$$

which is equivalent to (19). This equivalence means that even though the signal flow took place in a tree topology, it can be viewed from the same perspective as in the fully connected scenario. This shows that at each iteration of the algorithm, since the nodes have access to the same signals as in the

[2]There are several distributed algorithms for tree formation which are dependent on the predefined constraints, e.g., a tree with minimum energy cost [38]–[42]. However, this is not the main focus of this work.



$\mathbf{z}_l^i,\ \forall l \in \mathcal{T}_k\backslash\{q\}$
$\mathbf{z}_k^i$ (to node q)
$(\mathbf{G}_k^i)^{-H}$
$\mathbf{P}_k^{i\,H}\mathbf{y}$
$-$
$+$
$\boldsymbol{\eta}^i$ (from node q)
$\boldsymbol{\eta}_{-k}^i$
$\mathbf{y}_k$
$\mathbf{W}_{kk}^{i\,H}$
$\mathbf{G}_k^{i\,H}$
$\check{\mathbf{d}}_k^i$

Fig. 2: The signal flow of the TI-DANSE algorithm for a non-root node k. Note that for a leaf node, $\mathcal{T}_k\backslash\{q\} = \emptyset$ and the transmitted signal, $\mathbf{z}_k^i$, is equivalent to that of (28).

fully connected case, the same node updating procedure can be applied as given in Table I. The TI-DANSE in an ad-hoc topology is therefore equivalent at every iteration and hence the same convergence and optimality holds[3] as in Section III.

A depiction of the node-specific estimation procedure and proposed signal flow for a non-root node is given in Figure 2. We can see that the TI-DANSE in a fully connected topology can be considered a special case of the TI-DANSE in a tree topology which is explained as follows. Essentially, in the fully connected topology, the updating node can be chosen as the root node and the tree can then be thought of as a star topology where each non-root node is a leaf node and transmits its $\mathbf{z}_k^i$ based on (44). Once it has received all of the leaf node signals, the root node now transmits $\boldsymbol{\eta}^i$ back to the leaf nodes which use (47) to subtract their own compressed $\mathbf{P}_k^{i\,H}\mathbf{y}_k$ signal. However, it is noted that this can be made more efficient by using the fully connected operations described in Section III as the explained process accrues an additional communication hop.

Due to the proposed in-network summation the neighbors between nodes in the tree do not have to remain the same during the estimation procedure. The tree for the signal flow can be randomized at every iteration such that a new root node can be chosen and a different set of neighbors can be chosen, $\mathcal{T}_k^i \subseteq \mathcal{N}_k$. In fact, the signal flow does not need to occur on the same tree toward and away from the root node.

V. Simulations

Numerical simulations are first performed in a single sensing environment comparing the convergence of the 1) DANSE, 2) T-DANSE, 3) TI-DANSE in a fully connected topology (TI-DANSE (FC)) and 4) TI-DANSE in a tree topology (TI-DANSE (T)). The T-DANSE algorithm is then compared to the TI-DANSE (T) algorithm when the links between nodes are chosen randomly during every iteration. Finally, Monte-Carlo simulations are performed on 1000 sensing environments that are generated in the same fashion as in the single sensing environment, to show a broader comparison between the convergence properties of the algorithms.

The simulations are implemented in batch mode indicating that the estimation is performed on the entire length of data. For a signal of length T, the necessary statistics for (12) are estimated as

$$\mathbf{R}_{\mathbf{yy}} \approx \sum_{t=0}^{T-1} \mathbf{y}[t]\mathbf{y}[t]^H,\ \mathbf{R}_{\mathbf{nn}} \approx \sum_{t=0}^{T-1} \mathbf{n}[t]\mathbf{n}[t]^H \qquad (49)$$

and the distributed statistics are found in a similar fashion at each node k. In real-time scenarios, the data could be partitioned into frames and updated as in (17).

A. Single sensing environment

The randomly generated sensing environment contains 15 nodes each with 5 sensors, i.e., $M_k = 5,\ \forall k \in \mathcal{K}$. There are two desired source signals ($Q = 2$) of 10000 samples, which are independently and identically distributed uniformly on an interval of $[-0.5\ 0.5]$ and four localized additive white noise sources which are generated by a similar process and correspond to spatially correlated noise. Additionally, a spatially uncorrelated zero-mean white noise signal that is equal to 10% of the average noise power is also added to the sensor signals to model sensor and quantization noise. This results in a compression of the sensor signal observations of each node by a factor of $\frac{M_k}{Q} = \frac{5}{2}$.

The $\mathbf{W}_{kk}$ variables are all initialized randomly. The DANSE and T-DANSE algorithms apply different estimator coefficients, $\mathbf{G}_{kq}$, to the received signals from their neighbors (see [27] and [28] for further explanation) which are initialized to all-zero matrices. Likewise, for the TI-DANSE algorithm, the $\mathbf{G}_k$ coefficients are initialized to all-zero matrices. The compressor matrices, $\mathbf{P}_k$, are initially set to $\mathbf{P}_k = \mathbf{W}_{kk}$, but later updated according to (34). The nodes in the DANSE and TI-DANSE algorithm update their node-specific local parameters in a round-robin fashion, whereas the T-DANSE algorithm must follow a path based updating scheme to ensure convergence [29], i.e., after a node k updates its node-specific local parameters the next node q must be in $\mathcal{N}_k$.

The optimal centralized solution is first found at each node, assuming that each node transmits observations of its M_k sensor signals to the other nodes in the WSN. The DANSE and TI-DANSE algorithms are then performed where it is assumed that the WSN is fully connected. Next, an ad-hoc WSN is generated by first setting the communication radius of each node to 0 and then expanding the communication radius of each node until the WSN is connected, i.e., every node is reachable by some set of links to every other node in the WSN. This ad-hoc WSN is then pruned to a minimum spanning tree (MST) using Prim's algorithm where the edge weights are equal to the Euclidean distance between nodes. The T-DANSE algorithm and the TI-DANSE algorithm are then performed on the resulting MST. A depiction of the WSN with the ad-hoc and MST links is given in Figure 3. The total

[3]We note that this is not the case when comparing the DANSE algorithm in a fully connected topology to the T-DANSE algorithm.

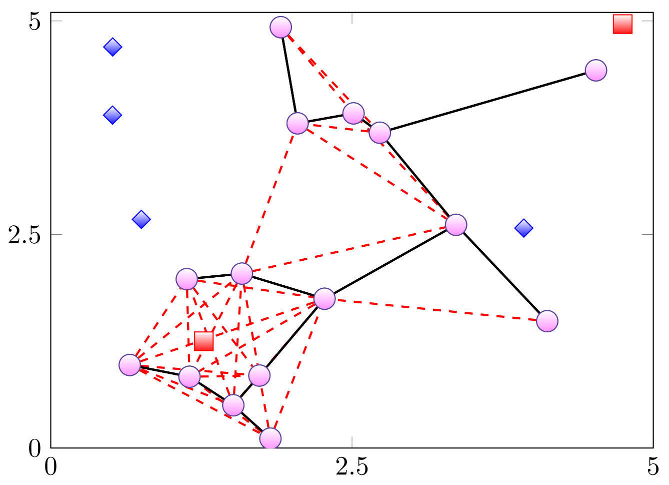


Fig. 3: A simulated environment with 2 desired sources (🟥), 4 uncorrelated noise sources (◆), and 15 nodes ($K = 15$ ●) each with 5 sensors. The dashed red lines indicate the ad-hoc connections and the black lines represent the MST formed using the Euclidean distance between nodes as edge weights.

mean square error (MSE) cost at every iteration, J^i_{Tot}, is found as the sum of individual costs of the nodes using their current node-specific filters, $J_k(\mathbf{W}^i_k)$,

$$J^i_{Tot} = \sum_{k=1}^{K} J_k(\mathbf{W}^i_k)\ . \tag{50}$$

The total MSE cost at every iteration for the various algorithms is shown in Figure 4. We see that DANSE in the fully connected WSN converges to the optimal solution in the fewest number of iterations. The TI-DANSE algorithm in the fully connected (FC) and in the MST topology (T) converge identically due to the independence of the algorithm to the actual topology albeit slower than the other implementations of the DANSE algorithm.

The slower convergence of the TI-DANSE algorithm can be attributed to the number of available degrees of freedom that a node has when finding its LMMSE estimator. In the fully connected case the DANSE algorithm has $M_k + (K-1)Q$ degrees of freedom per update and in a tree topology the T-DANSE algorithm has $M_k + |\mathcal{T}_k|Q$ degrees of freedom when performing an update at node k. The TI-DANSE algorithm has only $M_k + Q$ degrees of freedom when finding its LMMSE estimator in (21). However, this is a trade-off to allow the algorithm to run in a topology-independent fashion yielding more robustness to topology changes such as link failures.

B. Comparison of T-DANSE and TI-DANSE with dynamic connections

Simulations are now performed comparing the T-DANSE and TI-DANSE algorithm when the links in Figure 3 are chosen randomly at every iteration, i.e., $\mathcal{T}^i_k \subseteq \mathcal{N}_k$. This randomization is accomplished by assigning random link weights, which are uniformly distributed over the unit interval $(0, 1]$, to the original ad-hoc links at every iteration. Prim's algorithm is then used to find the MST based on these new randomized link weights. In order to apply the T-DANSE algorithm to the ad-hoc WSN, a different $\mathbf{G}_{kq}$ (see [28] for its definition) is initialized for every possible connection of a node k where $q \in \mathcal{N}_k$. The $\mathbf{G}_{kq}$'s are then updated depending on the current active links of a node during the corresponding iteration $q \in \mathcal{T}^i_k$ and they are kept fixed if the corresponding link disappears (until it appears again in a future tree).

We observe that the TI-DANSE algorithm converges identically as in the static tree given in Figure 4. However, the T-DANSE algorithm fluctuates which is due to the fact that the updates are topology-dependent, and hence cannot converge if the topology changes in between each iteration.

C. Monte-Carlo Simulations

Monte-Carlo simulations are performed on 1000 simulated environments that are initialized similarly to the single scenario presented in Subsection V-A. For every environment the

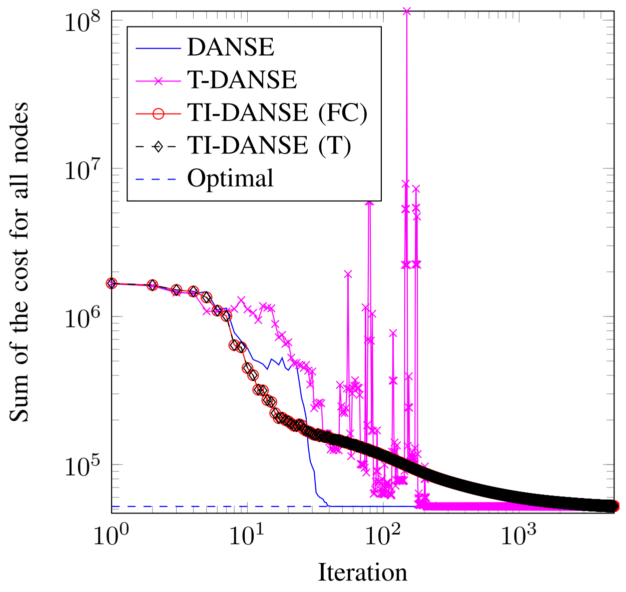


Fig. 4: Cost of the DANSE, T-DANSE, TI-DANSE (FC) and (T) algorithms versus the number of iterations.

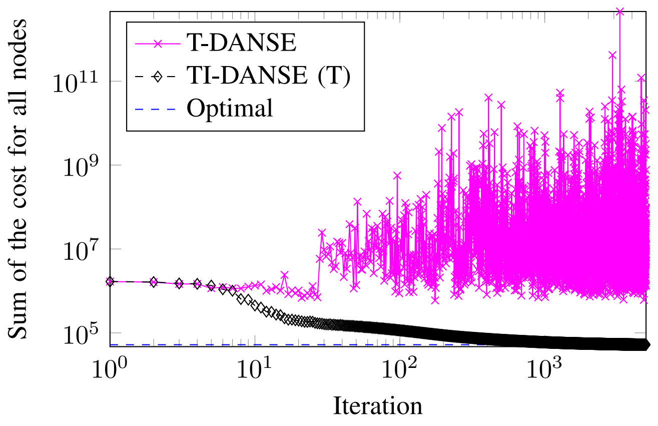


Fig. 5: Cost of the T-DANSE and TI-DANSE (T) algorithms with randomized links in the tree.

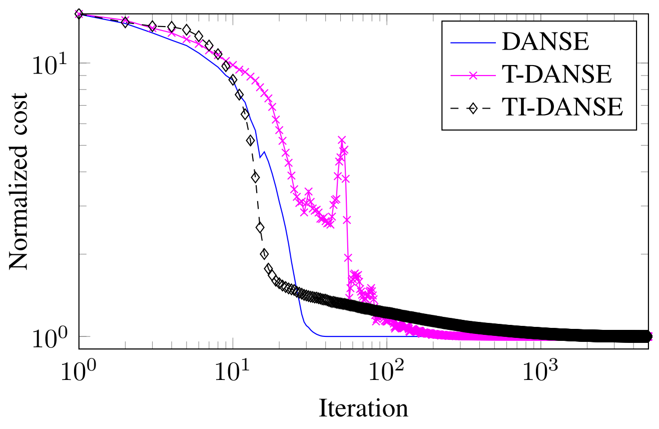


Fig. 6: Normalized cost of the DANSE, T-DANSE and TI-DANSE algorithms versus the number of iterations averaged over 1000 Monte-Carlo runs.

desired and noise sources, as well as 15 nodes, are randomly placed. A MST is found using the Euclidean distances between nodes and kept constant for each Monte-Carlo run. The TI-DANSE (FC) algorithm is not implemented as the convergence properties are identical to that of the TI-DANSE (T) algorithm.

Due to the random nature of the generated signals, the optimal value of the summed cost is different for each Monte-Carlo run. To account for this, the summed cost was normalized by the optimal value for every Monte-Carlo run, i.e.,

$$\tilde{J}^i_{Tot} = \frac{\sum_{k=1}^{K} J_k(\mathbf{W}_k^i)}{\sum_{k=1}^{K} J_k(\widehat{\mathbf{W}}_k)} \ . \tag{51}$$

The normalized sum of the cost for all nodes is then averaged for the 1000 Monte-Carlo runs and shown in Figure 6. We see on average that the convergence of the algorithms is similar to that in the single sensing environment.

VI. Conclusions

In this paper the TI-DANSE algorithm was introduced where nodes in a WSN estimate a node-specific desired signal in a distributed fashion, and independent from the per-iteration topology of the WSN. In using the TI-DANSE, the nodes were shown to be able to converge to their optimal node-specific signal estimates, as if every node had transmitted all of its sensor signals observations to all other nodes. As opposed to the original T-DANSE algorithm, the TI-DANSE algorithm achieves these solutions using any of the available links in the WSN. While the TI-DANSE algorithm typically converges slower when compared to other variations of the DANSE algorithm, it offers the flexibility in being able to be implemented in any topology as long as an in-network signal summation is performed.

Proof of Theorem 2

This proof first considers the centralized case and shows the relationship between two nodes that share the same desired signal up to a sequence of arbitrary linear transforms. An alternating optimization (AO) sequence is then introduced that shows the equivalence between the optimization problems between the two nodes. This result is then extended to show the equivalence between an arbitrary linear transform in the centralized case and a node that has the TI-DANSE algorithm in place. Finally it is shown that the optimization problem of the node with the TI-DANSE algorithm in place convergences to the same estimate as the centralized case.

We first consider the centralized case where it is assumed that all sensor signals are available at each node. We define, for any arbitrary node ν, the centralized cost function as

$$J_\nu(\mathbf{W}_\nu) = \mathcal{E}\left\{\left|\left|\overline{\mathbf{d}}_\nu - \mathbf{W}_\nu^H\mathbf{y}\right|\right|_2^2\right\} \tag{52}$$

where the filter $\mathbf{W}_\nu$ is given in the same form as (9), i.e.,

$$\mathbf{W}_\nu = \left[\ \mathbf{W}_{\nu 1}^T, \ldots, \ \mathbf{W}_{\nu K}^T\ \right]^T \ . \tag{53}$$

We also introduce a cost function that is similar to (52) where now the desired signal $\overline{\mathbf{d}}_\nu$ is transformed by an arbitrary $Q \times Q$ matrix, $\mathbf{Z}^i$, which yields

$$J_z^i(\mathbf{W}_z) = \mathcal{E}\left\{\left|\left|\mathbf{Z}^{i\ H}\overline{\mathbf{d}}_\nu - \mathbf{W}_z^H\mathbf{y}\right|\right|_2^2\right\} \tag{54}$$

where $\mathbf{W}_z$ is partitioned in that same manner as (53) and where the subscript z does not refer to a node. The iteration index i used here does not explicitly refer to the TI-DANSE iterations, but can refer to any generic iterative algorithm.

We define a sequence of transformation matrices that are used to transform the node-specific desired signal in (54) at every iteration i as

$$(\mathbf{Z}^i)_{i\in\mathbb{N}} = (\mathbf{Z}^0, \mathbf{Z}^1, \ldots) \tag{55}$$

which implies that at every iteration a different $\mathbf{Z}^i$ is applied to $\overline{\mathbf{d}}_\nu$.

We now consider a sequence of alternating optimizations (AO) where at each iteration i the LMMSE optimization corresponding to (52) is performed but with constraints added to all partitions of (53) except one, say $\mathbf{W}_{\nu k}$ where k changes in each iteration. The constraints ensure that the columns of $\mathbf{W}_{\nu,-k}$ remain in the current column space of $\mathbf{W}^i_{\nu,-k}$, where we use the notation $\mathbf{X}_{-k}$ to denote a node-by-node stacked matrix as in (53), but where the partition $\mathbf{X}_k$ corresponding to node k is removed. The formal description of this AO process is defined as AO1 in the left column of Table III.

A similar AO procedure AO2 can be described for the more general cost function in (54) where the same partitioning is applied to the optimization variable $\mathbf{W}_z$, but where the sequence of transformations (55) is used to define a new cost function (54) in each iteration of the AO procedure, as formalized in the right column of Table III. Note that, if $\mathbf{Z}^i = \mathbf{I}$, $\forall i$, where $\mathbf{I}$ is an identity matrix of appropriate size, we see that AO1 and AO2 become equivalent.

AO1 will generate the AO sequence

$$(\mathbf{W}_\nu^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^0, \mathbf{W}_\nu^1, \ldots) \ . \tag{56}$$

It is then clear from the definition of the AO procedure that J_ν will monotonically decrease at each step, i.e.,

$$J_\nu(\mathbf{W}_\nu^{i+1}) \leq J_\nu(\mathbf{W}_\nu^i) \ . \tag{57}$$

TABLE III: Alternating Optimization Procedure

AO1, using (52)	AO2, using (54)
	We assume a pre-defined sequence of $Q \times Q$ matrices $(\mathbf{Z}^i)_{i\in\mathbb{N}}$ as given in (55).
1) $k \leftarrow 1$, $i \leftarrow 0$	1) $k \leftarrow 1$, $i \leftarrow 0$
2) initialize $\mathbf{W}_\nu^0$ randomly	2) initialize $\mathbf{W}_z^0$ randomly
3) $(\widehat{\mathbf{X}}_\nu,\, \widehat{\mathbf{A}}_\nu) = \arg\min_{\mathbf{X}_\nu,\mathbf{A}_\nu} J_\nu(\mathbf{X}_\nu)$ s.t. $\mathbf{X}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \mathbf{A}_\nu$	3) $(\widehat{\mathbf{X}}_z,\, \widehat{\mathbf{A}}_z) = \arg\min_{\mathbf{X}_z,\mathbf{A}_z} J_z^{i+1}(\mathbf{X}_z)$ s.t. $\mathbf{X}_{z,-k} = \mathbf{W}_{z,-k}^i \mathbf{A}_z$
4) $\mathbf{W}_\nu^{i+1} \leftarrow \widehat{\mathbf{X}}_\nu$	4) $\mathbf{W}_z^{i+1} \leftarrow \widehat{\mathbf{X}}_z$
5) $k \leftarrow k \bmod K+1, i \leftarrow i+1$	5) $k \leftarrow k \bmod K+1, i \leftarrow i+1$
6) goto 3	6) goto 3

This is because the current estimate $\mathbf{W}_\nu^i$ itself is always in the constraint set of the optimization problem in step 3, and hence the updated estimate, $\mathbf{W}_\nu^{i+1}$, should yield an MSE that is at least as small. In fact, AO1 is actually a relaxed version of a Gauss-Seidel block-coordinate descent (GSBCD) method, where the latter puts an additional constraint that $\mathbf{A}_\nu = \mathbf{I}$, i.e., all entries in $\mathbf{W}_{\nu,-k}^i$ remain fixed. It is known that such a GSBCD method (and hence also its relaxed version) converges to a stationary point of the objective function if this objective function is convex [43]. Therefore, it follows that

$$\lim_{i\to\infty} \mathbf{W}_\nu^i = \widehat{\mathbf{W}}_\nu \tag{58}$$

where $\widehat{\mathbf{W}}_\nu$ is the global minimum of J_ν.

Before we continue the proof, we need the following result:

Lemma 3. *Consider the two AO procedures AO1 and AO2 as given in Table III, and assume that AO1 and AO2 are initialized in step 2 such that* $\mathbf{W}_z^0 = \mathbf{W}_\nu^0 \mathbf{Z}^0$ *. Then AO2 will produce the AO sequence*

$$(\mathbf{W}_z^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^i \mathbf{Z}^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^0\mathbf{Z}^0, \mathbf{W}_\nu^1\mathbf{Z}^1, \ldots) \tag{59}$$

i.e., the sequences $(\mathbf{W}_\nu^i)_{i\in\mathbb{N}}$ *and* $(\mathbf{W}_z^i)_{i\in\mathbb{N}}$ *are equivalent up to a* $Q \times Q$ *transformation for every iteration* i.

Proof. The constrained optimization problems in AO1 and AO2 in Table III can be transformed into unconstrained optimization problems by using the substitutions

$$\mathbf{X}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \mathbf{A}_\nu \tag{60}$$

$$\mathbf{X}_{z,-k} = \mathbf{W}_{z,-k}^i \mathbf{A}_z \tag{61}$$

inside the objective function $J_\nu(\mathbf{X}_\nu)$ and $J_z^{i+1}(\mathbf{X}_z)$, yielding:

$$\begin{bmatrix} \widehat{\mathbf{X}}_{\nu,k} \\ \widehat{\mathbf{A}}_\nu \end{bmatrix} = \arg\min_{\mathbf{X}_{\nu,k},\mathbf{A}_\nu} \mathcal{E}\left\{ \left\| \overline{\mathbf{d}}_\nu - \left[\mathbf{X}_{\nu,k}^H \;\; \mathbf{A}_\nu^H\right] \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H} \mathbf{y}_{-k} \end{bmatrix} \right\|_2^2 \right\} \tag{62}$$

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{A}}_z \end{bmatrix} = \arg\min_{\mathbf{X}_{z,k},\mathbf{A}_z} \mathcal{E}\left\{ \left\| \mathbf{Z}^{i+1\,H}\overline{\mathbf{d}}_\nu - \left[\mathbf{X}_{z,k}^H \;\; \mathbf{A}_z^H\right] \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{z,-k}^{i\,H} \mathbf{y}_{-k} \end{bmatrix} \right\|_2^2 \right\} \tag{63}$$

with $\mathbf{y}_{-k}$ denoting the vector $\mathbf{y}$ with $\mathbf{y}_k$ removed. The full matrix $\widehat{\mathbf{X}}_\nu^{i+1}$ is eventually found by including $\widehat{\mathbf{X}}_{\nu,k}$ into $\widehat{\mathbf{X}}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{A}}_\nu$ again at the correct place (corresponding to node k), and similarly for $\widehat{\mathbf{X}}_z^{i+1}$.

We will prove the lemma using an inductive argument. To this end, we first assume that the lemma holds up to iteration i, i.e., $\mathbf{W}_z^i = \mathbf{W}_\nu^i \mathbf{Z}^i$, from which we will show that also $\mathbf{W}_z^{i+1} = \mathbf{W}_\nu^{i+1}\mathbf{Z}^{i+1}$. If we indeed assume that $\mathbf{W}_z^i = \mathbf{W}_\nu^i \mathbf{Z}^i$, then (63) can be rewritten as

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{A}}_z \end{bmatrix} = \arg\min_{\mathbf{X}_{z,k},\mathbf{A}_z} \mathcal{E}\left\{ \left\| \mathbf{Z}^{i+1\,H}\overline{\mathbf{d}}_\nu - \left[\mathbf{X}_{z,k}^H \;\; \mathbf{A}_z^H \mathbf{Z}^{i\,H}\right] \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H} \mathbf{y}_{-k} \end{bmatrix} \right\|_2^2 \right\} \tag{64}$$

and using the substitution $\mathbf{B} = \mathbf{Z}^i \mathbf{A}_z$, we obtain

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{B}} \end{bmatrix} = \arg\min_{\mathbf{X}_{z,k},\mathbf{B}} \mathcal{E}\left\{ \left\| \mathbf{Z}^{i+1\,H}\overline{\mathbf{d}}_\nu - \left[\mathbf{X}_{z,k}^H \;\; \mathbf{B}^H\right] \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H} \mathbf{y}_{-k} \end{bmatrix} \right\|_2^2 \right\} \tag{65}$$

where $\widehat{\mathbf{X}}_{z,-k}$ is found as

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{B}} \; . \tag{66}$$

which can be derived from the fact that

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}_{z,-k}^i \widehat{\mathbf{A}}_z \tag{67}$$

$$= \mathbf{W}_{\nu,-k}^i \mathbf{Z}^i \widehat{\mathbf{A}}_z \tag{68}$$

$$= \mathbf{W}_{\nu,-k}^i \mathbf{Z}^i \left(\mathbf{Z}^i\right)^{-1} \widehat{\mathbf{B}} \tag{69}$$

$$= \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{B}} \; . \tag{70}$$

Let us now compare (62) and (65), and observe that they both define a LMMSE problem with a similar form as (21), and hence their solution will have a similar form as (23). Furthermore, (62) and (65) are identical optimization problems except for a multiplication of the desired signal $\overline{\mathbf{d}}_\nu$ with $\mathbf{Z}^{i+1\,H}$. Considering (23), their solutions will therefore only differ up to a right multiplication with $\mathbf{Z}^{i+1}$, i.e.,

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{B}} \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{X}}_{\nu,k} \\ \widehat{\mathbf{A}}_\nu \end{bmatrix} \mathbf{Z}^{i+1} \; . \tag{71}$$

Plugging the lower part of (71) in (66) yields

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{A}}_\nu \mathbf{Z}^{i+1} \tag{72}$$

and since

$$\widehat{\mathbf{X}}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{A}}_\nu \tag{73}$$

we obtain

$$\widehat{\mathbf{X}}_{z,-k} = \widehat{\mathbf{X}}_{\nu,-k} \mathbf{Z}^{i+1} \; . \tag{74}$$

Combining this with the upper part in (71), we eventually obtain that the combined matrices $\widehat{\mathbf{X}}_z$ and $\widehat{\mathbf{X}}_\nu$ are related as

$$\widehat{\mathbf{X}}_z = \widehat{\mathbf{X}}_\nu \mathbf{Z}^{i+1} \tag{75}$$

and hence, from Table III, also

$$\mathbf{W}_z^{i+1} = \mathbf{W}_\nu^{i+1} \mathbf{Z}^{i+1} \; . \tag{76}$$

We have thus shown that, if $\mathbf{W}_z^i = \mathbf{W}_\nu^i \mathbf{Z}^i$, then also $\mathbf{W}_z^{i+1} = \mathbf{W}_\nu^{i+1} \mathbf{Z}^{i+1}$. Since the former holds for $i = 0$ due to the particular initialization of both AOs, the lemma is proven for any iteration i by an induction argument. $\square$

An important corollary from this lemma is that, since the AO sequence $(\mathbf{W}_\nu^i)_{i \in \mathbb{N}}$ converges, (see 58) then it immediately follows from Lemma 3 that $(\mathbf{W}_z^i)_{i \in \mathbb{N}}$ will also converge up to a $Q \times Q$ transformation of its columns, i.e.,

$$\lim_{i \to \infty} \min_{\mathbf{U}} \left|\left| \widehat{\mathbf{W}}_\nu - \mathbf{W}_z^i \mathbf{U} \right|\right|_F = 0 \tag{77}$$

for any given sequence $(\mathbf{Z}^i)_{i \in \mathbb{N}} = (\mathbf{Z}^0, \mathbf{Z}^1, \ldots)$. Note that, in Lemma 3, $\mathbf{Z}^0$ merely defines the initialization of AO2 with respect to the initialization of AO1. However, since both initializations are arbitrary, we will assume that $\mathbf{Z}^0 = \mathbf{I}$ for any choice of the $\mathbf{Z}$-sequence (55) in the sequel.

Let us now consider a new AO procedure, referred to as AO3, which is defined as AO1, but where the objective function J_ν is replaced with the objective function $J_{k(i)}$, where the node-index $k(i)$ now increments in each iteration, looping over all nodes of $\mathcal{K}$ in a round-robin fashion. Similar as AO1 being a special case of AO2 (where $\mathbf{Z}^i = \mathbf{I}$, $\forall\, i \in \mathbb{N}$), also AO3 can be shown to be a special case of AO2. Indeed, we obtain AO3 by choosing

$$\mathbf{Z}^{i+1} = \left(\overline{\mathbf{\Psi}}_\nu\right)^{-H} \overline{\mathbf{\Psi}}_{k(i)}^H \tag{78}$$

in AO2. This means that the sequence of AO3 will also converge up to a $Q \times Q$ transformation similar to (77).

The changing node index in the objective function $J_{k(i)}$ used in AO3 allows to implement AO3 in a distributed fashion in a fully connected WSN, which is explained next. We will refer to this distributed algorithm as the D-AO3 algorithm. Similar to the TI-DANSE algorithm, we also define a specific compression at node k, which we denote as $\mathbf{V}_k^i$ (having a similar function as $\mathbf{P}_k^i$ in the TI-DANSE algorithm), and its corresponding broadcast signal is again denoted as $\mathbf{z}_k^i = \mathbf{V}_k^{i\,H} \mathbf{y}_k$. Using this notation, the D-AO3 algorithm is described in Table IV (with the introduction of some auxiliary variables $\mathbf{G}_k$ and $\mathbf{W}_{kk}$). In the description of D-AO3, we use an incremental node-index k instead of the notation $k(i)$.

If we now stack all the $\mathbf{V}_k^i$'s defined in the D-AO3 algorithm into a larger matrix

$$\mathbf{V}^i = \begin{bmatrix} \mathbf{V}_1^i \\ \vdots \\ \mathbf{V}_K^i \end{bmatrix} \tag{79}$$

then the sequence $(\mathbf{V}^i)_{i \in \mathbb{N}}$ of the D-AO3 algorithm will be identical to the sequence produced by the AO3 algorithm[4], i.e.,

$$(\mathbf{V}^i)_{i \in \mathbb{N}} = (\mathbf{W}_z^i)_{i \in \mathbb{N}} = (\mathbf{W}_\nu^i \mathbf{Z}^i)_{i \in \mathbb{N}} \tag{80}$$

TABLE IV: Description of the D-AO3 algorithm in a fully connected WSN.

1) Initialize $i \leftarrow 0$, $k \leftarrow 1$ (or $k(i) = k(0) = 1$)
2) Initialize $\mathbf{W}_{qq}^0$,$\mathbf{G}_q^0$, and $\mathbf{V}_q^0$ randomly, $\forall q \in \mathcal{K}$
3) Node k updates $\mathbf{W}_{kk}$ and $\mathbf{G}_k$ by minimizing its LMMSE criterion based on its own sensor signals and the summed broadcast signals from the other $K-1$ nodes

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk}, \mathbf{G}_k}{\arg\min}\ \mathcal{E}\left\{ \left|\left| \overline{\mathbf{d}}_k - \begin{bmatrix} \mathbf{W}_{kk} & \mathbf{G}_k \end{bmatrix}^H \tilde{\mathbf{y}}_k^i \right|\right|_2^2 \right\} \tag{82}$$

for which the solution is given by (23), and updates its compression matrix as

$$\mathbf{V}_k^{i+1} = \mathbf{W}_{kk}^{i+1} \ . \tag{83}$$

Furthermore, node k broadcasts $\mathbf{G}_k^{i+1}$ to the other nodes, who perform the following update on their local compression

$$\forall q \in \mathcal{K} \backslash \{k\} : \mathbf{V}_q^{i+1} = \mathbf{V}_q^i \mathbf{G}_k^{i+1} \ . \tag{84}$$

The other node-specific parameters are not updated:

$$\forall q \in \mathcal{K} \backslash \{k\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i \text{ and } \mathbf{G}_q^{i+1} = \mathbf{G}_q^i \tag{85}$$

4) $k \leftarrow (i \bmod K) + 1$ (or $k(i) = (i \bmod K) + 1$)
5) $i \leftarrow i$+1
6) return to 3

where $(\mathbf{Z}^i)_{i \in \mathbb{N}}$ is defined in (78). Therefore, using the result (77), we find that

$$\lim_{i \to \infty} \min_{\mathbf{U}} \left|\left| \widehat{\mathbf{W}}_\nu - \mathbf{V}^i \mathbf{U} \right|\right|_F = 0 \ . \tag{81}$$

Assume that, in the beginning of step 3 of iteration i in the D-AO3 algorithm, we replace all of the compression $\mathbf{V}_q^i$ as follows

$$\forall\, q \in \mathcal{K} : \mathbf{V}_q^i \leftarrow \mathbf{V}_q^i \mathbf{T} \tag{86}$$

with an arbitrary full rank $Q \times Q$ matrix $\mathbf{T}$. Then it can be shown that this will not have any influence on the D-AO3 algorithm in the sense that $\mathbf{V}^{i+1}$ will be the same for any choice of $\mathbf{T}$. This can be proven as follows. First, recall that an iteration of D-AO3 is equivalent to an iteration of AO3, which at its turn is equivalent to an iteration of AO2, for the specific choice of $(\mathbf{Z}^i)_{i \in \mathbb{N}}$ defined in (78), and where $\mathbf{V}^i$ corresponds to $\mathbf{W}_z^i$. Therefore, step 3 in the D-AO3 algorithm is equivalent to solving the constrained optimization problem in step 3 of AO2. Since (86) does not change the constraint set, i.e., the column space of $\mathbf{V}^i$, it will not influence the outcome of this constrained optimization problem, and hence will not influence the resulting $\mathbf{W}_z^{i+1}$, or equivalently $\mathbf{V}^{i+1}$.

We will now investigate the case where $\mathbf{T}$ is chosen as $\mathbf{T} = \left(\mathbf{G}_{k(i-1)}^i\right)^{-1}$, $\forall\, i \in \mathbb{N}$. This is equivalent[5] to replacing (83) and (84) in Table IV with

$$\mathbf{P}_k^{i+1} = \mathbf{W}_{kk}^{i+1} \left(\mathbf{G}_k^{i+1}\right)^{-1} \tag{87}$$

and

$$\forall q \in \mathcal{K} \backslash \{k\} : \mathbf{P}_q^{i+1} = \mathbf{P}_q^i \ , \tag{88}$$

[4]This follows from the fact that (82) is essentially the same as (63) where $\mathbf{V}_{-k}^i$ replaces $\mathbf{W}_{z,-k}^i$, such that $\boldsymbol{\eta}_{-k}^i = \mathbf{V}_{-k}^{i\,H} \mathbf{y}_{-k} = \mathbf{W}_{z,-k}^{i\,H} \mathbf{y}_{-k}$. The transmission of $\mathbf{G}_k^{i+1}$ to the other nodes, which then multiply their $\mathbf{V}_q^i$ with $\mathbf{G}_k^{i+1}$, corresponds to the resubstitution defined in (61). Therefore, each iteration of D-AO3 corresponds to an iteration of AO3, i.e., steps 3+4 in AO2 where $\mathbf{V}^i$ replaces $\mathbf{W}_z^i$.

[5]Note that applying $\mathbf{V}_q^i \leftarrow \mathbf{V}_q^i \left(\mathbf{G}_{k(i-1)}^i\right)^{-1}$ in the beginning of step 3 is equivalent to applying $\mathbf{V}_q^{i+1} \leftarrow \mathbf{V}_q^{i+1} \left(\mathbf{G}_{k(i)}^{i+1}\right)^{-1}$ at the end of step 3.

respectively, where we have also replaced the symbol $\mathbf{V}^i$ with $\mathbf{P}^i$ to distinguish between the transformed and the non-transformed case. A key observation is that, after performing the above replacements in Table IV, we actually obtain the TI-DANSE algorithm described in Table I.

Therefore, to prove convergence of the TI-DANSE algorithm, we have to analyze the sequence

$$\left(\mathbf{P}^i\right)_{i\in\mathbb{N}} \triangleq \left(\mathbf{V}^i\left(\mathbf{G}^i_{k(i)}\right)^{-1}\right)_{i\in\mathbb{N}} . \qquad (89)$$

From (81) and (89) it follows that $(\mathbf{P}^i)_{i\in\mathbb{N}}$ will again converge up to a $Q\times Q$ transformation of its columns, i.e.,

$$\lim_{i\to\infty}\min_{\mathbf{U}}\left|\left|\widehat{\mathbf{W}}_\nu - \mathbf{P}^i\mathbf{U}\right|\right|_F = 0 . \qquad (90)$$

We now let

$$\mathbf{Q}^i \triangleq \arg\min_{\mathbf{U}}\left|\left|\widehat{\mathbf{W}}_\nu - \mathbf{P}^i\mathbf{U}\right|\right|_F \qquad (91)$$

for any iteration i (note that $\mathbf{Q}^i$ cannot be computed in practice, since $\widehat{\mathbf{W}}_\nu$ is unknown). From (90) and (91), it follows that

$$i\to\infty \Rightarrow \mathbf{P}^i_k = \widehat{\mathbf{W}}_{\nu k}\left(\mathbf{Q}^i\right)^{-1}\ \forall k\in\mathcal{K} . \qquad (92)$$

If node k performs an update at iteration i, its estimate of its node-specific desired signal $\overline{\mathbf{d}}_k$ in TI-DANSE is given as (see (31)):

$$\check{\mathbf{d}}^{i+1}_k = \left(\mathbf{W}^{i+1}_{kk}\right)^H\mathbf{y}_k + \left(\mathbf{G}^{i+1}_k\right)^H\boldsymbol{\eta}^i_{-k} \qquad (93)$$

where

$$\boldsymbol{\eta}^i_{-k} \triangleq \sum_{\forall q\in\mathcal{K}\setminus\{k\}}\mathbf{z}^i_q\ \ \text{(see (19))} \qquad (94)$$

$$= \sum_{\forall q\in\mathcal{K}\setminus\{k\}}\mathbf{P}^{i\,H}_q\mathbf{y}_q \qquad (95)$$

$$= \mathbf{P}^{i\,H}_{-k}\mathbf{y}_{-k} \qquad (96)$$

where $\mathbf{y}_{-k}$ is defined as in (7), but with $\mathbf{y}_k$ removed.

Using (96) and (92), and for $i\to\infty$, (93) becomes

$$i\to\infty : \check{\mathbf{d}}^{i+1}_k = \left(\mathbf{W}^{i+1}_{kk}\right)^H\mathbf{y}_k + \left(\mathbf{G}^{i+1}_k\right)^H\left(\mathbf{Q}^i\right)^{-H}\left(\widehat{\mathbf{W}}_{\nu,-k}\right)^H\mathbf{y}_{-k} . \qquad (97)$$

If node k would choose the following values

$$i\to\infty : \mathbf{W}^{i+1}_{kk} = \widehat{\mathbf{W}}_{\nu k}\left(\overline{\boldsymbol{\Psi}}_\nu\right)^{-H}\overline{\boldsymbol{\Psi}}^H_k \qquad (98)$$

$$\mathbf{G}^{i+1}_k = \mathbf{Q}^i\left(\overline{\boldsymbol{\Psi}}_\nu\right)^{-H}\overline{\boldsymbol{\Psi}}^H_k \qquad (99)$$

then it is easy to see that, when using (18), (97) becomes

$$i\to\infty : \check{\mathbf{d}}^{i+1}_k = \left(\overline{\boldsymbol{\Psi}}_k\right)\overline{\boldsymbol{\Psi}}^{-1}_\nu\widehat{\mathbf{W}}^H_\nu\mathbf{y} = \widehat{\mathbf{W}}^H_k\mathbf{y} \qquad (100)$$

i.e., node k converges to the solution of the network-wide LMMSE estimation problem by making a proper choice for its local parameters, which proves convergence and optimality of the node-specific signal estimates. This concludes the proof. We can, however, elaborate a bit more on the convergence of the internal parameters $\mathbf{W}_{kk}$ and $\mathbf{G}_k$. Due to the strict convexity of the local LMMSE estimation problem at node k, the optimality of (100) can only be achieved if node k indeed exactly performs the update given in (98) and (99). The right-hand side of (98) is not dependent on the iteration index i, and hence this implies convergence for $\mathbf{W}_{kk}$, $\forall k\in\mathcal{K}$. However, this does not hold for the right-hand side of (99). To prove convergence of the $\mathbf{G}_k$'s, $\mathbf{Q}^i$ must also converge to ensure that the right-hand side of (99) also becomes independent of the iteration index.

Using (98) and (99) when $i\to\infty$ yields

$$i\to\infty : \mathbf{P}^{i+1}_k \triangleq \mathbf{W}^{i+1}_{kk}\left(\mathbf{G}^{i+1}_k\right)^{-1} \qquad (101)$$

$$= \widehat{\mathbf{W}}_{\nu k}\overline{\boldsymbol{\Psi}}^{-H}_\nu\overline{\boldsymbol{\Psi}}^H_k\overline{\boldsymbol{\Psi}}^{-H}_k\overline{\boldsymbol{\Psi}}^H_\nu\left(\mathbf{Q}^i\right)^{-1} \qquad (102)$$

$$= \widehat{\mathbf{W}}_{\nu k}\left(\mathbf{Q}^i\right)^{-1} \qquad (103)$$

which with (92), yields

$$i\to\infty : \mathbf{P}^{i+1}_k = \mathbf{P}^i_k . \qquad (104)$$

This together with (91), shows that

$$i\to\infty : \mathbf{Q}^{i+1} = \mathbf{Q}^i \qquad (105)$$

and with (99)

$$i\to\infty : \mathbf{G}^{i+1}_k = \mathbf{G}^i_k \qquad (106)$$

which proves that subsequent values of $\mathbf{G}^i_k$, $\forall k$ become identical when $i\to\infty$.

Remark 2. It is noted that there is a slight abuse of the '=' notation in (105) and (106), as well as the expressions that were used to derive these. A correct way would be:

$$\lim_{i\to\infty}\|\Delta^i\| = 0 \qquad (107)$$

$$\Delta^{i+1} \triangleq \mathbf{G}^{i+1}_k - \mathbf{G}^i_k . \qquad (108)$$

Note that (107) does not truly imply convergence of $\mathbf{G}^i_k$ in the strict mathematical sense, since $\sum_{i=0}^{\infty}\Delta^i$ can still be infinitely large. Nevertheless, the optimality and convergence of the signal estimates, as stated in (100), will still hold, even without $\mathbf{G}^i_k$ converging. □

References

[1] Z.-Q. Luo, G. Giannakis, and S. Zhang, "Optimal linear decentralized estimation in a bandwidth constrained sensor network," in *Proc. of the Int. Symp. on Inform. Theory (ISIT)*, Sep. 2005, pp. 1441–1445.

[2] C. Yu and G. Sharma, "Distributed estimation using reduced dimensionality sensor observations: A separation perspective," in *Proc. of the Annu. Int. Conf. on Inform. Sciences and Systems (CISS)*, Mar. 2008, pp. 150–154.

[3] I. Schizas, G. Giannakis, and Z.-Q. Luo, "Distributed estimation using reduced-dimensionality sensor observations," *IEEE Trans. on Signal Process.*, vol. 55, no. 8, pp. 4284–4299, Aug. 2007.

[4] M. Gastpar, P.-L. Dragotti, and M. Vetterli, "The distributed Karhunen - Loeve transform," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5177–5196, Dec. 2006.

[5] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. on Signal Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[6] A. Das and M. Mesbahi, "Distributed linear parameter estimation over wireless sensor networks," *IEEE Trans. on Aerosp. and Electron. Syst.*, vol. 45, no. 4, pp. 1293–1306, Oct. 2009.

[7] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[8] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. on Signal Process.*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

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

[10] S. Chouvardas, K. Slavakis, and S. Theodoridis, “Trading off complexity with communication costs in distributed adaptive learning via krylov subspaces for dimensionality reduction,” *IEEE Journal of Selected Topics in Signal Process.*, vol. 7, no. 2, pp. 257–273, Apr. 2013.

[11] 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, Sep. 2015.

[12] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

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

[14] D. Shah, “Gossip algorithms,” *Foundations and Trends in Networking*, vol. 3, no. 1, pp. 1–125, 2008.

[15] L. Li, A. Scaglione, and J. H. Manton, “Distributed principal subspace estimation in wireless sensor networks,” *IEEE Journal of Topics in Signal Process.*, vol. 5, no. 4, pp. 725–738, Aug. 2011.

[16] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[17] J. Chen, C. Richard, and A. H. Sayed, “Diffusion LMS for clustered multitask networks,” in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, May 2014, pp. 5487–5491.

[18] J. Plata-Chaves, N. Bogdanovic, and K. Berberidis, “Distributed diffusion-based LMS for node-specific adaptive parameter estimation,” *IEEE Trans. on Signal Process.*, vol. PP, no. 99, pp. 1–1, 2015.

[19] N. Bogdanovic, J. Plata-Chaves, and K. Berberidis, “Distributed incremental-based LMS for node-specific adaptive parameter estimation,” *IEEE Trans. on Signal Process.*, vol. 62, no. 20, pp. 5382–5397, Oct. 2014.

[20] J. F. Cardoso, “Blind signal separation: statistical principles,” *in the Proc. of the IEEE*, vol. 86, no. 10, pp. 2009–2025, Oct. 1998.

[21] J. Murray-Bruce and P. L. Dragotti, “Estimating localized sources of diffusion fields using spatiotemporal sensor measurements,” *IEEE Transactions on Signal Processing*, vol. 63, no. 12, pp. 3018–3031, June 2015.

[22] S. Kitić, L. Albera, N. Bertin, and R. Gribonval, “Physics-driven inverse problems made tractable with cosparse regularization,” *IEEE Transactions on Signal Processing*, vol. 64, no. 2, pp. 335–348, Jan 2016.

[23] A. Bertrand and M. Moonen, “Robust distributed noise reduction in hearing aids with external acoustic sensor nodes,” *EURASIP J. on Advances in Signal Process.*, 2009.

[24] A. Hassani, A. Bertrand, and M. Moonen, “Cooperative integrated noise reduction and node-specific direction-of-arrival estimation in a fully connected wireless acoustic sensor network,” *Signal Processing*, vol. 107, pp. 68–81, Feb. 2015.

[25] S. Doclo, M. Moonen, T. Van den Bogaert, and J. Wouters, “Reduced-bandwidth and distributed MWF-based noise reduction algorithms for binaural hearing aids,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 17, no. 1, pp. 38–51, Jan. 2009.

[26] A. Bertrand, “Distributed signal processing for wireless EEG sensor networks,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 6, pp. 923–935, 2015.

[27] A. Bertrand and M. Moonen, “Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating,” *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5277–5291, Oct. 2010.

[28] ——, “Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology,” *IEEE Trans. on Signal Process.*, vol. 59, no. 5, pp. 2196–2210, May 2011.

[29] J. Szurley, A. Bertrand, and M. Moonen, “Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks,” *Signal Processing*, vol. 117, pp. 44–60, Dec. 2015.

[30] A. Dimakis, S. Kar, J. Moura, M. Rabbat, and A. Scaglione, “Gossip algorithms for distributed signal processing,” *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[31] Y. Zeng and R. C. Hendriks, “Distributed delay and sum beamformer for speech enhancement via randomized gossip,” *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 22, pp. 260 – 273, Jan. 2014.

[32] Y. Zeng, R. C. Hendriks, and R. Heusdens, “Clique-based distributed beamforming for speech enhancement in wireless sensor networks,” in *Proc. of the European Signal Process. Conf. (EUSIPCO)*, Marrakech, Morocco, Sep. 2013.

[33] S. Doclo and M. Moonen, “GSVD-based optimal filtering for single and multimicrophone speech enhancement,” *IEEE Trans. on Signal Process.*, vol. 50, no. 9, pp. 2230–2244, Sep. 2002.

[34] J. Hu, J. Gao, and K. White, “Estimating measurement noise in a time series by exploiting nonstationarity,” *Chaos, Solitons & Fractals*, vol. 22, no. 4, pp. 807 – 819, 2004.

[35] S. Rangachari and P. C. Loizou, “A noise-estimation algorithm for highly non-stationary environments,” *Speech Communication*, vol. 48, no. 2, pp. 220 – 231, 2006.

[36] R. Serizel, M. Moonen, B. Van Dijk, and J. Wouters, “Low-rank approximation based multichannel Wiener filter algorithms for noise reduction with application in cochlear implants,” *IEEE/ACM Trans. on Audio, Speech, and Language Process.*, vol. 22, no. 4, pp. 785–799, Apr. 2014.

[37] S. Markovich-Golan, A. Bertrand, M. Moonen, and S. Gannot, “Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks,” *Signal Processing*, vol. 107, pp. 4 – 20, 2015.

[38] R. G. Gallager, P. A. Humblet, and P. M. Spira, “A distributed algorithm for minimum-weight spanning trees,” *ACM Trans. Program. Lang. Syst.*, vol. 5, no. 1, pp. 66–77, Jan. 1983.

[39] H. Chen, A. M. Campbell, B. W. Thomas, and A. Tamir, “Minimax flow tree problems,” *Networks*, vol. 54, no. 3, pp. 117–129, 2009.

[40] Y. Choi, M. Khan, V. Kumar, and G. Pandurangan, “Energy-optimal distributed algorithms for minimum spanning trees,” *IEEE J. Sel. Areas Commun.*, vol. 27, no. 7, pp. 1297–1304, Sep. 2009.

[41] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “Tag: A tiny aggregation service for ad-hoc sensor networks,” *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, Dec. 2002.

[42] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks,” in *Proc. of the ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom)*, 1999, pp. 174–185.

[43] J. C. Bezdek and R. J. Hathaway, “Some notes on alternating optimization,” in *Advances in Soft Computing*. Springer Berlin / Heidelberg, 2002, pp. 187–195.

Joseph Szurley received his B.A. degree in physics from the University of California at Berkeley (2005) his M.Sc. in electrical engineering with honors from California State University Long Beach (2010) and his PhD in engineering science from KU Leuven, Belgium (2015). His research interests include audio signal processing, wireless (acoustic) sensor networks and distributed signal and speech enhancement. In 2003 he worked at Lawrence Berkeley National Laboratory in the Atomic Molecular and Optical Sciences (AMSO) division. From 2004-2007 he was a research assistant for Takeda Pharmaceuticals. From 2009-2010 he worked at Conexant Systems, Inc. in the product engineering group. Currently he is a research scientist at Bosch Research and Technology Center focusing on audio analytics for medium scale sensor deployments.

Alexander Bertrand (M08, SM'16) received the Ph.D. degree in Engineering Sciences (Electrical Engineering) from KU Leuven, Belgium, in 2011. He is currently assistant professor with the Electrical Engineering Department, KU Leuven, doing research in signal processing for neural engineering and distributed signal estimation.. He was a visiting researcher at University of California, Los Angeles, at Imec-NL Eindhoven, and at University of California, Berkeley. He received the 2012 FWO/IBM Award, and the 2013 KU Leuven Research Council Award. He is currently a board member in the Benelux chapter of the IEEE Signal Processing Society. He has served as a technical program committee member for EUSIPCO and IEEE GlobalSIP, as lead guest editor for Signal Processing, and as Associate Editor for the Biomedical Data Processing Track at IEEE EMBC 2016.

Marc Moonen (M'94, SM'06, F'07) is a Full Professor at the Electrical Engineering Department of KU Leuven, where he is heading a research team working in the area of numerical algorithms and signal processing for digital communications, wireless communications, DSL and audio signal processing.

He received the 1994 KU Leuven Research Council Award, the 1997 Alcatel Bell (Belgium) Award (with Piet Vandaele), the 2004 Alcatel Bell (Belgium) Award (with Raphael Cendrillon), and was a 1997 Laureate of the Belgium Royal Academy of Science. He received journal best paper awards from the IEEE Transactions on Signal Processing (with Geert Leus and with Daniele Giacobello) and from Elsevier Signal Processing (with Simon Doclo).

He was chairman of the IEEE Benelux Signal Processing Chapter (1998-2002), a member of the IEEE Signal Processing Society Technical Committee on Signal Processing for Communications, and President of EURASIP (European Association for Signal Processing, 2007-2008 and 2011-2012).

He has served as Editor-in-Chief for the EURASIP Journal on Applied Signal Processing (2003-2005), Area Editor for Feature Articles in IEEE Signal Processing Magazine (2012-2014), and has been a member of the editorial board of IEEE Transactions on Circuits and Systems II, IEEE Signal Processing Magazine, Integration-the VLSI Journal, EURASIP Journal on Wireless Communications and Networking, and Signal Processing. He is currently a member of the editorial board of EURASIP Journal on Advances in Signal Processing.