Distributed Adaptive Node-Specific Signal Estimation in Fully Connected Sensor Networks—Part I: Sequential Node Updating

Alexander Bertrand, *Student Member, IEEE*, and Marc Moonen, *Fellow, IEEE*

***Abstract*—We introduce a distributed adaptive algorithm for linear minimum mean squared error (MMSE) estimation of node-specific signals in a fully connected broadcasting sensor network where the nodes collect multichannel sensor signal observations. We assume that the node-specific signals to be estimated share a common latent signal subspace with a dimension that is small compared to the number of available sensor channels at each node. In this case, the algorithm can significantly reduce the required communication bandwidth and still provide the same optimal linear MMSE estimators as the centralized case. Furthermore, the computational load at each node is smaller than in a centralized architecture in which all computations are performed in a single fusion center. We consider the case where nodes update their parameters in a sequential round robin fashion. Numerical simulations support the theoretical results. Because of its adaptive nature, the algorithm is suited for real-time signal estimation in dynamic environments, such as speech enhancement with acoustic sensor networks.**

***Index Terms*—Adaptive estimation, distributed estimation, wireless sensor networks (WSNs).**

I. Introduction

In a sensor network [1] a general objective is to utilize all sensor signal observations available in the entire network to perform a certain task, such as the estimation of a parameter or signal. Gathering all observations in a fusion center to calculate an optimal estimate may however require a large communication bandwidth and computational power. This approach is often referred to as centralized fusion or estimation. An alternative is a distributed approach where each node has its own processing unit and the estimation relies on distributed processing and cooperation. This approach is preferred, especially so when it is scalable in terms of its communication bandwidth requirement and computational complexity.

In many sensor network estimation frameworks, the sensor signal observations are used to estimate a common network-wide desired parameter or signal, denoted here by $\mathbf{d}$. This means that all nodes contribute to a common goal, i.e., the estimation of the globally defined variable $\mathbf{d}$, which is the same for all nodes (see for example [2]–[8]). This can be viewed as a special case of the more general problem, which is considered here, where each node in the network estimates a different node-specific desired signal, i.e., node k estimates the locally defined signal $\mathbf{d}_k$. This means that all nodes have a different local objective, which they pursue through cooperation with other nodes. We describe a distributed adaptive node-specific signal estimation (DANSE) algorithm that operates in an ideal fully connected network. The nodes broadcast compressed multichannel sensor signal observations that can be captured by all other nodes in the network, possibly with the help of relay nodes. The computational load is distributed over the different nodes in the network.

The DANSE algorithm is designed for the case where the node-specific desired signals share a common (unknown) latent signal subspace. If this signal space has a small dimension compared to the number of available sensor channels at each node, the DANSE algorithm exploits this common interest of the nodes to significantly compress the data to be broadcast, and yet converge to the optimal linear minimum mean squared error (MMSE) estimators as if all sensor signal observations were available at each node. Although the DANSE algorithm implicitly assumes a specific structure in the relationship between the desired signals of the different nodes, it is noted that the actual parameters of these latent dependencies are not assumed to be known, i.e., nodes do not know how their desired signal is related to the desired signals of other nodes. The model that is assumed in the DANSE algorithm naturally emerges in adaptive signal estimation problems in dynamic scenarios where the target signal statistics and the transfer functions to the sensors are not known and may change during operation of the algorithm. Therefore, the original target signal cannot be recovered, and so an option is then to let the nodes optimally estimate the signal as it is observed locally by the node's sensors. In this case, the desired signals of the different nodes are differently filtered

Manuscript received October 21, 2009; accepted March 21, 2010. Date of publication June 10, 2010; date of current version September 15, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ta-Sung Lee. The work of A. Bertrand was supported by a Ph.D. grant of the I.W.T. (Flemish Institute for the Promotion of Innovation through Science and Technology). This work was carried out at the ESAT Laboratory of Katholieke Universiteit Leuven, in the frame of K.U. Leuven Research Council CoE EF/05/006 Optimization in Engineering (OPTEC), Concerted Research Action GOA-AMBioRICS, Concerted Research Action GOA-MaNet, the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office IUAP P6/04 (DYSCO, "Dynamical systems, control and optimization," 2007–2011), and Research Project FWO nr. G.0600.08 ("Signal processing and network design for wireless acoustic sensor networks"). The scientific responsibility is assumed by its authors.

The authors are with the Department of Electrical Engineering (ESAT-SCD/SISTA), Katholieke Universiteit Leuven, B-3001 Leuven, Belgium (e-mail: alexander.bertrand@esat.kuleuven.be; marc.moonen@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2010.2052612

versions of the same target signal, i.e., they share a common latent signal subspace.

Because of its adaptive nature, the DANSE algorithm is suited for real-time applications in dynamic environments. Typical applications are vibration monitoring, wireless acoustic sensor networks (for surveillance, video conferencing, domotics, audio recording...), and noise reduction in hearing aids with external sensor nodes and/or cooperation between multiple hearing aids [9], [10]. Node-specific estimation is particularly important in applications where a target signal needs to be estimated as it is observed at a specific sensor position. For instance, in acoustic surveillance, it is often required to be able to locate a sound source, so spatial information in the observations of different nodes must be retained in the estimation process. In cooperating hearing aids, it is important to estimate the signal as it impinges at the hearing aid itself, to preserve the auditory cues for directional hearing [11], [12].

The DANSE algorithm is based on linear compression of multichannel sensor signal observations. Linear compression of sensor signal observations for data fusion has been the topic of earlier work, e.g., [5]–[8]. The presented techniques, however, assume prior knowledge of the intra- and intersensor (cross-)correlation structure in the entire network. This must be obtained by *a priori* training using all uncompressed sensor signal observations, or must be derived from a specific data model. Such assumptions make it difficult to apply the resulting algorithms in adaptive networks or dynamic environments where the statistics of the desired signals or sensor signals may change. The DANSE algorithm can adapt to these changes because nodes estimate and reestimate all required statistical quantities on the compressed data during operation. For this, we assume that each node can adaptively estimate the cross correlation between its local sensor signals and its desired signal. It is noted that the acquisition of these signal statistics is often difficult or impossible, since the target signal is assumed to be unknown. However, we will explain that in particular cases, it is possible to estimate the required statistics, e.g., when the target signal has an ON–OFF behavior (such as speech signals), or when the target source periodically transmits *a priori* known training sequences. In cases where the local statistics cannot be estimated adaptively, the DANSE algorithm can still be used in a semi-adaptive context, i.e., scenarios with static noise statistics but with changing target signal statistics or vice versa, assuming that the static correlation structure is *a priori* known.

In [13], a batch-mode description of the DANSE algorithm was briefly introduced. In this paper, we provide more details, i.e., we include a convergence proof and introduce a truly adaptive version. In addition, we address implementation aspects, and provide extensive simulation results, both in batch mode and in a dynamic scenario. We only consider the case where nodes update their parameters in a sequential round robin fashion. The case where nodes update simultaneously or asynchronously is treated in a companion paper [14]. In [10], a pruned version of the DANSE algorithm has been used for microphone-array based speech enhancement in binaural hearing aids, where it was referred to as distributed multichannel Wiener filtering. In this application, two hearing aids in a binaural configuration exchange a linear combination of their microphone signals to estimate the target sound that is recorded by their reference microphone. Convergence of the two-node system has been proven for the special case where there is a single target speaker. The more general DANSE algorithm provided in this paper allows for a nontrivial extension to a scenario with multiple target speakers and a network with more than two nodes. Using extra acoustic sensor nodes that communicate with the hearing aids generally improves the noise reduction performance, since the acoustic sensors physically cover a larger area [9].

The paper is organized as follows. The problem formulation and notation are presented in Section II. In Section III, we first address the simple case in which the node-specific desired signals are scaled versions of each other and we prove convergence of the DANSE algorithm to the optimal linear MMSE estimators when nodes update their parameters sequentially. In Section IV, this algorithm is generalized to the case in which the node-specific desired signals share a common latent Q-dimensional signal subspace. In Section V, we address some implementation details of DANSE and we study the complexity of the algorithm. Finally, Section VI illustrates the convergence results with numerical simulations. Conclusions are given in Section VII.

II. Problem Formulation and Notation

A. Node-Specific Linear MMSE Estimation

We consider an ideal fully connected network with sensor nodes $\{1,\ldots,J\} = \mathcal{J}$, in which data broadcast by a node can be captured by all other $(J-1)$ nodes in the network through an ideal link. Node k collects observations of a complex[1] valued M_k-channel signal $\mathbf{y}_k[t]$, where $t \in \mathbb{N}$ is the discrete time index, and where $\mathbf{y}_k[t]$ is an M_k-dimensional column vector. Each channel $y_{kn}[t]$, $\forall n \in \{1,\ldots,M_k\}$, of the signal $\mathbf{y}_k[t]$ corresponds to a sensor signal to which node k has access. We assume that all signals are stationary and ergodic. In practice, the stationarity and ergodicity assumption can be relaxed to short-term stationarity and ergodicity, in which case the theory should be applied to finite signal segments that are assumed to be stationary and ergodic. For the sake of an easy exposition, we will omit the time index when referring to a signal, and we will only write the time index when referring to one specific observation, i.e., $\mathbf{y}_k[t]$ is the observation of the signal $\mathbf{y}_k$ at time t. We define $\mathbf{y}$ as the M-channel signal in which all $\mathbf{y}_k$ are stacked, where $M = \sum_{k=1}^{J} M_k$. This scenario is described in Fig. 1.

It is noted that this problem formulation also allows for hierarchical network architectures, in which the sensors are grouped in J clusters. The sensors of a specific cluster then transmit their observations to a nearby fusion center, i.e., a "higher level" node. The J fusion centers then correspond to the J nodes in the above framework, and the collected observations in sensor cluster k correspond to the M_k-channel signals $\mathbf{y}_k$ as explained above. Fig. 2 shows such a scenario for a network with three fusion centers $(J = 3)$.

We first consider the centralized estimation problem, i.e., we assume that each node has access to the observations of the entire M-channel signal $\mathbf{y}$. This corresponds to the case where

[1]Throughout this paper, all signals are assumed to be complex valued to permit frequency-domain descriptions.

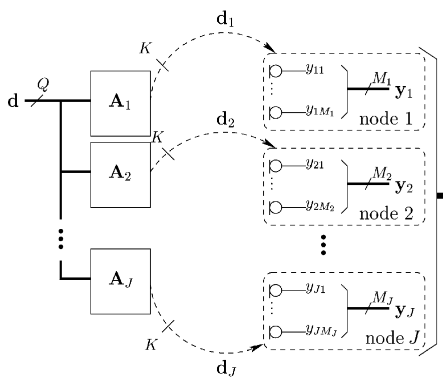


Fig. 1. Description of the scenario. The network contains J sensor nodes, $k = 1 \ldots J$, where node k collects M_k-channel sensor signal observations and estimates a node-specific desired signal $\mathbf{d}_k$, which is a mixture of the Q channels of a common latent signal $\mathbf{d}$.

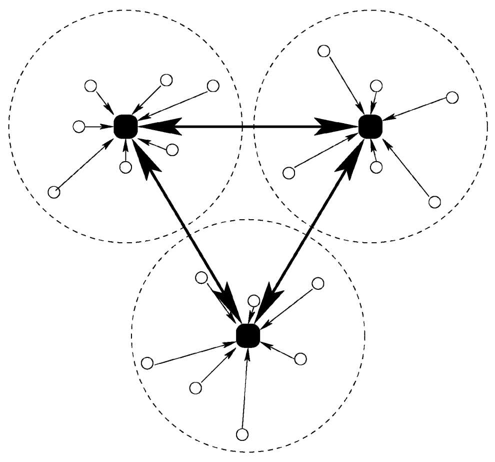


Fig. 2. A hierarchical architecture with 3 fusion centers $(J = 3)$, each one collecting sensor signals from nearby sensors.

nodes broadcast their uncompressed observations to all other nodes. In Sections III and IV, the general goal will be to compress the broadcast signals, while preserving the estimation performance of this centralized estimator. The objective for node k is to estimate a complex valued node-specific signal d_k, referred to as the desired signal, from the observations of $\mathbf{y}$. We consider the general case where d_k is not an observed signal, i.e., it is assumed to be unknown, as it is the case in signal enhancement (e.g., in speech enhancement, d_k is the speech component in a noisy microphone signal). Node k uses a linear estimator $\mathbf{w}_k$ to estimate d_k as $\overline{d}_k = \mathbf{w}_k^H \mathbf{y}$ where $\mathbf{w}_k$ is a complex valued M-dimensional vector, and where superscript H denotes the conjugate transpose operator. We assume that the M-channel signal $\mathbf{y}$ is correlated to the node-specific desired signals, but unlike [6], [8], we do not restrict ourselves to any data model generating the sensor signals, nor do we make any assumptions on the probability distributions of the involved signals. We consider linear MMSE estimation based on a node-specific estimator $\hat{\mathbf{w}}_k$, i.e.

$$\hat{\mathbf{w}}_k = \arg\min_{\mathbf{w}_k} E\left\{\left|d_k - \mathbf{w}_k^H \mathbf{y}\right|^2\right\} \tag{1}$$

with $E\{.\}$ the expected value operator. Assuming that the correlation matrix $\mathbf{R}_{yy} = E\{\mathbf{y}\mathbf{y}^H\}$ has full rank,[2] the unique solution of (1) is [15]:

$$\hat{\mathbf{w}}_k = \mathbf{R}_{yy}^{-1} \mathbf{r}_{yd_k} \tag{2}$$

with $\mathbf{r}_{yd_k} = E\{\mathbf{y} d_k^*\}$, where d_k^* denotes the complex conjugate of d_k. Based on the assumption that the signals are ergodic, $\mathbf{R}_{yy}$ and $\mathbf{r}_{yd_k}$ can be estimated by time averaging. The $\mathbf{R}_{yy}$ is directly estimated from the sensor signal observations. Since d_k is assumed to be unknown, the estimation of the correlation vector $\mathbf{r}_{yd_k}$ has to be done indirectly, based on specific strategies, e.g., by exploiting the ON–OFF behavior of the target signal (e.g., for speech enhancement [9], [10]), by using training sequences, or by using partial prior knowledge when the estimation is performed in a semi-adaptive context. We will provide more details on these strategies in Section V-A. In the sequel, we assume that $\mathbf{r}_{yd_k}$ can be estimated during operation of the algorithm.

In the above estimation procedure, temporal correlation appears to be ignored. However, differently delayed versions of one or more sensor signals at node k can be added to the channels of $\mathbf{y}_k$, to also exploit the temporal information in the signals. For example, assume that node k has access to 4 sensor signals. Then each of these signals is delayed with 1, up to $N-1$ sample delays, resulting in $N-1$ extra (delayed) channels. In this case, the dimension of $\mathbf{y}_k$ is $M_k = 4N$.

It is noted that our problem statement differs from [2]–[4], where each node collects different spatio–temporal observations of two correlated signals y and d. The objective is then to find the best common linear fit between these observations, with a single set of coefficients $\mathbf{w}$, which is assumed to be the same for each node. Since the coefficients in $\mathbf{w}$ are of interest, only the locally estimated $\mathbf{w}$'s must be shared between nodes, whereas the sensor observations themselves are only used locally to update the estimate of $\mathbf{w}$. Since all nodes are assumed to estimate the same set of coefficients, incremental or diffusive averaging strategies can be used.

B. Common Latent Signal Subspace

In our problem statement, each node k only collects observations of $\mathbf{y}_k$ which corresponds to a subset of the channels of the full signal $\mathbf{y}$. To find the optimal MMSE solution (2), each node k therefore in principle has to broadcast its observations of $\mathbf{y}_k$ to all other nodes in the network, which requires a large communication bandwidth. One possibility to reduce the required bandwidth is to broadcast only a few linear combinations of the components of the $\mathbf{y}_k$ observations instead of all M_k components. Finding the optimal linear compression is often a nontrivial task, and in general this will not lead to the optimal solutions (2). In many practical cases, however, the d_k signals share a common latent signal subspace, and then this can be exploited in the compression. The most simple case is when all $d_k = d$, i.e., the desired signal is the same for all nodes. We will first handle the slightly more general case where all d_k are scaled versions of a common latent single-channel signal d. For this

[2]This assumption is mostly satisfied in practice because of a noise component at every sensor that is independent of other sensors, e.g., thermal noise. If not, pseudoinverses should be used. A further comment on the rank-deficient case is made in Section IV-C.

scenario, we will introduce the DANSE_1 algorithm, in which the data to be broadcast by each node k is compressed by a factor M_k. Despite this compression, the algorithm converges to the optimal node-specific solution (2) at every node as if no compression were used for the broadcasts.

This scenario can then be extended to the more general case where the desired signals share a common Q-dimensional signal subspace, i.e.

$$\forall\, k \in \mathcal{J} : d_k = \mathbf{a}_k^T \mathbf{d} \tag{3}$$

with $\mathbf{a}_k$ defining an unknown Q-dimensional complex vector, and $\mathbf{d}$ a latent complex valued Q-channel signal defining the Q-dimensional signal subspace that contains all d_k signals. This model applies to situations where the desired signal is generated by multiple latent processes simultaneously (e.g., measuring vibrations when there are multiple exciters, or recording a conversation between multiple speakers [9]). Since the statistics of the latent signals as well as the propagation properties to the different sensors are generally unknown, the signal estimation procedure can only use statistics that can be obtained from the local sensor signal observations. The desired signal of each node is then the linear mixture of the latent target signals as locally observed by a reference sensor.

In the sequel, we consider the general case where node k estimates a K-channel desired signal $\mathbf{d}_k$

$$\forall\, k \in \mathcal{J} : \mathbf{d}_k = \mathbf{A}_k \mathbf{d} \tag{4}$$

with $\mathbf{A}_k$ a $K \times Q$ complex valued matrix. This data model is depicted in Fig. 1. It is noted that the matrix $\mathbf{A}_k$ and the latent signal $\mathbf{d}$ are assumed to be unknown, i.e., nodes do not know how their node-specific desired signals $\mathbf{d}_k$ are related to each other. Since we also consider complex valued signals, (4) can correspond to a frequency domain description of a convolutive mixture in the time domain, as in [9], [10]. Expression (4) then defines a different estimation problem for each specific frequency. This yields frequency dependent estimators $\mathbf{w}_k(f)$, which translate to multitap filters in the time domain.

Notice that, if $K \geq Q$, the desired signal $\mathbf{d}_k$ spans the complete signal subspace defined by the Q-channel signal $\mathbf{d}$ (provided that the $K \times Q$ matrix $\mathbf{A}_k$ has full rank). If this holds for each node in the network, we will show that the data to be broadcast by node k can be compressed by a factor M_k/Q. This means that node k only needs to broadcast Q linear combinations of the components of its observations of $\mathbf{y}_k$, while the optimal node-specific solution (2) is still obtained at all nodes. Notice that in practical applications, the actual signal(s) of interest can be a subset of the entries in $\mathbf{d}_k$, in which case the other entries should be seen as auxiliary channels to capture the latent Q-dimensional signal subspace that contains the $\mathbf{d}_k$'s. For instance, consider the case where nodes estimate the target signal as observed by their reference sensor, i.e., node k estimates the node-specific desired signal d_k as in (3). Node k then selects $K-1$ extra auxiliary reference sensors, and also estimates the target signal as it arrives on these sensors. The resulting K-channel desired signal $\mathbf{d}_k$ then spans the complete signal subspace if $K \geq Q$.

III. DANSE With Single-Channel Broadcast Signals ($K = 1$)

The algorithm introduced in this paper is an iterative scheme referred to as distributed adaptive node-specific signal estimation (DANSE), since its objective is to estimate a node-specific signal at each node in a distributed fashion. In the general scheme, each node k broadcasts $\min\{K, M_k\}$-component compressed sensor signal observations. We will refer to this as DANSE_K, where the subscript K refers to the number of channels of the broadcast signals. For the sake of an easy exposition, we first introduce the DANSE algorithm for the simple case where $K = 1$ and we will show that DANSE_1 converges to the optimal filters if $Q = 1$, i.e., if the single-channel desired signals d_k are nonzero scaled versions of the same latent single-channel signal d. In Section IV we generalize this to the more general DANSE_K algorithm, and we will show that this algorithm converges to the optimal filters if $Q = K$ and if all $\mathbf{A}_k$ in (4) have rank K.

A. DANSE_1 *Algorithm*

The goal for each node k is to estimate the signal d_k with a linear estimator that uses all observations in the entire network, i.e., $\overline{d}_k = \mathbf{w}_k^H \mathbf{y}$. We aim to obtain the MMSE solutions (2), without the need for each node to broadcast all M_k components of the $\mathbf{y}_k$ observations. For this, we define a partitioning of the estimator $\mathbf{w}_k$ as $\mathbf{w}_k = [\mathbf{w}_{k1}^T \ldots \mathbf{w}_{kJ}^T]^T$ with $\mathbf{w}_{kq}$ denoting the M_k-dimensional subvector of $\mathbf{w}_k$ that is applied to $\mathbf{y}_q$, and with superscript T denoting the transpose operator. In this way, (1) is equivalent to

$$\hat{\mathbf{w}}_k = \begin{bmatrix} \hat{\mathbf{w}}_{k1} \\ \vdots \\ \hat{\mathbf{w}}_{kJ} \end{bmatrix} = \underset{\{\mathbf{w}_{k1},\ldots,\mathbf{w}_{kJ}\}}{\arg\min}\; E\left\{ \left| d_k - \sum_{q=1}^{J} \mathbf{w}_{kq}^H \mathbf{y}_q \right|^2 \right\}. \tag{5}$$

Since node k only has access to the sensor signal observations of $\mathbf{y}_k$, it can only control a specific part of the estimator $\mathbf{w}_k$, namely $\mathbf{w}_{kk}$. In the DANSE_1 algorithm, each node k broadcasts the output of this partial estimator, i.e., observations of the compressed signal $z_k = \mathbf{w}_{kk}^H \mathbf{y}_k$. This reduces the data to be broadcast by a factor M_k. It is noted that $\mathbf{w}_{kk}$ acts both as a compressor and as a part of the estimator $\mathbf{w}_k$, i.e., the observations of the compressed signal z_k that is broadcast by node k is also used in the estimation of d_k at node k itself.

A node k now has access to $M_k + J - 1$ input channels, i.e., its own M_k sensor signals and $(J-1)$ signals that it receives from the other nodes. Node k will compute the optimal linear combiner of these $M_k + J - 1$ input channels to estimate d_k. The coefficient that is applied to the signal observations of z_q at node k is denoted by g_{kq}. A schematic illustration of this scheme (for $J = 3$) is shown in Fig. 3. Notice that there is no decompression involved, i.e., node k does not expand the observations of the z_q signal, but only scales these with a scaling

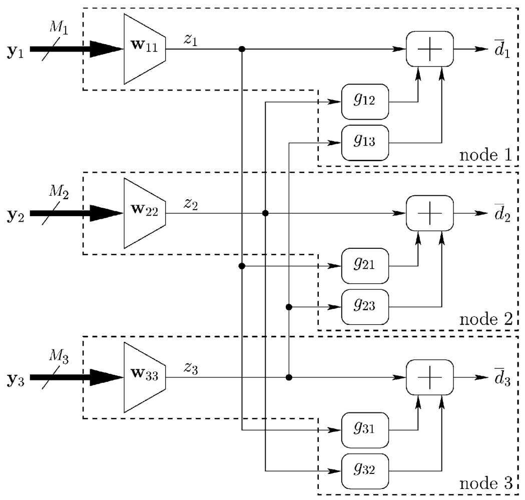


Fig. 3. The DANSE_1 scheme with three nodes ($J = 3$). Each node k estimates a signal d_k using its own M_k-channel sensor signal observations, and two single-channel signals broadcast by the other two nodes.

factor g_{kq}. As visualised in Fig. 3, the parametrization of the $\mathbf{w}_k$ now effectively applied at node k is therefore

$$\widetilde{\mathbf{w}}_k = \begin{bmatrix} g_{k1}\mathbf{w}_{11} \\ g_{k2}\mathbf{w}_{22} \\ \vdots \\ g_{kJ}\mathbf{w}_{JJ} \end{bmatrix} \tag{6}$$

i.e., each $\mathbf{w}_k$ is now defined by the set of $\mathbf{w}_{qq}$'s ($q \in \mathcal{J}$) together with a vector $\mathbf{g}_k = [g_{k1} \dots g_{kJ}]^T$, defining the scaling parameters. We use a tilde to indicate that the estimator is parametrized according to (6), which defines a solution space for $(\mathbf{w}_1, \dots, \mathbf{w}_J)$ with a specific structure. In this parametrization, node k can only manipulate the parameters $\mathbf{w}_{kk}$ and $\mathbf{g}_k$. In the sequel, we set $g_{kk} = 1$ to remove the ambiguity in $g_{kk}\mathbf{w}_{kk}$ (hence g_{kk} is omitted in Fig. 3). Notice that the solution space $(\widetilde{\mathbf{w}}_1, \dots, \widetilde{\mathbf{w}}_J)$ of DANSE_1 is $(M + J(J-1))$-dimensional, which is smaller[3] than the original MJ-dimensional solution space $(\mathbf{w}_1, \dots, \mathbf{w}_J)$ corresponding to the centralized algorithm, i.e., the solution space of the optimization problem (1). Still, the goal of the DANSE_1 algorithm is to iteratively update the parameters of (6) until $(\widetilde{\mathbf{w}}_1, \dots, \widetilde{\mathbf{w}}_J) = (\hat{\mathbf{w}}_1, \dots, \hat{\mathbf{w}}_J)$.

In the sequel, we will use the following notation and definitions. In general, we will use X^i to denote X at iteration i, where X can be a signal or a parameter. The J-channel signal $\mathbf{z}$ is defined as $\mathbf{z} = [z_1 \dots z_J]^T$. We define $\mathbf{g}_{k-q}$ as the vector $\mathbf{g}_k$ with entry g_{kq} omitted. Similarly, we define $\mathbf{z}_{-k}$ as the vector $\mathbf{z}$ with entry z_k omitted.

At every iteration i in the DANSE_1 algorithm, one specific node k will update its local parameters $\mathbf{w}_{kk}^i$ and $\mathbf{g}_{k-k}^i$, by solving its local node-specific MMSE problem with respect to its input signals, consisting of its own sensor signal observations $\mathbf{y}_k$ and the compressed signal observations of $\mathbf{z}_{-k}^i$, i.e., it solves

$$\begin{bmatrix} \mathbf{w}_{kk}^{i+1} \\ \hline \mathbf{g}_{k-k}^{i+1} \end{bmatrix} = \underset{\mathbf{w}_{kk}, \mathbf{g}_{k-k}}{\arg\min}\, E\left\{ \left| d_k - \left[\mathbf{w}_{kk}^H | \mathbf{g}_{k-k}^H\right] \begin{bmatrix} \mathbf{y}_k \\ \hline \mathbf{z}_{-k}^i \end{bmatrix} \right|^2 \right\}. \tag{7}$$

Let $\widetilde{\mathbf{y}}_k^i$ denote the stacked version of the local input signals at node k, i.e.

$$\widetilde{\mathbf{y}}_k^i = \begin{bmatrix} \mathbf{y}_k \\ \hline \mathbf{z}_{-k}^i \end{bmatrix}. \tag{8}$$

Then the solution of (7) is

$$\begin{bmatrix} \mathbf{w}_{kk}^{i+1} \\ \hline \mathbf{g}_{k-k}^{i+1} \end{bmatrix} = \left(\mathbf{R}_{\tilde{y}_k \tilde{y}_k}^i\right)^{-1} \mathbf{r}_{\tilde{y}_k d_k}^i \tag{9}$$

with

$$\mathbf{R}_{\tilde{y}_k \tilde{y}_k}^i = E\left\{\widetilde{\mathbf{y}}_k^i \widetilde{\mathbf{y}}_k^{iH}\right\} \tag{10}$$

$$\mathbf{r}_{\tilde{y}_k d_k}^i = E\left\{\widetilde{\mathbf{y}}_k^i d_k^*\right\}. \tag{11}$$

Since there is no decompression involved, the local estimation problems (7) have a smaller dimension than the original network-wide estimation problems (1), $\forall k \in \mathcal{J}$, i.e., the matrix $\mathbf{R}_{\tilde{y}_k \tilde{y}_k}^i$ is smaller than the $\mathbf{R}_{yy}$ matrix in (2).

We define a block size B which denotes the number of observations that the nodes collect in between two successive node updates, i.e., in between two increments of i. The DANSE_1 algorithm now consists of the following steps:

1) Initialize: $i \leftarrow 0$, $u \leftarrow 1$
   Initialize $\mathbf{w}_{kk}^0$ and $\mathbf{g}_{k-k}^0$ with random vectors, $\forall k \in \mathcal{J}$.
2) Each node $k \in \mathcal{J}$ performs the following operation cycle:
   - Collect the sensor observations $\mathbf{y}_k[iB + n]$, $n = 0 \dots B - 1$.
   - Compress these M_k-dimensional observations to

$$z_k^i[iB + n] = \mathbf{w}_{kk}^{iH} \mathbf{y}_k[iB + n], \quad n = 0 \dots B - 1. \tag{12}$$

   - Broadcast the compressed observations $z_k^i[iB + n]$, $n = 0 \dots B - 1$, to the other nodes.
   - Collect the $(J - 1)$-dimensional data vectors $\mathbf{z}_{-k}^i[iB + n]$, $n = 0 \dots B - 1$, which are stacked versions of the compressed observations received from the other nodes.
   - Update the estimates of $\mathbf{R}_{\tilde{y}_k \tilde{y}_k}^i$ and $\mathbf{r}_{\tilde{y}_k d_k}^i$, by including the newly collected data.[4]
   - Update the node-specific parameters:

$$\begin{bmatrix} \mathbf{w}_{kk}^{i+1} \\ \hline \mathbf{g}_{k-k}^{i+1} \end{bmatrix} = \begin{cases} \left(\mathbf{R}_{\tilde{y}_k \tilde{y}_k}^i\right)^{-1} \mathbf{r}_{\tilde{y}_k d_k}^i & \text{if } k = u \\ \begin{bmatrix} \mathbf{w}_{kk}^i \\ \hline \mathbf{g}_{k-k}^i \end{bmatrix} & \text{if } k \neq u. \end{cases} \tag{13}$$

[3]It is assumed here that $J < M$, i.e., $M_k \geq 1, \forall k \in \mathcal{J}$, and there is at least one node k for which $M_k > 1$.

[4]In Section V-A, we will suggest some possible strategies to estimate these parameters.

- Compute the estimate of $d_k[iB+n], n=0\ldots B-1$, as

$$\overline{d}_k[iB+n] = \mathbf{w}_{kk}^{i+1H}\mathbf{y}_k[iB+n] + \mathbf{g}_{k-k}^{i+1H}\mathbf{z}_{-k}^i[iB+n]. \quad (14)$$

3) $i \leftarrow i+1$.
4) $u \leftarrow (u \bmod J)+1$.
5) Return to step 2)

Remark I: Notice that the different iterations are spread out over time. Therefore, iterative characteristics of the algorithm do not have an impact on the amount of data that is transmitted, i.e., each sample is only broadcast once since the time index in (12) and (14) shifts together with the iteration index.

Remark II: In the above algorithm description, it is not mentioned how the correlation matrix $\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i$ and the correlation vector $\mathbf{r}_{\tilde{y}_k d_k}^i$ should be estimated. This estimation process depends on the application and the signals involved. In Section V-A, we will suggest some possible strategies to estimate $\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i$ and $\mathbf{r}_{\tilde{y}_k d_k}^i$.

Remark III: It is noted that, when a node k updates its node-specific parameters $\mathbf{w}_{kk}^i$ and $\mathbf{g}_{k-k}^i$, the signal statistics of z_k change, i.e., z_k^i changes to z_k^{i+1}. Therefore, the next node to perform an update needs a sufficient number of observations of z_k to reliably estimate the correlation coefficients involving this signal. Therefore, the block-length B should be chosen large enough.

B. Convergence and Optimality of DANSE_1 *if* $Q=1$ *and Nonzero Desired Signals*

We now assume that all d_k are a nonzero scaled version of the same signal d, i.e., $d_k=\rho_k d$, with ρ_k a nonzero complex scalar but unknown to the individual nodes. Formula (2) shows that in this case, all $\hat{\mathbf{w}}_k$ are parallel, i.e.

$$\hat{\mathbf{w}}_k = \rho_{kq}\hat{\mathbf{w}}_q \quad \forall\, k,q \in \mathcal{J} \quad (15)$$

with $\rho_{kq}=(\rho_k^*/\rho_q^*)$. Therefore, the set $(\hat{\mathbf{w}}_1,\ldots,\hat{\mathbf{w}}_J)$ belongs to the solution space used by DANSE_1, as specified by (6), i.e., $(\hat{\mathbf{w}}_1,\ldots,\hat{\mathbf{w}}_J) \in (\widetilde{\mathbf{w}}_1,\ldots,\widetilde{\mathbf{w}}_J)$.

In the theoretical convergence analysis in the sequel, we assume that the correlation matrices $\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i$ and the correlation vectors $\mathbf{r}_{\tilde{y}_k d_k}^i$, $\forall k \in \mathcal{J}$, are perfectly estimated, i.e., as if they are computed over an infinite observation window. Under this assumption, the following theorem guarantees convergence and optimality of the DANSE_1 algorithm.

Theorem III.1: If the sensor signal correlation matrix $\mathbf{R}_{yy}$ has full rank, and if $d_k=\rho_k d, \forall k \in \mathcal{J}$, with d a complex valued single-channel signal and $\rho_k \in \mathbb{C}\setminus\{0\}$, then the DANSE_1 algorithm converges for any initialization of its parameters to the MMSE solution (2) for all $k \in \mathcal{J}$.

Before proving this theorem, we introduce some additional notation. The vector $\mathbf{w}$ (without subscript) denotes the stacked vector of all $\mathbf{w}_{kk}$ vectors, i.e.

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_{11} \\ \mathbf{w}_{22} \\ \vdots \\ \mathbf{w}_{JJ} \end{bmatrix}. \quad (16)$$

We also define the following MSE cost functions corresponding to node k:

$$J_k(\mathbf{w}_k) = E\left\{\left|d_k - \mathbf{w}_k^H\mathbf{y}\right|^2\right\} \quad (17)$$

$$\widetilde{J}_k(\mathbf{w},\mathbf{g}_k) = J_k(\widetilde{\mathbf{w}}_k) \quad (18)$$

where $\widetilde{\mathbf{w}}_k$ is defined from $\mathbf{w}$ and $\mathbf{g}_k$ as in (6). Notice that $\mathbf{g}_k$ contains the entry g_{kk}, which is a fictitious variable that is never actually computed by the DANSE_K algorithm. We define $F_k(\mathbf{w}^i)$ as the function that generates $\mathbf{w}_{kk}^{i+1}$ according to (9), i.e.

$$F_k(\mathbf{w}^i) = \mathbf{w}_{kk}^{i+1} = \left[\mathbf{I}_{M_k}|\mathbf{O}_{M_k\times(J-1)}\right]\left(\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i\right)^{-1}\mathbf{r}_{\tilde{y}_k d_k}^i \quad (19)$$

with $\mathbf{I}_P$ denoting a $P\times P$ identity matrix and $\mathbf{O}_{P\times Q}$ denoting an all-zero $P\times Q$ matrix. It is noted that the right-hand side of (19) depends on all entries of the argument $\mathbf{w}^i$ through the signal $\mathbf{z}_{-k}^i$, which is not explicitly revealed in this expression.

The proof of Theorem III.1 provided here differs from the proof in [10], where a scheme similar to DANSE_1 with $J=2$ has been proved to converge to the optimal solution. Unlike the proof in [10], our proof allows for a generalization to the DANSE_K case with $K>1$, it allows $J>2$, and provides more insight in the convergence properties of the algorithm. We first prove the convergence statement of Theorem III.1, and then the optimality statement.

Proof of Convergence: We prove that the sequence $(\mathbf{w}^i)_{i\in\mathbb{N}}$ and the sequences $(\mathbf{g}_k^i)_{i\in\mathbb{N}}$ $\forall k \in \mathcal{J}$ converge to a limit point $\mathbf{w}^\infty$ and $\mathbf{g}_k^\infty$ respectively. When node k performs an update of its variables $\mathbf{w}_{kk}$ and $\mathbf{g}_{k-k}$ at iteration i, these are replaced by the solution of the local MMSE problem (7), repeated here for convenience:

$$\min_{\mathbf{w}_{kk},\mathbf{g}_{k-k}} E\left\{\left|d_k - \left[\mathbf{w}_{kk}^H|\mathbf{g}_{k-k}^H\right]\begin{bmatrix}\mathbf{y}_k \\ \hline \mathbf{z}_{-k}^i\end{bmatrix}\right|^2\right\}. \quad (20)$$

If another node q were to optimize the variables $\mathbf{w}_{kk}$ and $\mathbf{g}_{q-k}$ with respect to its own node-specific estimation problem, it would solve the problem

$$\min_{\mathbf{w}_{kk},\mathbf{g}_{q-k}} E\left\{\left|d_q - \left[\mathbf{w}_{kk}^H|\mathbf{g}_{q-k}^H\right]\begin{bmatrix}\mathbf{y}_k \\ \hline \mathbf{z}_{-k}^i\end{bmatrix}\right|^2\right\}. \quad (21)$$

Since $d_k^*=\rho_{kq}d_q^*$ with $\rho_{kq}=\rho_k^*/\rho_q^*$, the solution of (20) and (21) are identical up to a scalar ρ_{kq}. This means that an update of $\mathbf{w}_{kk}$ and $\mathbf{g}_{k-k}$ at node k, which is an optimization leading to a decrease of $\widetilde{J}_k$, will also lead to a decrease of $\widetilde{J}_q$ for any $q \in \mathcal{J}$ if node q were allowed to also perform a responding optimization of its $\mathbf{g}_q$. This shows that for any i (independent of the selection of the node u that actually performs an update at iteration i)

$$\forall\, k \in \mathcal{J}: \min_{\mathbf{g}_k}\widetilde{J}_k(\mathbf{w}^{i+1},\mathbf{g}_k) \le \min_{\mathbf{g}_k}\widetilde{J}_k(\mathbf{w}^i,\mathbf{g}_k). \quad (22)$$

Since all $\widetilde{J}_k$ have a lower bound, each sequence $(\min_{\mathbf{g}_k}\widetilde{J}_k(\mathbf{w}^i,\mathbf{g}_k))_{i\in\mathbb{N}}$ converges to a limit L_k, i.e.

$$\forall\, k\in\mathcal{J}: \\ i\to\infty: \min_{\mathbf{g}_k}\widetilde{J}_k(\mathbf{w}^{i+1},\mathbf{g}_k) = \min_{\mathbf{g}_k}\widetilde{J}_k(\mathbf{w}^i,\mathbf{g}_k) = L_k. \quad (23)$$

If we again assume that node k performs an update at iteration i, then because of the strict convexity of the cost function in (20), the following expression holds:

$$\min_{\mathbf{g}_k} \widetilde{J}_k(\mathbf{w}^{i+1}, \mathbf{g}_k) = \min_{\mathbf{g}_k} \widetilde{J}_k(\mathbf{w}^i, \mathbf{g}_k) \Leftrightarrow \mathbf{w}_{kk}^{i+1} = \beta_k \mathbf{w}_{kk}^i \tag{24}$$

with

$$\beta_k = \arg\min_{g_{kk}} \left(\min_{\mathbf{g}_{k-k}} \widetilde{J}_k(\mathbf{w}^i, \mathbf{g}_k) \right). \tag{25}$$

This shows that, after convergence of the sequences $(\min_{\mathbf{g}_k} \widetilde{J}_k(\mathbf{w}^i, \mathbf{g}_k))_{i\in\mathbb{N}}$, $\forall k \in \mathcal{J}$, any update of a $\mathbf{w}_{kk}$ must correspond to a scaling. Notice however that

$$F_k\left(\begin{bmatrix} \mathbf{w}_{11}^i \\ \vdots \\ \mathbf{w}_{JJ}^i \end{bmatrix}\right) = F_k\left(\begin{bmatrix} \beta_1\mathbf{w}_{11}^i \\ \vdots \\ \beta_J\mathbf{w}_{JJ}^i \end{bmatrix}\right) \tag{26}$$

i.e., a scaling of a $\mathbf{w}_{qq}$ in node q does not change the update of $\mathbf{w}_{kk}$ in node k, since the scaling is implicitly compensated in F_k by the parameter g_{kq}. This proves convergence of the sequence $(\mathbf{w}^i)_{i\in\mathbb{N}}$ to a limit point $\mathbf{w}^\infty$ and therefore also the sequences $(\mathbf{g}_k^i)_{i\in\mathbb{N}}$ must converge to a limit point $\mathbf{g}_k^\infty$, $\forall k \in \mathcal{J}$. Notice that after convergence, based on what was stated earlier

$$\forall\, k, q \in \mathcal{J} : \widetilde{\mathbf{w}}_{kq}^\infty = \rho_{kq}\mathbf{w}_{qq}^\infty \tag{27}$$

or equivalently

$$\forall\, k, q \in \mathcal{J} : g_{kq}^\infty = \rho_{kq}. \tag{28}$$

■

From the proof of convergence, one can also conclude that convergence of the cost functions J_k will be monotonic, when sampled at the iteration steps in which node k updates its parameters. Indeed, whenever node q optimizes its own local MMSE problem, it also optimizes the corresponding MMSE problem in node k, at least when the latter is allowed to perform a responding update of its parameter g_{kq}. This shows that the DANSE_1 algorithm is at least as fast as a centralized equivalent that would use an alternating optimization (AO) technique [16], which is often referred to as the nonlinear Gauss-Seidel algorithm [17], with partitioning following directly from the parameters J and M_k for each node.

Proof of Optimality: We now prove that $\widetilde{\mathbf{w}}_k^\infty$ is the solution of (1) for every node k, which is equivalent to proving that the gradient of J_k is zero when evaluated at equilibrium, i.e.

$$\forall\, k \in \mathcal{J} : \nabla J_k\left(\widetilde{\mathbf{w}}_k^\infty\right) = 0. \tag{29}$$

Because the solution of (20) sets the partial gradient of $\widetilde{J}_k$ with respect to $\mathbf{w}_{kk}$ to zero, we find that

$$\forall\, k \in \mathcal{J} : \nabla_{\mathbf{w}_{kk}} \widetilde{J}_k\left(\mathbf{w}^\infty, \mathbf{g}_k^\infty\right) = \nabla_{\mathbf{w}_{kk}} J_k\left(\widetilde{\mathbf{w}}_k^\infty\right) = 0. \tag{30}$$

Since $d_k^* = \rho_{kq} d_q^*$, we can show that

$$\forall\, k, q \in \mathcal{J} : \nabla_{\mathbf{w}_{qq}} J_q(\mathbf{w}_q) = 0 \Leftrightarrow \nabla_{\mathbf{w}_{kq}} J_k(\rho_{kq}\mathbf{w}_q) = 0. \tag{31}$$

Combining (30) and (31) yields

$$\forall\, k, q \in \mathcal{J} : \nabla_{\mathbf{w}_{kq}} J_k\left(\rho_{kq}\widetilde{\mathbf{w}}_q^\infty\right) = 0. \tag{32}$$

Notice that (27) is equivalent with

$$\forall\, k, q \in \mathcal{J} : \widetilde{\mathbf{w}}_k^\infty = \rho_{kq}\widetilde{\mathbf{w}}_q^\infty. \tag{33}$$

Substituting (33) in (32) yields

$$\forall\, k, q \in \mathcal{J} : \nabla_{\mathbf{w}_{kq}} J_k\left(\widetilde{\mathbf{w}}_k^\infty\right) = 0 \tag{34}$$

which is equivalent to (29). This proves the theorem. ■

IV. DANSE With K-Channel Broadcast Signals

A. DANSE_K *Algorithm*

In the DANSE_K algorithm, each node broadcasts $\min\{K, M_k\}$-component compressed sensor signal observations to the other nodes. This compresses the data to be sent by node k by a factor of $\max\{(M_k/K), 1\}$. We assume that each node k estimates a K-channel desired signal $\mathbf{d}_k = [d_k(1) \ldots d_k(K)]^T$. Assuming that the desired signals $\mathbf{d}_k$ share a common Q-dimensional latent signal subspace, we will show in Section IV-B that DANSE_K achieves the optimal estimators if K is chosen equal to Q. Notice that the actual signal(s) of interest can be a subset of the vector $\mathbf{d}_k$, and the other entries should then be seen as auxiliary channels to fully capture the latent signal subspace, as explained in Section II-B. Generally, these auxiliary channels are obtained by choosing $K-1$ extra reference sensors at node k.

Again, we use a linear estimator $\mathbf{W}_k$ to estimate $\mathbf{d}_k$ as $\overline{\mathbf{d}}_k = \mathbf{W}_k^H\mathbf{y} = [\mathbf{w}_k(1) \ldots \mathbf{w}_k(K)]^H\mathbf{y}$. The objective for node k is to find the linear MMSE estimator

$$\hat{\mathbf{W}}_k = \arg\min_{\mathbf{W_k}} E\left\{\left\|\mathbf{d}_k - \mathbf{W}_k^H\mathbf{y}\right\|^2\right\}. \tag{35}$$

The solution of (35) is

$$\hat{\mathbf{W}}_k = \mathbf{R}_{yy}^{-1}\mathbf{R}_{yd_k} \tag{36}$$

with $\mathbf{R}_{yd_k} = E\{\mathbf{y}\mathbf{d}_k^H\}$. Again, we define a partitioning of the estimator $\mathbf{W}_k$ as $\mathbf{W}_k = [\mathbf{W}_{k1}^T \ldots \mathbf{W}_{kJ}^T]^T$ with $\mathbf{W}_{kq}$ denoting the $M_k \times K$ submatrix of $\mathbf{W}_k$ that is applied to $\mathbf{y}_q$. We wish to obtain (36) without the need for each node to broadcast all M_k components of the $\mathbf{y}_k$ observations. Instead each node k will broadcast observations of the K-channel compressed signal $\mathbf{z}_k = \mathbf{W}_{kk}^H\mathbf{y}_k$. Since the K channels of $\mathbf{z}_k$ will be highly correlated, further joint compression is possible, but we will not take this into consideration throughout this paper.

A node k can transform the observations of $\mathbf{z}_q$ that it receives from node q by a $K \times K$ transformation matrix $\mathbf{G}_{kq}$. Again, it is noted that $\mathbf{G}_{kq}$ does not decompress the observations of the signal $\mathbf{z}_q$, but makes K new linear combinations of their components. The parametrization of the $\mathbf{W}_k$ effectively applied at node k is then

$$\widetilde{\mathbf{W}}_k = \begin{bmatrix} \mathbf{W}_{11}\mathbf{G}_{k1} \\ \vdots \\ \mathbf{W}_{JJ}\mathbf{G}_{kJ} \end{bmatrix} \tag{37}$$

which is a generalization of (6). Here, node k can only optimize the parameters $\mathbf{W}_{kk}$ and $\mathbf{G}_k = [\mathbf{G}_{k1}^T \ldots \mathbf{G}_{kJ}^T]^T$. We set $\mathbf{G}_{kk} = \mathbf{I}_K$ with $\mathbf{I}_K$ denoting the $K \times K$ identity matrix.

The (KJ)-channel signal $\mathbf{z} = [\mathbf{z}_1^T \ldots \mathbf{z}_J^T]^T$ is a stacked version of all the broadcast signals. Similarly to the notation in Section III, we define the signal $\mathbf{z}_{-k}$ as the signal $\mathbf{z}$ with $\mathbf{z}_k$ omitted, and we define $\mathbf{G}_{k-q}$ as the matrix $\mathbf{G}_k$ with the submatrix $\mathbf{G}_{kq}$ omitted. The MMSE problem that is solved at node k, at iteration i, is now

$$\left[\frac{\mathbf{W}_{kk}^{i+1}}{\mathbf{G}_{k-k}^{i+1}}\right] = \underset{\mathbf{W}_{kk},\mathbf{G}_{k-k}}{\arg\min}\ E\left\{\left\|\mathbf{d}_k - \left[\mathbf{W}_{kk}^H | \mathbf{G}_{k-k}^H\right]\left[\frac{\mathbf{y}_k}{\mathbf{z}_{-k}^i}\right]\right\|^2\right\}. \tag{38}$$

The solution of (38) is

$$\left[\frac{\mathbf{W}_{kk}^{i+1}}{\mathbf{G}_{k-k}^{i+1}}\right] = \left(\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i\right)^{-1}\mathbf{R}_{\tilde{y}_k d_k}^i \tag{39}$$

with $\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i$ defined as in (10) and with

$$\mathbf{R}_{\tilde{y}_k d_k}^i = E\left\{\tilde{\mathbf{y}}_k^i \mathbf{d}_k^H\right\}. \tag{40}$$

The DANSE_K algorithm consists of the following steps:

1) Initialize: $i \leftarrow 0$, $u \leftarrow 1$.
Initialize $\mathbf{W}_{kk}^0$ and $\mathbf{G}_{k-k}^0$ with random matrices, $\forall k \in \mathcal{J}$.
2) Each node $k \in \mathcal{J}$ performs the following operation cycle:
   - Collect the sensor observations $\mathbf{y}_k[iB+n]$, $n = 0 \ldots B-1$.
   - Compress these M_k-dimensional observations to K-dimensional vectors

$$\mathbf{z}_k^i[iB+n] = \mathbf{W}_{kk}^{iH}\mathbf{y}_k[iB+n], \quad n = 0\ldots B-1. \tag{41}$$

   - Broadcast the compressed observations $\mathbf{z}_k^i[iB+n]$, $n = 0 \ldots B-1$, to the other nodes.
   - Collect the $K(J-1)$-dimensional data vectors $\mathbf{z}_{-k}^i[iB+n]$, $n = 0 \ldots B-1$, which are stacked versions of the compressed observations received from the other nodes.
   - Update the estimates of $\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i$ and $\mathbf{R}_{\tilde{y}_k d_k}^i$, by including the newly collected data.
   - Update the node-specific parameters:

$$\left[\frac{\mathbf{W}_{kk}^{i+1}}{\mathbf{G}_{k-k}^{i+1}}\right] = \begin{cases} \left(\mathbf{R}_{\tilde{y}_k\tilde{y}_k}^i\right)^{-1}\mathbf{R}_{\tilde{y}_k d_k}^i & \text{if } k = u \\ \left[\frac{\mathbf{W}_{kk}^{i}}{\mathbf{G}_{k-k}^{i}}\right] & \text{if } k \neq u. \end{cases} \tag{42}$$

   - Compute the estimate of $\mathbf{d}_k[iB+n]$, $n = 0 \ldots B-1$, as

$$\overline{\mathbf{d}}_k[iB+n] = \mathbf{W}_{kk}^{i+1H}\mathbf{y}_k[iB+n] + \mathbf{G}_{k-k}^{i+1H}\mathbf{z}_{-k}^i[iB+n]. \tag{43}$$

3) $i \leftarrow i+1$.
4) $u \leftarrow (u \bmod J) + 1$.
5) Return to step 2)

DANSE_K is a straightforward generalization of the DANSE_1 algorithm as explained in Section III-A, where all vector-variables are replaced by their matrix equivalent. Similarly, expressions (16)–(19) can be straightforwardly generalized to their matrix equivalent.

B. Convergence and Optimality of DANSE_K if $Q = K$ and $\mathbf{A}_k$ Full Rank

We now assume that $\mathbf{d}_k = \mathbf{A}_k\mathbf{d}$, $\forall k \in \mathcal{J}$, with $\mathbf{A}_k$ a $K \times K$ matrix of rank K and $\mathbf{d}$ a complex valued K-channel signal. This means that all desired signals $\mathbf{d}_k$ share the same K-dimensional latent signal subspace (i.e., $Q = K$). Formula (36) shows that in this case all $\hat{\mathbf{W}}_k$ have the same column space, i.e.

$$\forall\, k, q \in \mathcal{J} : \hat{\mathbf{W}}_k = \hat{\mathbf{W}}_q \mathbf{A}_{kq} \tag{44}$$

with $\mathbf{A}_{kq} = \mathbf{A}_q^{-H}\mathbf{A}_k^H$. Therefore, the set $(\hat{\mathbf{W}}_1, \ldots, \hat{\mathbf{W}}_J)$ belongs to the solution space used by DANSE_K, as specified by (37), i.e., $(\hat{\mathbf{W}}_1, \ldots, \hat{\mathbf{W}}_J) \in (\widetilde{\mathbf{W}}_1, \ldots, \widetilde{\mathbf{W}}_J)$. The following theorem generalizes Theorem III.1.

Theorem IV.1: If the sensor signal correlation matrix $\mathbf{R}_{yy}$ has full rank, and if $\mathbf{d}_k = \mathbf{A}_k\mathbf{d}$, $\forall k \in \mathcal{J}$, with $\mathbf{d}$ a complex valued K-channel signal and $\mathbf{A}_k$ a $K \times K$ matrix of rank K, then the DANSE_K algorithm converges for any initialization of its parameters to the MMSE solution (36) for all $k \in \mathcal{J}$.

Proof: The proof of Theorem III.1 can straightforwardly be generalized to prove Theorem IV.1, by replacing every $\mathbf{w}$ and $\mathbf{g}$ by its matrix version $\mathbf{W}$ and $\mathbf{G}$. ■

In practice, the matrices $\mathbf{A}_k$ should be well-conditioned to obtain the optimal estimators, which is reflected in Theorem IV.1 by the condition that $\mathbf{A}_k$ has full rank. If the K-channel desired signal $\mathbf{d}_k$ is defined as the target signal in K reference sensors at node k, this matrix can be ill-conditioned if the reference sensors are close to each other. This problem is investigated in [9], where the DANSE algorithm is used for noise reduction in acoustic sensor networks, and a solution is proposed to tackle this problem.

C. DANSE Under Rank Deficiency

Until now, we have avoided the case where $\mathbf{R}_{yy}$ does not have full rank or when the parameter K is overestimated, i.e., $K > Q$. Both cases can result in broadcast data for which the correlation matrix is rank deficient.[5] In this case, (38) becomes ill-posed since singular correlation matrices are involved. The DANSE_K algorithm can cope with these situations by adding

[5]In the case where $K > Q$, (44) has multiple solutions for $\mathbf{A}_{kq}$ since $\text{rank}(\hat{\mathbf{W}}_k) = Q, \forall k \in \mathcal{J}$. Therefore, the correlation matrix of the broadcast signal $\mathbf{z}_k^i$ becomes singular, once the $M_k \times K$ submatrix $\mathbf{W}_{kk}^i$ reaches this rank deficiency.

a minimum-norm constraint to the local MMSE problems (38), i.e., using the pseudo-inverse instead of a matrix inverse in the computation of the solution of (38) [15]. Extensive simulations have shown that with this modification, the DANSE_K algorithm still converges to an MMSE solution for rank deficient estimation problems (see Section VI).

However, if the matrix $\mathbf{R}_{yy}$ does not have full rank, the solution of (1) is not unique. Simulations have shown that the solutions $\widetilde{\mathbf{W}}_k^{\infty}$ obtained by the DANSE_K algorithm, although leading to a minimal MSE cost at node k, are generally different from the solutions provided by the centralized minimum norm version, i.e.

$$\hat{\mathbf{W}}_k = \mathbf{R}_{yy}^{\dagger}\mathbf{R}_{yd_k} \tag{45}$$

where superscript † denotes the pseudoinverse.

V. DANSE_K Implementation Aspects

A. Estimation of the Signal Statistics

In the theoretical analysis of the DANSE_K algorithm, it is assumed that the second order signal statistics, which are needed to solve the MMSE problem (38) are perfectly known. However, in a practical application, the correlation matrices $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}$ and $\mathbf{R}^i_{\tilde{y}_k d_k}$ have to be estimated, based on the collected signal observations. In this section, we will describe some strategies to estimate these quantities.

Estimation of signal correlation matrices is typically done by time averaging. This means that some assumptions are made on short-term ergodicity and stationarity of the signals involved. However, this stationarity assumption is not necessarily strict. Even when the signals involved are nonstationary (such as in speech processing), the DANSE_K algorithm can provide good estimators. By using long-term correlation matrices, the influence of rapidly changing temporal statistics is smoothed out, yielding estimators that mainly exploit the spatial coherence between the sensors. Since spatial coherence typically changes slowly, the DANSE_K algorithm is able to provide good estimators, even when the signals themselves are highly nonstationary (this is e.g., demonstrated by the multichannel speech enhancement experiments in [9]).

We let $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]$ denote the estimate of $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}$ at time t. Signal correlation matrices are often estimated in practice by means of a forgetting factor $0 < \lambda < 1$, i.e.

$$\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t] = \lambda\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1] + (1-\lambda)\tilde{\mathbf{y}}^i_k[t]\tilde{\mathbf{y}}^i_k[t]^H. \tag{46}$$

Notice that in the DANSE_K algorithm, the statistics change every time a node updates its parameters. Therefore, (46) is not suited to compute $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]$ and $\mathbf{R}^i_{\tilde{y}_k d_k}[t]$, since it uses an infinite time window. A better alternative is a simple time averaging in a finite observation window, i.e.

$$\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t] = \frac{1}{L}\sum_{n=t-L+1}^{t}\tilde{\mathbf{y}}^i_k[n]\tilde{\mathbf{y}}^i_k[n]^H \tag{47}$$

where L is the length of the observation window. The procedure (46) puts more emphasis on the most recent samples, whereas (47) applies an equal weight to all past samples in the observation window. The procedure (47) can be implemented recursively by means of an updating and a downdating term, i.e.

$$\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t] = \mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1] + \frac{1}{L}\tilde{\mathbf{y}}^i_k[t]\tilde{\mathbf{y}}^i_k[t]^H - \frac{1}{L}\tilde{\mathbf{y}}^i_k[t-L+1]\tilde{\mathbf{y}}^i_k[t-L+1]^H. \tag{48}$$

Notice that the window length L introduces a trade-off between tracking performance and estimation performance. Indeed, to have a fast tracking, the statistics must be estimated from short signal segments, yielding larger estimation errors in the correlation matrices that are used to compute the estimators at the different nodes. However, as will be demonstrated in Section VI-B, the DANSE_K algorithm is more robust to these errors, compared to the equivalent centralized algorithm, due to the fact that DANSE_K uses correlation matrices with smaller dimensions than the network-wide estimation problem.

The estimation of $\mathbf{R}^i_{\tilde{y}_k d_k}$ is less straightforward since the signal $\mathbf{d}_k$ cannot be observed directly. However, depending on the application and the signals involved, some strategies can be developed to estimate $\mathbf{R}^i_{\tilde{y}_k d_k}$, as explained in the following two examples.

If the transmitting sources are controlled by the application itself, as it is the case in a communications scheme, the source signals that define the different channels in $\mathbf{d}$ can be manipulated directly. At periodic intervals, a deterministic training sequence can be broadcast by the transmitters. If the nodes have knowledge about these training sequences, they can use this to compute $\mathbf{R}^i_{\tilde{y}_k d_k}[t]$ in a similar way as in (48), during the broadcast of these training sequences. After the broadcast, the estimate is fixed until new training sequences are broadcast.

A different strategy can be applied if the desired signal $\mathbf{d}_k$ has an ON–OFF behavior.[6] Assume that the sensor signals in $\mathbf{y}$ consist of a desired component $\mathbf{x}$ and an additive noise component $\mathbf{n}$, i.e., $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{x}$ has an ON–OFF behavior, and where then $\mathbf{d}_k = [x_{k1} \dots x_{kK}]^T$. In many practical applications, it can also be assumed that $\mathbf{x}$ and $\mathbf{n}$ are independent, and therefore[7]

$$E\{\mathbf{x}\mathbf{x}^H\} = E\{\mathbf{y}\mathbf{y}^H\} - E\{\mathbf{n}\mathbf{n}^H\}. \tag{49}$$

If there is a detection mechanism available that detects whether the signal $\mathbf{x}$ is present or not, one can estimate $E\{\mathbf{n}\mathbf{n}^H\}$ in time segments where only noise is observed ("noise-only segments"). Since the noise is uncorrelated to the desired component $\mathbf{d}_k$, we find that

$$\mathbf{R}^i_{\tilde{y}_k d_k} = E\left\{\tilde{\mathbf{x}}^i_k\tilde{\mathbf{x}}^{iH}_k\right\}\mathbf{E}_K \tag{50}$$

with

$$\mathbf{E}_K = \begin{bmatrix}\mathbf{I}_K \\ \mathbf{O}_{(M_k+(J-2)K)\times K}\end{bmatrix} \tag{51}$$

where $\tilde{\mathbf{x}}^i_k$ is the desired component in the signal $\tilde{\mathbf{y}}^i_k$. The selection matrix $\mathbf{E}_K$ is used to select the first K columns corre-

[6]This is often used in speech enhancement applications, since a speech signal typically contains a lot of silent pauses in between words or sentences.

[7]For the sake of an easy exposition, we assume that the signals $\mathbf{x}$ and $\mathbf{n}$ have zero mean.

sponding to $\mathbf{d}_k = [x_{k1} \ldots x_{kK}]^T$. Define the noise correlation matrix

$$\mathbf{R}^i_{\tilde{n}_k\tilde{n}_k} = E\left\{\tilde{\mathbf{n}}^i_k \tilde{\mathbf{n}}^{iH}_k\right\} \tag{52}$$

where $\tilde{\mathbf{n}}^i_k$ denotes the noise component in the signal $\tilde{\mathbf{y}}^i_k$. With (50), and similarly to (49), we readily find that

$$\mathbf{R}^i_{\tilde{y}_k d_k} = \left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k} - \mathbf{R}^i_{\tilde{n}_k\tilde{n}_k}\right)\mathbf{E}_K. \tag{53}$$

Using (53), one can compute $\mathbf{R}^i_{\tilde{y}_k d_k}[t]$ as the difference between $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]$ and $\mathbf{R}^i_{\tilde{n}_k\tilde{n}_k}[t]$, where the latter is computed as in (48), during noise-only periods.

Notice that, even if the target signal does not have this ON–OFF behavior, the above strategy can be used in a semi-adaptive context, i.e., where the target signal statistics may change but the noise statistics are static and *a priori* known (or vice versa). Indeed, if $E\{\mathbf{n}\mathbf{n}^H\}$ is known, then (53) can be used to compute the required statistics. Notice that $\mathbf{R}^i_{\tilde{n}_k\tilde{n}_k}$ in (53) is a compressed version of $E\{\mathbf{n}\mathbf{n}^H\}$, i.e., it depends on the current parameters in $\mathbf{W}^i$. Therefore, each node k has to broadcast the entries of $\mathbf{W}^i_{kk}$, which are needed in the other nodes to compress the corresponding submatrices in $E\{\mathbf{n}\mathbf{n}^H\}$. Since these values change only once for each JB observations that are collected by the sensors, the resulting increase in bandwidth is negligible compared to the transmission of the samples of $\mathbf{z}^i_k$.

B. Computational Complexity

The estimation of the correlation matrices $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}$ and $\mathbf{R}^i_{\tilde{y}_k d_k}$, and the inversion of the former, are the most computationally expensive steps of the DANSE$_K$ algorithm. From (48) it follows that an update of $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]$ at node k, has a computational complexity of

$$O\left((M_k + K(J-1))^2\right) \tag{54}$$

i.e., it is quadratic in the number of nodes J, the number of channels K in the broadcast signals, and the number of channels M_k of the signal $\mathbf{y}_k$. If node k updates its parameters $\mathbf{W}^i_{kk}$ and $\mathbf{G}^i_{k-k}$ according to (39), it performs a matrix inversion, which is computationally more expensive than (54). However, instead of computing this inversion, node k can directly update the inverse of $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]$ at each time t by means of the matrix inversion lemma [15], i.e.

$$\begin{aligned}\left(\mathbf{R}^{i\,\mathrm{temp}}_{\tilde{y}_k\tilde{y}_k}\right)^{-1} &= \left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1]\right)^{-1} \\ &\quad - \frac{\left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1]\right)^{-1}\tilde{\mathbf{y}}^i_k[t]\tilde{\mathbf{y}}^i_k[t]^H\left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1]\right)^{-1}}{1+\tilde{\mathbf{y}}^i_k[t]^H\left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t-1]\right)^{-1}\tilde{\mathbf{y}}^i_k[t]}\end{aligned} \tag{55}$$

$$\begin{aligned}\left(\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}[t]\right)^{-1} &= \left(\mathbf{R}^{i\,\mathrm{temp}}_{\tilde{y}_k\tilde{y}_k}\right)^{-1} \\ &\quad + \frac{\left(\mathbf{R}^{i\,\mathrm{temp}}_{\tilde{y}_k\tilde{y}_k}\right)^{-1}\tilde{\mathbf{y}}^i_k[t-L+1]\tilde{\mathbf{y}}^i_k[t-L+1]^H\left(\mathbf{R}^{i\,\mathrm{temp}}_{\tilde{y}_k\tilde{y}_k}\right)^{-1}}{1-\tilde{\mathbf{y}}^i_k[t-L+1]^H\left(\mathbf{R}^{i\,\mathrm{temp}}_{\tilde{y}_k\tilde{y}_k}\right)^{-1}\tilde{\mathbf{y}}^i_k[t-L+1]}.\end{aligned} \tag{56}$$

This update also has computational complexity (54), and therefore this is the overall complexity for a single node in the DANSE$_K$ algorithm.

VI. Numerical Simulations

In this section, we provide simulation results to demonstrate the behavior of the DANSE$_K$ algorithm. In Section VI-A, we perform batch mode simulations where the required statistics are computed over the full length signals, and where the $\mathbf{d}_k$'s are available[8] to compute $\mathbf{R}^i_{\tilde{y}_k d_k}$. In the batch version of DANSE$_K$, all iterations are performed on the same set of signal observations. In Section VI-B, a more practical scenario with moving sources is considered. The DANSE$_K$ algorithm adapts to the changes in the scenario, and each set of observations is only broadcast once, i.e., subsequent iterations are performed over different observation sets. Furthermore, a practical estimation of the correlation matrices is used, where the $\mathbf{d}_k$'s are assumed to be unavailable.

A. Batch Mode Simulations

In this section, we simulate the DANSE$_K$ algorithm in batch mode. This means that all iterations are performed on the full signal length. The network consists of four nodes ($J = 4$), each having 10 sensors ($M = 40$). The dimension of the latent signal subspace defined by $\mathbf{d}$ is $Q = 3$. All 3 channels of $\mathbf{d}$ are uniformly distributed random processes on the interval $[-0.5, 0.5]$ from which $N = 10\,000$ samples are generated. The coefficients in $\mathbf{A}_k$ are generated by a uniform random process on the unit interval. The sensor signals in $\mathbf{y}$ consist of the different random mixtures of the latent Q-channel signal $\mathbf{d}$ to which zero-mean white noise is added with half the power of the channels of $\mathbf{d}$. The initial values of all $\mathbf{W}_{kk}$ and $\mathbf{G}_k$ are taken from a uniform random distribution on the unit interval.

The batch mode performance of the DANSE$_1$ algorithm as well as the DANSE$_3$ algorithm is simulated for this particular scenario. All evaluations of the MSE cost functions J_k are performed on the equivalent least-squares (LS) cost functions, i.e.

$$\sum_{t=0}^{N}\left\|\mathbf{d}_k[t] - \mathbf{W}^H_k\mathbf{y}[t]\right\|^2. \tag{57}$$

Also, the correlation matrices are replaced by their least squares equivalent, i.e., $E\{\mathbf{y}\mathbf{y}^H\}$ is replaced by $\mathbf{Y}\mathbf{Y}^H$ where $\mathbf{Y}$ denotes the $M \times N$ sample matrix that contains samples of the variable $\mathbf{y}$ in its columns.

The results are illustrated in Fig. 4, showing the LS cost of node 1 versus the iteration index i. Node 1 is the first node that performs an update. It is observed that the DANSE$_3$ algorithm converges to the optimal linear LS solution, whereas the DANSE$_1$ algorithm does not since $K < Q$ in this case. Downsampling the curve corresponding to DANSE$_3$ by a factor $J = 4$, keeping only the iterations in which node 1 updates its parameters, results in a monotonically decreasing cost. This is because of expression (22), showing that the cost indeed monotonically decreases whenever a node optimizes its $\mathbf{G}$ parameters. If the curve corresponding to DANSE$_1$ is downsampled

[8]This is similar to using *a priori* known training sequences.

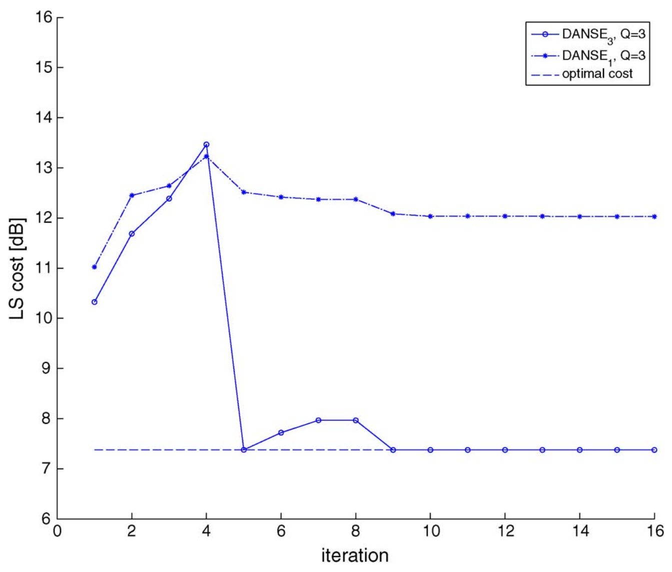


Fig. 4. LS error of node 1 versus iteration i for four different scenarios in a network with $J = 4$ nodes. Each node has 10 sensors.

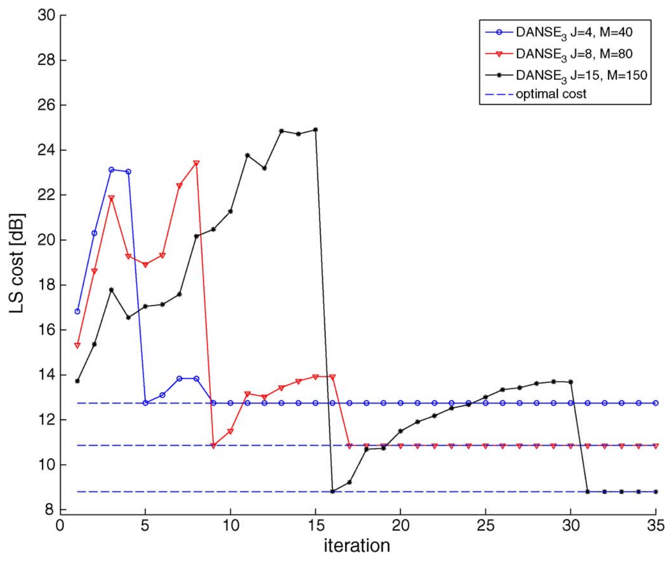


Fig. 5. LS error of node 1 versus iteration i for networks with $J = 4$, $J = 8$ and $J = 15$ nodes respectively. Each node has 10 sensors.

with the same factor, we do not obtain a monotonically decreasing cost, since expression (22) is not valid anymore for this case.

In Fig. 5, we vary the number of nodes J, keeping all other parameters unchanged. All nodes again have 10 sensors. Not surprisingly, the convergence time of DANSE_3 increases linearly with J since the effective number of updates per time unit in node 1 is reduced. As soon as each node has updated its parameters three times, the cost is almost at its minimum at each node.

In Fig. 6(a), we increase the value of K while keeping $Q = 3$. Notice that this corresponds to the case where K is overestimated and hence communication bandwidth is used inefficiently. The estimation problem becomes rank deficient in this case, and so the algorithm should be modified by replacing matrix inversions by pseudoinversions (see Section IV-C). The algorithm still converges, and the optimal LS cost is again reached after three iterations per node when K is overestimated. In Fig. 6(b), we increase the value of K together with Q, keeping $Q = K$. This is again observed to have a negligible effect on convergence time.

As a general conclusion, we can state that for all settings of the parameters K, Q, J, the DANSE_K algorithm approximately achieves convergence as soon as each node has updated its parameters three times.

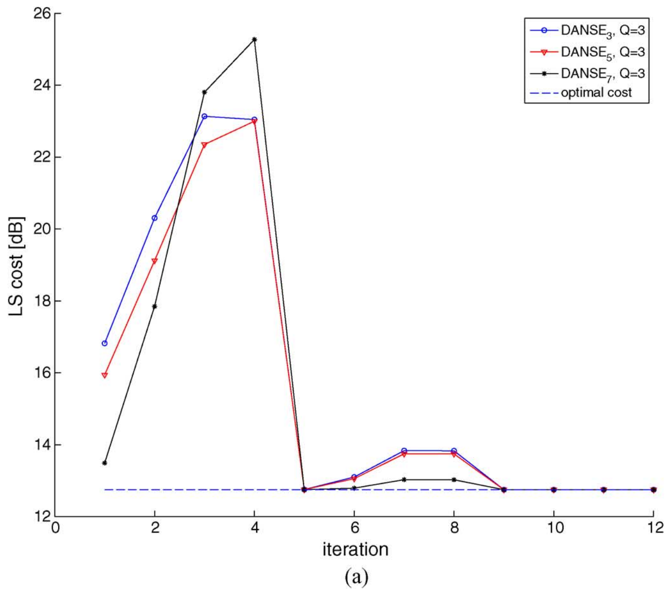
Simulation results with speech signals are provided in a follow-up paper [9]. In this paper, a distributed speech enhancement algorithm based on DANSE_K and its variations, is tested in a simulated acoustic sensor network scenario.

B. Adaptive Implementation

In this section, we show simulation results of a practical implementation of the DANSE_K algorithm in a scenario with moving sources. The main difference with the batch mode simulations is that subsequent iterations are now performed on different signal segments, i.e., the same data is never used twice. This yields larger estimation errors, since shorter signal segments are used to estimate the statistics of the input signals. Furthermore, we will use a practical estimation procedure to estimate the correlation matrices $\mathbf{R}^i_{\tilde{y}_k\tilde{y}_k}$ and $\mathbf{R}^i_{\tilde{y}_k d_k}$, yielding larger estimation errors.

The scenario is depicted in Fig. 7. The network contains $J = 4$ nodes ($\Diamond$). Each node has a reference sensor at the node itself, and can collect observations of five additional sensors ($\circ$) that are uniformly distributed within a 1.6-m radius around the node. Eight localized white Gaussian noise sources (∇) are present. Two target sources ($\square$) move back and forth over the indicated straight lines at a speed of 1 m/s, and halt for 2 s at the end points of these lines. The first source (moving on the vertical line) transmits a low-pass filtered white noise signal with a cut-off frequency of 1600 Hz. The other source transmits a band-pass filtered white noise signal in the frequency range from 1600 to 3200 Hz. Both target sources have an ON–OFF behavior with a period of 0.2 s and both are active 66% of the time. It is assumed that at each time t, all nodes can detect whether the sources are active or not. The time between two consecutive updates is 0.4 s, which corresponds to two ON–OFF cycles of the target sources. This means that, every 0.4 s, the iteration index i changes to $i+1$. The sensors observe their signals at a sampling frequency of $f_s = 16$ kHz.

The target source signals have half the power of the noise sources. In addition to the spatially correlated noise, independent white Gaussian sensor noise is added to each sensor signal. This noise component is 10% of the power of the localized noise signals. The individual signals originating from the target sources and the noise sources that are collected by a specific sensor are attenuated in power and summed. The attenuation factor of the signal power is $1/r$, where r denotes the distance between the source and the sensor. We assume that there is no time delay in the transmission path between the sources and the



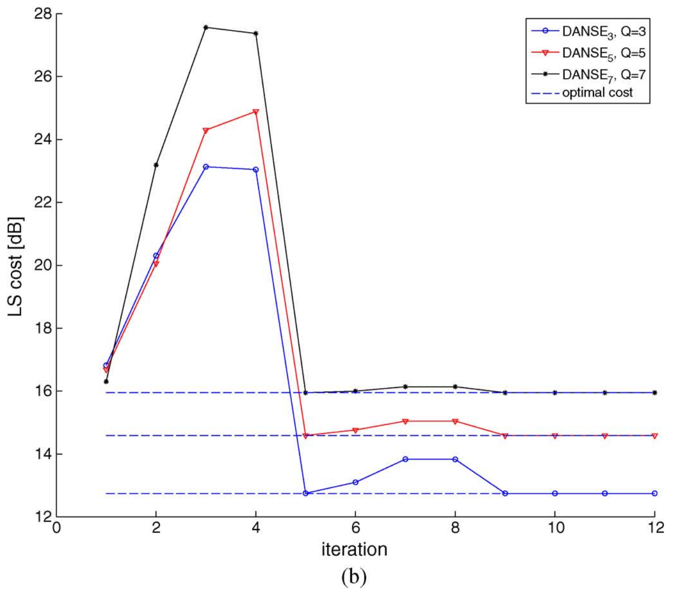


Fig. 6. LS error of node 1 versus iteration i in a network with $J = 4$ nodes. Each node has 10 sensors. (a) Different values of K, keeping $Q = 3$ and (b) different values of $K = Q$.

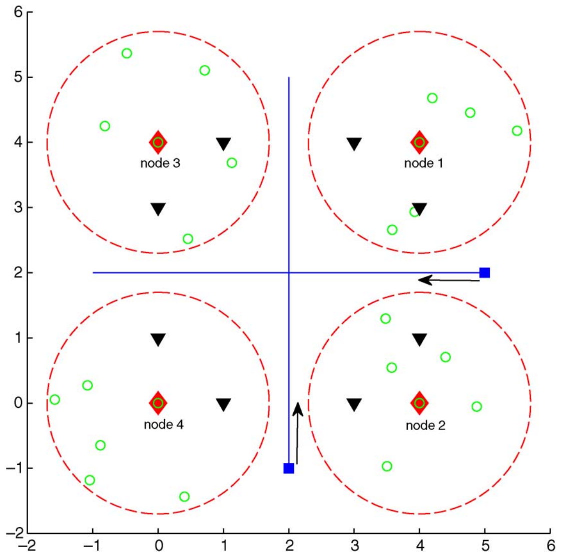


Fig. 7. Description of the simulated scenario. The network contains four nodes ($\diamondsuit$), each node collecting observations in a cluster of six sensors ($\circ$). One sensor of each cluster is positioned at the node itself. Two target sources ($\square$) are moving over the indicated straight lines. Eight noise sources are present (∇).

sensors.[9] Each node collects six sensor signal observations, and uses five differently delayed versions of each of these signals in its estimation process to exploit the temporal correlation in the target source signals. This means that $M_k = 30$.

We let y_{k1} denote the signal that is collected at the reference sensor of node k. It consists of an unknown mixture d_{k1} of the two target source signals, and a noise component n_{k1}, i.e.

$$y_{k1} = d_{k1} + n_{k1} = \mathbf{a}_{k1}^T \mathbf{d} + n_{k1} \tag{58}$$

[9]Since the time delays are the same for all sensors, the spatial information is purely energy based in this case. Therefore, the nodes cannot perform any beamforming towards specific locations by exploiting different delay paths between sources and sensors.

where $\mathbf{d}$ is the two-channel signal containing the two target source signals, and where $\mathbf{a}_{k1}$ denotes an unknown mixture vector. The goal for node k is to estimate the signal d_{k1}, i.e., the target source component in its reference sensor. Since $Q = 2$, the DANSE_2 algorithm is used, and therefore an auxiliary desired channel is used to obtain a two-channel desired signal $\mathbf{d}_k$ at every sensor. The auxiliary channel of $\mathbf{d}_k$ consists of the target source component d_{k2} in the signal y_{k2} that is collected by another sensor of node k. This component consists of another unknown mixture of the target sources, so that the conditions of Theorem IV.1 are satisfied.

The correlation matrix $\mathbf{R}^i_{\tilde{y}_k d_k}[t]$ is computed according to (53). The estimates $\mathbf{R}^i_{\tilde{y}_k \tilde{y}_k}[t]$ and $\mathbf{R}^i_{\tilde{n}_k \tilde{n}_k}[t]$ are computed similarly to (48) with a window length of $L_1 = 4200$ and $L_2 = 2200$, respectively, which matches the time between two consecutive updates.

We will use the signal-to-error ratio (SER) as a measure to assess the performance of the estimators. The instantaneous SER for node k at time t and iteration i is computed over 3200 samples, and is defined as

$$\mathrm{SER}^i_k[t] = \frac{\sum_{n=t+1}^{t+3200} |d_{k1}[n]|^2}{\sum_{n=t+1}^{t+3200} \left|d_{k1}[n] - \tilde{\mathbf{w}}^i_k(1)^H \mathbf{y}[n]\right|^2} \tag{59}$$

where $\tilde{\mathbf{w}}^i_k(1)$ denotes the first column of the estimator $\widetilde{\mathbf{W}}^i_k$, as defined in (37). Notice that this is the estimator that is of actual interest, since it estimates the desired component d_{k1} in the reference sensor. The other column of $\widetilde{\mathbf{W}}^i_k$ is viewed as an auxiliary estimator that is used for the generation of the second channel of the broadcast signal $\mathbf{z}^i_k$.

Fig. 8 shows the SER of the four nodes at different time instants. Dashed vertical lines are plotted to indicate the points in time where both sources start moving, and full vertical lines indicate when they stop moving. The sources stand still in the time intervals [0–4] s, [10–12] s, and [18–20] s. The performance is

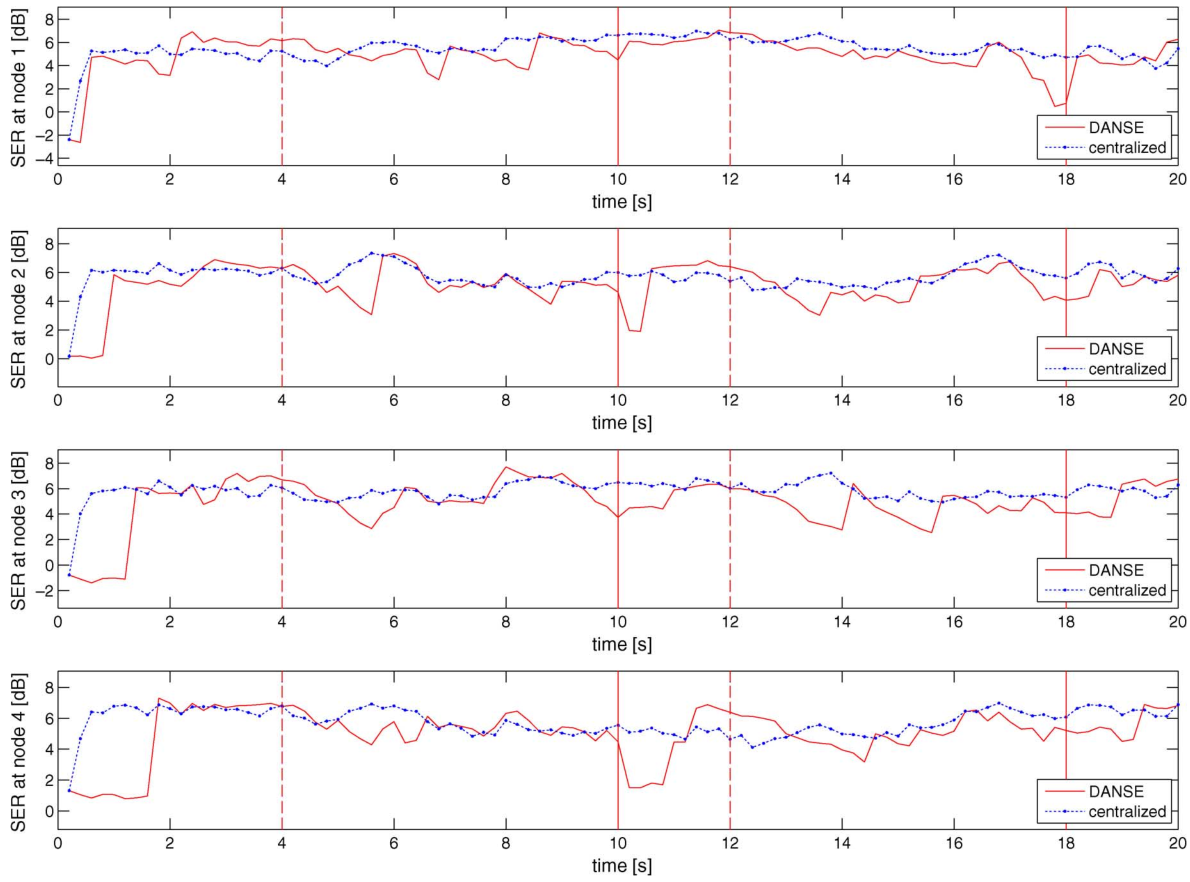


Fig. 8. SER versus time at the four nodes depicted in Fig. 7. The centralized version is added as a reference. Window lengths are $L_1 = 4200$ and $L_2 = 2200$.

compared to the centralized version, in which all sensor signals are centralized in a single fusion center that computes the optimal estimators according to (2).

In the first 4 s, both sources stand still. The DANSE_2 algorithm needs some time to reach a good estimator at each node (about 2 s), whereas the centralized algorithm converges much faster. This is because the DANSE_2 algorithm updates its nodes one at a time, with 0.4 s in between two consecutive updates. The centralized algorithm on the other hand, can update its estimators every time a new sample is collected. After a number of iterations however, the DANSE_2 algorithm converges to the optimal estimators.

Not surprisingly, it is observed that the centralized algorithm has better tracking capabilities than the DANSE_2 algorithm. This is again a consequence of the fact that the centralized version computes a new estimator each time a new sample is collected, yielding a much faster convergence. However, the DANSE_2 algorithm is able to react to changes in the scenario and always regains optimality after a number of iterations.

Notice that, once the DANSE_2 algorithm has converged, it outperforms the centralized algorithm. This can be explained by the fact that the DANSE_2 algorithm uses correlation matrices with smaller dimension compared to the correlation matrices that are used by the centralized algorithm. Small matrices are generally better conditioned and have a smaller estimation error than larger matrices. This performance increase of DANSE_K compared to its centralized version is observed to become more significant when the number of sensors M increases, yielding larger matrices, or when the window length L decreases, yielding larger estimation errors in the correlation matrices. Fig. 9 shows the performance of DANSE_2 and its centralized version, now with window lengths $L_1 = 2100$ and $L_2 = 1100$, i.e., roughly half the sizes of the first experiment. It is observed that the estimation performance of the centralized algorithm significantly decreases compared to the first experiment, whereas the DANSE_2 algorithm is less influenced by the short window length. This observation demonstrates that DANSE_K is more robust to estimation errors in the correlation matrices compared to its centralized equivalent. Notice that DANSE_K converges much faster in the second experiment, since the time between two consecutive updates is now 0.2 s instead of 0.4 s, due to the shorter window lengths. As already mentioned in Section V, this faster tracking comes with the drawback that the estimation performance decreases due to larger errors in the estimation of the correlation matrices.

In [14], a modified DANSE_K algorithm is studied, where an improved tracking performance is obtained, by letting nodes update simultaneously.

VII. Conclusion

In this paper, we have introduced a distributed adaptive algorithm (DANSE_K) for linear MMSE estimation of node-specific signals in a fully connected broadcasting sensor network,

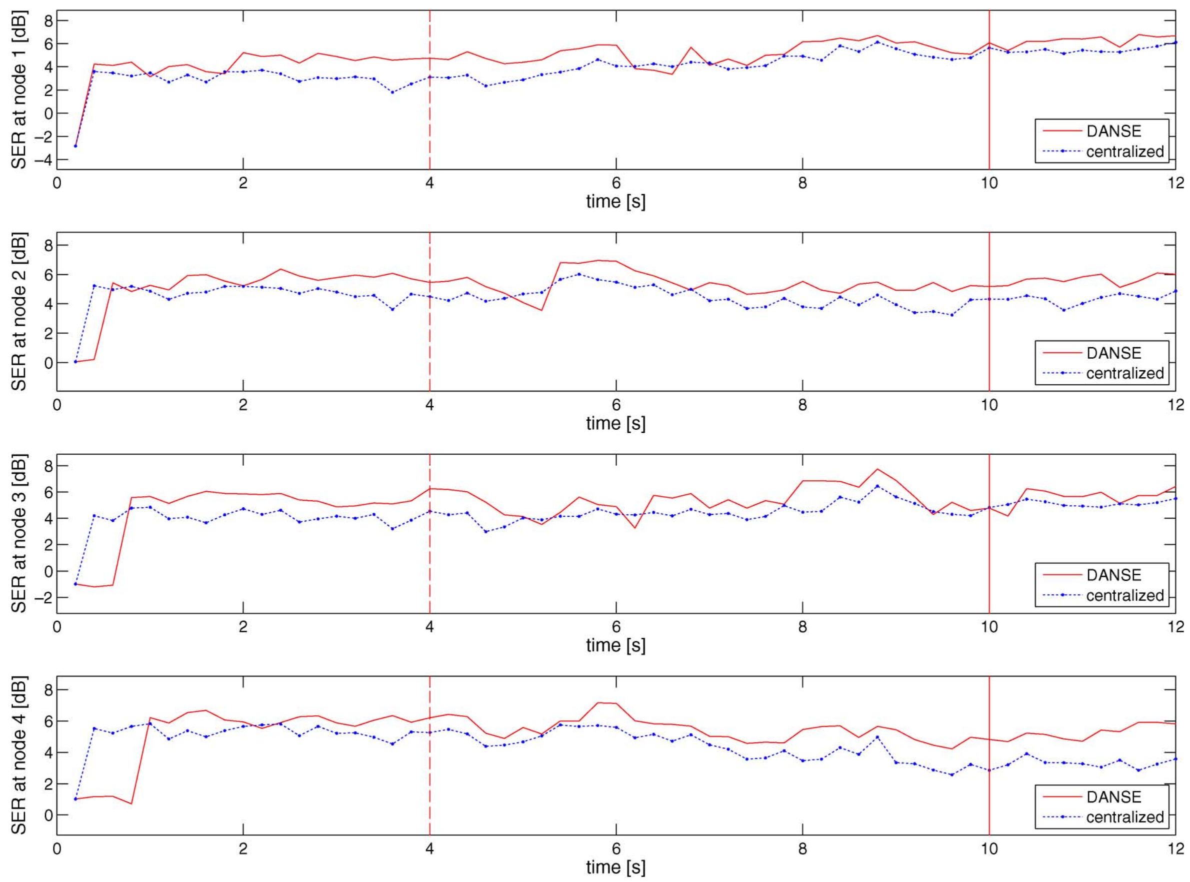


Fig. 9. SER versus time at the four nodes depicted in Fig. 7. The centralized version is added as a reference. Window lengths are $L_1 = 2100$ and $L_2 = 1100$.

where each sensor node collects multichannel sensor signal observations. The algorithm significantly compresses the data to be broadcast, and the computational load is shared amongst the nodes. It is shown that, if the node-specific desired signals share a common low-dimensional latent signal subspace, DANSE_K converges and provides the optimal linear MMSE estimator for every node-specific estimation problem, as if all nodes have access to all the sensor signals in the network. Simulations demonstrate that the algorithm achieves the same performance as a centralized algorithm. A practical adaptive implementation of the algorithm is described and simulated, demonstrating the tracking capabilities of the algorithm in a dynamic scenario. It is observed that the DANSE_K algorithm is more robust to estimation errors in the correlation matrices, compared to its centralized equivalent. In this paper, we have only considered the case where nodes update their parameters in a sequential round robin fashion. A modified DANSE_K algorithm is studied in a companion paper [14], where an improved tracking performance is obtained, by letting nodes update simultaneously.

Acknowledgment

The authors would like to thank B. Cornelis and the anonymous reviewers for their valuable comments after proof-reading this paper.

References

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, “Instrumenting the world with wireless sensor networks,” in *Proc. 2001 IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP ’01)*, 2001, vol. 4, pp. 2033–2036.

[2] C. G. Lopes and A. H. Sayed, “Incremental adaptive strategies over distributed networks,” *IEEE Trans. Signal Processing*, vol. 55, pp. 4064–4077, Aug. 2007.

[3] C. G. Lopes and A. H. Sayed, “Diffusion least-mean squares over adaptive networks: Formulation and performance analysis,” *IEEE Trans. Signal Processing*, vol. 56, pp. 3122–3136, Jul. 2008.

[4] F. Cattivelli, C. G. Lopes, and A. H. Sayed, “Diffusion recursive least-squares for distributed estimation over adaptive networks,” *IEEE Trans. Signal Processing*, vol. 56, pp. 1865–1877, May 2008.

[5] I. Schizas, G. Giannakis, and Z.-Q. Luo, “Distributed estimation using reduced-dimensionality sensor observations,” *IEEE Trans. Signal Processing*, vol. 55, pp. 4284–4299, Aug. 2007.

[6] Z.-Q. Luo, G. Giannakis, and S. Zhang, “Optimal linear decentralized estimation in a bandwidth constrained sensor network,” in *Proc. 2005 Int. Symp. Inf. Theory (ISIT)*, Sept. 2005, pp. 1441–1445.

[7] K. Zhang, X. Li, P. Zhang, and H. Li, “Optimal linear estimation fusion—Part VI: Sensor data compression,” in *Proc. 2003 Sixth Int. Conf. Inf. Fusion*, 2003, vol. 1, pp. 221–228.

[8] Y. Zhu, E. Song, J. Zhou, and Z. You, “Optimal dimensionality reduction of sensor data in multisensor estimation fusion,” *IEEE Trans. Signal Processing*, vol. 53, pp. 1631–1639, May 2005.

[9] A. Bertrand and M. Moonen, “Robust distributed noise reduction in hearing aids with external acoustic sensor nodes,” *EURASIP J. Adv. Signal Process.*, vol. 2009, p. 14, 2009, 10.1155/2009/530435, Article ID 530435.

[10] S. Doclo, T. van den Bogaert, M. Moonen, and J. Wouters, “Reduced-bandwidth and distributed MWF-based noise reduction algorithms for binaural hearing aids,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 17, pp. 38–51, Jan. 2009.

[11] T. Klasen, T. Van den Bogaert, M. Moonen, and J. Wouters, "Binaural noise reduction algorithms for hearing aids that preserve interaural time delay cues," *IEEE Trans. Signal Processing*, vol. 55, pp. 1579–1585, April 2007.

[12] S. Doclo, T. Klasen, T. Van den Bogaert, J. Wouters, and M. Moonen, "Theoretical analysis of binaural cue preservation using multi-channel Wiener filtering and interaural transfer functions," in *Proc. Int. Workshop Acoust. Echo Noise Contr. (IWAENC)*, Paris, France, Sep. 2006.

[13] A. Bertrand and M. Moonen, "Distributed adaptive estimation of correlated node-specific signals in a fully connected sensor network," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 2009, pp. 2053–2056.

[14] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks—Part II: Simultaneous and asynchronous node updating," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5292–5306, Oct. 2010.

[15] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: The Johns Hopkins University Press, 1996.

[16] J. C. Bezdek and R. J. Hathaway, "Some notes on alternating optimization," in *Advances in Soft Computing*. Berlin, Germany: Springer, 2002, pp. 187–195.

[17] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Belmont, MA: Athena Scientific, 1997.

Alexander Bertrand (S'08) was born in Roeselare, Belgium, in 1984. He received the M.Sc. degree in electrical engineering from Katholieke Universiteit Leuven, Belgium, in 2007.

He is currently pursuing the Ph.D. degree with the Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven, and was supported by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). His research interests are in multichannel signal processing, *ad hoc* sensor arrays, wireless sensor networks, distributed signal enhancement, speech enhancement, and distributed estimation.

Marc Moonen (M'94–SM'06–F'07) received the electrical engineering degree and the Ph.D. degree in applied sciences from Katholieke Universiteit Leuven, Belgium, in 1986 and 1990, respectively.

Since 2004, he has been a Full Professor with the Electrical Engineering Department, Katholieke Universiteit Leuven, where he is heads a research team working in the area of numerical algorithms and signal processing for digital communications, wireless communications, DSL, and audio signal processing.

Dr. Moonen received the 1994 KU Leuven Research Council Award, the 1997 Alcatel Bell (Belgium) Award (with P. Vandaele), the 2004 Alcatel Bell (Belgium) Award (with R. Cendrillon), and was a 1997 "Laureate of the Belgium Royal Academy of Science." He received a journal Best Paper award from the IEEE TRANSACTIONS ON SIGNAL PROCESSING (with G. Leus) and from Elsevier *Signal Processing* (with S. Doclo). He was chairman of the IEEE Benelux Signal Processing Chapter (1998–2002), and is currently Past-President of European Association for Signal Processing (EURASIP) and a member of the IEEE Signal Processing Society Technical Committee on Signal Processing for Communications. He served as Editor-in Chief for the EURASIP *Journal on Applied Signal Processing* (2003–2005), and has been a member of the editorial board of *Integration*, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II (2002–2003) and IEEE SIGNAL PROCESSING MAGAZINE (2003–2005) and *Integration, the VLSI Journal*. He is currently a member of the editorial board of EURASIP *Journal on Advances in Signal Processing*, EURASIP *Journal on Wireless Communications and Networking*, and *Signal Processing*.