Signal Processing 107 (2015) 109–122

Contents lists available at ScienceDirect

Signal Processing

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

Distributed estimation of the inverse of the correlation matrix for privacy preserving beamforming

Yuan Zeng*, Richard C. Hendriks

Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands

ARTICLE INFO

Article history:
Received 3 February 2014
Received in revised form
9 July 2014
Accepted 9 July 2014
Available online 24 July 2014

Keywords:
Distributed estimation
Gossip processing
Correlation matrix
Wireless acoustic sensor networks

ABSTRACT

In this paper, we consider a privacy preserving scenario where users in the network want to perform distributed target source estimation with a wireless acoustic sensor network (WASN), without revealing the actual source of interest to other entities in the network. This implies that users do not share the steering vector of the beamformer with any other party. For distributed multi-channel noise reduction in WASNs, distributed estimation of the inverse noise or noise+target correlation matrix is an important aspect and in general a challenging problem.

To make both privacy preservation and distributed multi-channel noise reduction possible, we make use of the fact that recursive estimation of the inverse correlation matrix can be structured as a consensus problem and can be realized in a distributed manner via the randomized gossip algorithm. This makes it possible to compute the MVDR in distributed manner without revealing the steering vector to any of the other entities in the network, and providing privacy about the actual source of interest. We provide theoretical analysis and numerical simulations to investigate the convergence error between the gossip-based estimated correlation matrix and the centralized estimated correlation matrix. It is shown that the convergence error accumulates across time without using a sufficient number of transmissions in the gossip-based algorithm. To eliminate this convergence error, we propose in addition a clique-based algorithm for distributed estimation of the inverse correlation matrix (CbDECM). Theoretical analysis shows that the CbDECM algorithm converges to the centralized estimate of the matrix inverse.

We investigate the performance of the presented clique-based distributed framework in combination with a distributed privacy preserving MVDR beamformer, where information about the actual source of interest is kept private. Simulations show that the proposed algorithm converges to the centralized MVDR beamformer.

1. Introduction

To improve the quality and intelligibility of speech processing applications under noisy environments, it is customary to equip them with a single- or multi-microphone noise reduction algorithm (for an overview see e.g., [1–3]). As multi-microphone noise reduction algorithms can take advantage of spatial diversity, they usually lead to better speech quality and intelligibility than their single-microphone counterparts. In particular it is the number of microphones and their placement that determine the potential performance of a multi-microphone noise reduction algorithm. However, as most mobile speech processing

* Corresponding author.
E-mail addresses: Y.Zeng@tudelft.nl (Y. Zeng), R.C.Hendriks@tudelft.nl (R.C. Hendriks).

http://dx.doi.org/10.1016/j.sigpro.2014.07.006
0165-1684/

devices have relatively small dimensions, the number of microphones as well as their placement is rather restricted.

Using the so-called wireless acoustic sensor networks (WASNs), it is possible to use a much larger number of microphones that are distributed in the environment and where their placement is not restricted by the device itself. This allows a further increase in noise reduction performance. However, the conventional multi-microphone noise reduction algorithms (e.g. [2,3]) are characterized by having one processor where all data is processed centrally. Such centralized algorithms are less suitable for a WASN, as they may require higher energy consumption or transmission bandwidth than necessary. The fact that the sensors in a WASN are all equipped with a (simple) processor potentially owned by different users allows us to perform intermediate processing of data without the need to first send all data to a single point in the network. This has recently led to an increased research interest to distributed speech enhancement algorithms, see e.g., [4–7].

As the processors and sensors in the WASN context are not necessarily anymore owned by a single user, distributed processing might come with serious privacy risks. These could range from an increased risk of being eavesdropped to an increased risk that private data or information becomes public. Within the speech enhancement context, such privacy issues were first addressed in [8,9] for two scenarios. The scenario in [8] considered the case where a user keeps the exact source of interest private for other users, while [9] considered the scenario where eavesdropping by untrusted third parties is overcome. Both contributions employed homomorphic encryption [10] to provide the necessary privacy. However, homomorphic encryption is computationally very complex, and requires very high bit rates for data transmission. In the current paper, we consider a different approach and develop a framework for distributed signal estimation employing a WASN, while providing the user a certain level of privacy with respect to the source of interest. We consider the case where the users of the network do not want to share to which specific source in the environment they are listening, while they do want to make use of the WASN to estimate their signal of interest.

More specifically, the application scenario that we consider in this paper is the one where multiple users make use of a WASN that consists of many processors (including their own) and where each processor is equipped with multiple microphones. The users can use the additional sensors in the WASN to obtain an improved estimate of their signal of interest, which can be different for each user and is usually determined by the steering vector of the beamformer. However, due to privacy reasons, the users want to keep their source of interest private (i.e., the steering vector). Although the microphone signals might be public, hiding the steering vector will overcome that the exact combination of microphone signals required for specific target signal estimation is publicly known. Moreover, hiding the steering vector makes sure that none of the entities with access to the network is able to reveal which conversation or source is apparently of interest for a particular user. This will guarantee a certain amount of privacy to the users of the network. One way to guarantee privacy preservation on the source of interest would be to send all data to all nodes and compute a conventional beamformer in every node. In this way, users do not need to make the steering vector public. However, this requires a lot of data transmission. Performing calculations in a distributed way will reduce the number of data transmissions, due to the fact that local nodes perform intermediate calculations. This leads to a data compression depending on the number microphones per node. We investigate thus the possibility that each user in the WASN estimates his signal of interest by performing distributed computations on the WASN data, while keeping the particular source of interest private. To do so, we concentrate on distributed estimation of one of the most well-known beamformers, the minimum variance distortionless response (MVDR) beamformer.

The MVDR beamformer depends on the inverse of the noise or noise+target spectral correlation matrix. Computing this inverse in a distributed manner is not trivial, as the data in a WASN is not centrally present and each element of the inverse of the correlation matrix is a function of the statistics of the noise or the noise+target at multiple microphones. In [7], a distributed MVDR was presented based on a randomized gossip algorithm [11] where the inversion of the noise correlation matrix was overcome by assuming the correlation matrix to be diagonal. This simplifies distributed computation of the MVDR, but it also compromises the performance as the noise is assumed to be uncorrelated across microphones. In [12] the MVDR was computed using a message passing algorithm [13]. However, this requires the network topology to be consistent with the noise correlation matrix, where two nodes are neighbors if their noise cross correlation is unequal to zero. This would require to adjust the transmission range of the nodes in the network to the correlation matrix and consequently, increase the energy usage for transmission or decrease the connectivity in the network. In [5] and [6], computation of the inverse of the correlation matrix was overcome by employing the generalized sidelobe canceller structure. However, this algorithm also constrains the topology of the network to be fully connected.

The contribution of this paper is twofold. First, we present a method where each user can estimate a different signal of interest from a mix of many different signals by means of a distributed MVDR beamformer without the need to reveal the source of interest to other entities in the network. In order to do this, we develop an algorithm that enables distributed estimation of the inverse of a correlation matrix, which is the second contribution of this paper. This algorithm for distributed estimation of the matrix inverse is based on the observation that in practice, correlation matrices are usually estimated recursively by exponential smoothing. Using the Sherman–Morrison formula [14], estimation of the inverse of the correlation matrix can be seen as a consensus problem and can be realized using gossip algorithms. Although the convergence error per time frame is decreased with increasing number of iterations when using gossip algorithms, estimation errors might accumulate. This is caused by the fact that the correlation matrix is recursively estimated across time. These convergence errors can be eliminated using

the distributed clique-based algorithm that we propose in this paper. The performance of the proposed clique-based distributed estimation of the inverse correlation matrix is compared with the centralized estimation approach in terms of data transmissions in the scenario at hand where users want to estimate their signal of interest without revealing this to other entities. In addition, we show that this algorithm for distributed matrix inverse estimation can be used in the privacy preserving scenario to estimate a certain signal of interest.

The remainder of this paper is organized as follows. In Section 2 we introduce the notation that we will use throughout this paper and describe the problem. To guide the reader, we give in Section 3 a brief overview of gossip based algorithms. In Section 4 we show how estimation of the inverse correlation matrix can be seen as a consensus problem, and in Section 5 we introduce a framework to compute a privacy preserving MVDR beamformer in a distributed way, after which we show in Section 6 how this can be turned into a distributed estimation problem using gossip techniques. Then, in Section 7 we introduce a clique-based distributed algorithm in order to reduce the convergence error of the estimated inverse correlation matrix, which might otherwise accumulate across time. In Section 8 we present simulation results to demonstrate the presented algorithm and compare its performance in terms of computational costs with centralized estimation. Finally, in Section 9 conclusions are drawn.

2. Notation and problem description

Let $Y_m(f,k)$ denote a degraded speech short-time discrete Fourier transform (DFT) coefficient obtained on a frame-by-frame basis at a microphone with index-number m, frequency-bin index f and time-frame index k. The challenge for a speech enhancement algorithm is to estimate the underlying clean speech, given realizations of the noise+target DFT coefficients.

A common model that often underlies such algorithms is an additive noise model where the different sources are assumed to be mutually uncorrelated. Let $S_m(f,k)$ and $V_m(f,k)$ denote the target and the disturbance DFT coefficient, respectively. The noise+target DFT coefficients are then given by

$$Y_m(f,k) = S_m(f,k) + V_m(f,k), \tag{1}$$

with $S_m(f,k) = d_m(f,k)S(f,k)$ and $S(f,k)$ being the clean speech at the target location, and $d_m(f,k)$ being the acoustic transfer function. Let index i denote a user (i.e., node) in the network. In the scenario that we consider, each user (i.e., each node) in the network can have a different target source $S(f,k)$, say $S_i(f,k)$, thus with a different acoustic transfer function (notice that this allows multiple microphones per node/user). The remaining sources are considered as disturbance (noise) for this user and are symbolized by the disturbance or noise DFT coefficient $V_i(f,k)$. What is considered to be noise for one user might be the target signal for another user. As such, (1) is different for all users. However, to simplify notation, we consider here the noise model for one specific user.

The target and noise DFT coefficients are often assumed to be independent across time and frequency. This allows us to omit the time and frequency indices for notational convenience. Further, we will use a stacked vector notation, that is, $\mathbf{Y} = [Y_1, \ldots, Y_M]^T$, with M being the total number of microphones in the network and $(\cdot)^T$ denotes the transposition of a vector or a matrix. We use bold symbols to represent vectors or matrices, while scalars are denoted by non-bold symbols. For symbols representing random variables, we use the upper case to denote the random variable, and the corresponding lower case to denote its realization. The speech and noise vectors $\mathbf{S}$ and $\mathbf{V}$ are defined in the same way as $\mathbf{Y}$. Let $\mathbf{d} = [d_1, \ldots, d_M]^T$ denote the steering vector representing the acoustic transfer function from the speech source to all microphones. Altogether this gives for one specific user

$$\mathbf{Y} = \mathbf{d}S + \mathbf{V} = \mathbf{S} + \mathbf{V}.$$

We assume that the M microphones in the WASN are grouped in N nodes. Each node has M_i microphones with $M = \sum_{i=1}^{N} M_i$. The different nodes in the network are connected via wireless links, while the microphones within the same node are assumed to be connected via wired connections. Each node in the network symbolizes a different device in the network potentially owned by a different user (hearing aid, mobile phone, etc.).

The goal of a multi-microphone noise reduction algorithm is to make an estimate of the clean speech DFT coefficient, say $\hat{S}$. Although many alternatives exist, an often used multi-microphone noise reduction algorithm is the MVDR beamformer. The MVDR beamformer is given by [2]

$$\mathbf{w} = \frac{\mathbf{R}_{\mathbf{Y}}^{-1}\mathbf{d}}{\mathbf{d}^{\mathbf{H}}\mathbf{R}_{\mathbf{Y}}^{-1}\mathbf{d}} \tag{2}$$

where $\mathbf{R}_{\mathbf{Y}} = E[\mathbf{Y}\mathbf{Y}^H]$, with $E[\cdot]$ denoting the statistical expectation operator, and $(\cdot)^H$ denoting the Hermitian transposition. Similarly we define $\mathbf{R}_{\mathbf{V}} = E[\mathbf{V}\mathbf{V}^H]$. Alternatively, applying the matrix inversion lemma to (2) in combination with the assumption that target and noise are uncorrelated, the MVDR beamformer can also be written as

$$\mathbf{w} = \frac{\mathbf{R}_{\mathbf{V}}^{-1}\mathbf{d}}{\mathbf{d}^{\mathbf{H}}\mathbf{R}_{\mathbf{V}}^{-1}\mathbf{d}}. \tag{3}$$

The problem statement in this paper is to allow all users in the network to estimate their own signal of interest in a distribute way using a WASN with the MVDR beamformer by means of (2), while keeping the source of interest private. This implies that the steering vector $\mathbf{d}_i$ for user (node) i should not be shared with other users. In practical applications, the steering vector $\mathbf{d}_i$ has to be estimated. In order to concentrate on the distributed estimation of the inverse of the correlation matrix, we assume here that each user i knows the steering vector towards the source of his or her interest. Among other methods, the steering vector can be determined by estimation of the microphone locations and location/direction of the source of interest or by estimation of the relative transfer function [15].

By keeping the steering vector private, each node i can estimate its own source of interest S_i without revealing to other users which particular source this is. In this scenario it is thus the exact linear combination specified by the steering vector $\mathbf{d}_i$ that is kept secret. With this assumption, an estimate of the target S_i can be obtained as

$$\hat{S}_i = \frac{\mathbf{d}_i^H \mathbf{R}_\mathbf{Y}^{-1} \mathbf{Y}}{\mathbf{d}_i^H \mathbf{R}_\mathbf{Y}^{-1} \mathbf{d}_i}. \tag{4}$$

We will consider distributed estimation of the MVDR beamformer using the noise+target correlation matrix $\mathbf{R}_\mathbf{Y}$ in the remaining part of this paper. In case the objective is to estimate the MVDR based on the noise correlation matrix (as in (3)), the proposed algorithm can be combined with a voice activity detector (VAD) to distinguish between noise+target and noise-only segments.

An often used procedure to estimate the correlation matrix is recursive exponential smoothing, that is,

$$\hat{\mathbf{R}}_\mathbf{Y}(k) = \lambda \hat{\mathbf{R}}_\mathbf{Y}(k-1) + (1-\lambda)\mathbf{Y}(k)\mathbf{Y}^H(k), \tag{5}$$

where $0 \le \lambda \le 1$ denotes the exponential weighting factor and $\hat{\mathbf{R}}_\mathbf{Y}(k)$ denotes an estimate of $\mathbf{R}_\mathbf{Y}$ at time-frame k. With conventional centralized processing, this operation would be performed in a fusion center, where the observations for all nodes are gathered and the correlation matrix is estimated and transmitted to other nodes in the network that would require this estimate. In this work we employ an alternative way to estimate the inverse correlation matrix based on the Sherman–Morrison formula. This appears to be an important aspect, as it not only enables us to compute the inverse correlation matrix in a distributed way, but also enables us to compute the MVDR beamformer in a distributed fashion without the need to share the steering vector with other users.

3. Gossip algorithms

To guide the reader, this section presents a brief overview of gossip algorithms. Gossip algorithms have been widely studied for in-network information processing in wireless sensor networks [11]. They can be used to solve consensus problems in a distributed way without any requirement of network topology. Given a randomly connected network of N nodes and an initial value $g_i(0)$ at each node i, a possible objective of a gossip algorithm could be to estimate the average value $g_{\text{ave}} = (1/N)\sum_{i=1}^{N} g_i(0)$ of the initial values at each node i by using only local processing. By allowing neighboring nodes to exchange information and update this with a convex combination of their own and neighboring values (i.e., a linear combination of points with non-negative weights that sum up to one), convergence will be reached under certain conditions.

Gossip algorithms can be categorized into two classes: randomized, where each pair of neighboring nodes is chosen randomly based on a probabilistic model to update information; and deterministic, where neighboring nodes are chosen in a deterministic way (e.g., by using knowledge on the network topology) to update information. With deterministic gossip the consensus will be achieved asymptotically, and with randomized gossip, consensus will be achieved asymptotically almost surely [16]. In typical gossip algorithms, nodes in the connected network make a convex combination of their own value and their values received from their neighbors. Let $g_i(t)$ denote the value of node i at the end of iteration t. In a given time-slot t, a typical iteration of a gossip algorithm consists of the selection of multiple nodes, e.g., the pair (i,j), and communication and update of their estimates, for example, $g_i(t) = g_j(t) = (g_i(t-1) + g_j(t-1))/2$. Depending on the exact protocol, the averaging operations can be performed asynchronously or synchronously.

4. The estimated correlation matrix

This section discusses that estimation of the inverse correlation matrix can be seen as a consensus problem.

The Sherman–Morrison formula [14] provides an explicit formula for the inverse of a matrix $\mathbf{B} = \mathbf{A} + \mathbf{u}\mathbf{v}^T$, where $\mathbf{A}$ is an invertible $M \times M$ matrix and $\mathbf{u}$ and $\mathbf{v}$ are the M-dimensional column vectors. Matrix $\mathbf{B}$ is invertible if and only if $1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u} \neq 0$. In this case, the Sherman–Morrison formula is given by

$$\mathbf{B}^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{A}^{-1}}{1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}}. \tag{6}$$

From (5) and (6), the inverse correlation matrix $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k)$ can be obtained as [17]

$$\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k) = \lambda^{-1}\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1) - \frac{\lambda^{-2}(1-\lambda)\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k)\mathbf{Y}^H(k)\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)}{1 + \lambda^{-1}(1-\lambda)\mathbf{Y}^H(k)\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k)}. \tag{7}$$

Notice that the computational complexity of (7) in each time-frame is only $O(M^2)$ when a previous time-frame estimate $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)$ is available, while the computational complexity of directly computing the inverse in $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k)$ is $O(M^3)$. Besides the lower computational complexity of (7) over the inverse that results from (5), the structure of (7) makes it possible to estimate the inverse correlation matrix in a distributed fashion. More specifically, each node will have an estimate $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)$ available from the iteration performed in the previous time frame $k-1$. To compute $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k)$, it is required to compute $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k)$ and $\mathbf{Y}^H(k)\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k)$ in a distributed way.

Let $\mathbf{r}_1, \ldots, \mathbf{r}_M$ be the columns of $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)$. We then have

$$\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k) = [\mathbf{r}_1, \ldots, \mathbf{r}_M]\mathbf{Y}(k) = \sum_{m=1}^{M} \mathbf{r}_m Y_m, \tag{8}$$

i.e., a weighted sum of the columns of $\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)$, where the weights are determined by the noise+target DFT coefficients in $\mathbf{Y}(k)$. In addition, let $\mathbf{a} = \hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k)$. We can then write

$$\mathbf{Y}^H(k)\hat{\mathbf{R}}_\mathbf{Y}^{-1}(k-1)\mathbf{Y}(k) = \mathbf{Y}^H(k)\mathbf{a} = \sum_{m=1}^{M} Y_m^* a_m, \tag{9}$$

which is a weighted sum over $\mathbf{a}$ with the noise+target DFT coefficients as weights. Obviously, given $\mathbf{Y}(k)$ and

$\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$, the expressions in (8) and (9) can also be seen as averaging operations, if a normalization over the number of nodes N would be included. Such averaging operations can be computed in a distributed manner using gossip algorithms. To do so, we need two rounds of gossip iterations. The first gossip round is used to compute

$$\mathcal{E}_{\text{ave}}^{1}(k)=\frac{1}{N}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k), \tag{10}$$

and the second round is used to compute

$$\mathcal{E}_{\text{ave}}^{2}(k)=\frac{1}{N^2}\mathbf{Y}^{H}(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k). \tag{11}$$

For notational convenience we will denote intermediate estimates of $\mathcal{E}_{\text{ave}}^{1}(k)$ in frame k, iteration t and node i by $\mathcal{E}_{i,t}^{1}(k)$ and intermediate estimates of $\mathcal{E}_{\text{ave}}^{2}(k)$ in frame k and iteration t by $\mathcal{E}_{i,t}^{2}(k)$.

5. Distributed privacy preserving MVDR computation

This section introduces an approach to perform a distributed MVDR beamformer based on the presented distributed estimation of the inverse correlation matrix with privacy preservation of a source of interest. We first demonstrate how any user in the network can employ the presented framework for (distributed) matrix inverse estimation in order to compute an MVDR beamformer in a distributed fashion without revealing their source of interest.

Given that (8) and (9) are computed in a distributed fashion, every user can compute an estimate of the inverse correlation matrix by means of (7), that is $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$. By means of (8), every user has a local estimate of $\mathbf{a}=\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$. Given that the user knows the steering vector for his source of interest, say $\mathbf{d}_i$, the target that is of interest for the user at node i can be estimated as $\hat{S}_i=\mathbf{d}_i^H\mathbf{a}/\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{d}_i$. Here $\mathbf{d}_i$ denotes the steering vector towards the source of interest for user i (at node i). Both $\mathbf{a}$ and $\mathbf{R}_{\mathbf{Y}}$ are computed in distributed fashion using gossip techniques. This means that no shared central processor is needed, but that every user has its own estimates of $\mathbf{a}$ and $\mathbf{R}_{\mathbf{Y}}$. The steering vector $\mathbf{d}_i$ is only known locally by the user. In this way, every user can compute his signal of interest without sharing the steering vector. An alternative to the presented distributed MVDR would be the use of one centralized MVDR that calculates $\mathbf{R}_{\mathbf{Y}}^{-1}$ in one centralized node and subsequently transmits this matrix together with $\mathbf{Y}(k)$ to all other nodes. However, as will be shown in the analysis in Section 6, this will lead to a much larger transmission cost compared to distributed computation. Using distributed computations, intermediate calculations can be performed that compress the data prior to transmission.

6. Gossip-based distributed estimation of the correlation matrix

In this section, we first discuss a gossip-based algorithm for distributed estimation of the inverse correlation matrix. Next, we give a convergence error analysis of the algorithm and show that the convergence errors accumulate across time due to the recursive estimation procedure. Therefore, we present in Section 7 an alternative approach that eliminates the accumulating convergence error. Although our interest is to estimate the inverse correlation matrix, the error analysis will be based on the correlation matrix as this will be more insightful.

6.1. Estimation of $\mathbf{R}_{\mathbf{Y}}^{-1}(k)$ using gossip

Let $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$ denote the estimated inverse of the correlation matrix at node i and time-frame k. To estimate $\mathbf{R}_{\mathbf{Y}}^{-1}(k)$ recursively as given in (7), we assume that at each node i initializes the inverse of the noise+target correlation matrix as $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(0)=\mathbf{I}$, where $\mathbf{I}$ is a $M\times M$ dimensional unit matrix. This requires that the dimension M, i.e., the total number of microphones in the network is known. When this is unknown, it can be estimated using gossip based techniques, see e.g., [18] and references therein.

Before starting gossip iterations between nodes in a time-frame k, first the initial value $\mathcal{E}_{i,0}^{1}(k)$ needs to be determined for each node i. This is obtained by computing $\mathcal{E}_{i,0}^{1}(k)=\sum_{m\in M_i}\mathbf{r}_m Y_m$. Notice that $\mathcal{E}_{i,0}^{1}(k)$ is an $M\times 1$ dimensional vector, since $\mathbf{r}_m$ is $M\times 1$ dimensional vector. Then, gossip iterations can be used to estimate $(1/N)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ in frame k in a distributed manner, by computing the average in each iteration t between two nodes i and j, i.e., $(\mathcal{E}_{i,t}^{1}(k)+\mathcal{E}_{j,t}^{1}(k))/2$. Let $\mathcal{E}_{i,T_1}^{1}(k)$ denote the final estimate of round 1 at node i and iteration T_1. According to the convergence properties of gossip algorithms, the estimates $\mathcal{E}_{i,T_1}^{1}$ at all nodes are guaranteed to converge to the average value (10) after a sufficient number of iterations.

As soon as $\mathcal{E}_{i,T_1}^{1}(k)$ is known accurately enough at all nodes, a second gossip round can be started to estimate $(1/N^2)\mathbf{Y}^{H}(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$. First, each node i determines the initial value $\mathcal{E}_{i,0}^{2}(k)$, that is, $\mathcal{E}_{i,0}^{2}(k)=\sum_{m\in M_i}Y_m^{*}[\mathcal{E}_{i,T_1}^{1}(k)]_m$, where $[\cdot]_m$ indicates the mth element of the corresponding vector. Given $\mathcal{E}_{i,0}^{2}(k)$, gossip iterations can be performed to estimate $\mathcal{E}_{\text{ave}}^{2}(k)$, by computing the average in each iteration t between two nodes i and j, e.g., $(\mathcal{E}_{i,t}^{2}(k)+\mathcal{E}_{j,t}^{2}(k))/2$. The final $\mathcal{E}_{i,T_2}^{2}$ at all nodes are guaranteed to converge to the average value (11) when using enough iterations T_1 and T_2 in both gossip rounds. Notice that the convergence error in the second estimation round depends on the convergence error in the first estimation round. Based on the estimates of $\mathcal{E}_{i,T_1}^{1}(k)$ and $\mathcal{E}_{i,T_2}^{2}(k)$, each node i can locally update the estimate $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$ using (7) as

$$\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)=\lambda^{-1}\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k-1)-\frac{\lambda^{-2}(1-\lambda)N^2\mathcal{E}_{i,T_1}^{1}(k)\mathcal{E}_{i,T_1}^{1,H}(k)}{1+\lambda^{-1}(1-\lambda)N^2\mathcal{E}_{i,T_2}^{2}(k)}. \tag{12}$$

6.2. Convergence error analysis

To assess the performance of the gossip-based distributed estimation algorithm, we define the squared error

(SE) between the inverse of the gossip-based distributed estimate of the inverse correlation matrix and the centralized optimal estimate of the correlation matrix for a given time-frame k as

$$\mathrm{SE}(k) = \|\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) - \hat{\mathbf{R}}_{\mathbf{Y},c}(k)\|_{\mathrm{fro}}, \tag{13}$$

where $\|\cdot\|_{\mathrm{fro}}$ denotes the Frobenius norm, $\hat{\mathbf{R}}_{\mathbf{Y},c}(k)$ denotes the centralized estimated correlation matrix using (5) and $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k)$ is the inverse of the estimated inverse correlation matrix using T_1 and T_2 iterations in the first and second gossip rounds, respectively. Depending on T_1, T_2 and the network topology, a convergence error might be introduced by the gossip-based algorithm, and as the matrix inverse is computed recursively across time, these errors might accumulate. To investigate this, we define the error introduced by the gossip operation at time-frame k by the matrix Δ_k as

$$\Delta_k = \hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) - \lambda\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k-1) - (1-\lambda)\mathbf{Y}(k)\mathbf{Y}^H(k). \tag{14}$$

From (14) in combination with the initial value $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0)$, we can write $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k)$ as

$$\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) = \sum_{n=1}^{k} \lambda^{k-n}\Delta_n + \lambda^k\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0) + \sum_{n=1}^{k} \lambda^{k-n}(1-\lambda)\mathbf{Y}(n)\mathbf{Y}^H(n). \tag{15}$$

Further, we can write $\hat{\mathbf{R}}_{\mathbf{Y},c}(k)$ as

$$\hat{\mathbf{R}}_{\mathbf{Y},c}(k) = \lambda^k\hat{\mathbf{R}}_{\mathbf{Y},c}(0) + \sum_{n=1}^{k} \lambda^{k-n}(1-\lambda)\mathbf{Y}(n)\mathbf{Y}^H(n). \tag{16}$$

From (15) and (16) in combination with (13) and the fact that $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0) = \hat{\mathbf{R}}_{\mathbf{Y},c}(0) = \mathbf{I}$, it then follows that

$$\mathrm{SE}(k) = \left\| \sum_{n=1}^{k} \lambda^{k-n}\Delta_n \right\|_{\mathrm{fro}}. \tag{17}$$

Eq. (17) shows that the SE depends on the summation of the gossip error across all time frames, which indicates that the SE between the output of the gossip-based distributed estimation and the output of the centralized estimation accumulates with increasing number of time frames. The reason that the SE accumulates across time frames is that the correlation matrix is recursively updated across time frames and the gossip algorithms have a convergence error at each time frame. This depends on the number of iterations and on the way the sequence of gossip operations is performed. Although it is interesting to analyze the changes of the SE as a function of time frames, it is not straightforward to do this using analytic expressions. In Section 8, we will use simulations to illustrate the SE behavior versus time frames.

7. Clique-based distributed estimation of the inverse correlation matrix

One way to eliminate the convergence error described in Section 6.2 is to make sure that the gossip algorithm aggregates the information from all nodes. To do this in an efficient manner, we first compress the graph using non-overlapping cliques as previously presented in [19] for gossip-based estimation. For this compressed graph we then determine the spanning tree.

A clique is a fully connected sub-graph. The cliques of a graph $\mathcal{G}$ can be overlapping, since each node can belong to multiple cliques. Here we consider non-overlapping cliques only, where each node belongs to only one clique. In this section, we first present a clique-based distributed (CbD) algorithm based on the non-overlapping cliques of a graph $\mathcal{G}$. Then we study the performance of the CbD algorithm and compare the clique-based distributed estimation of the inverse correlation matrix with a centralized estimation algorithm in terms of required data transmissions. The performance comparison will be made for a fully connected and string connected network.

7.1. Clique-based distributed algorithm

In [19] it was proposed how a method determines all non-overlapping cliques in a distributed fashion given a randomly connected graph $\mathcal{G}$. We assume that there are C non-overlapping cliques in graph $\mathcal{G}$. These cliques can be used to compress the original graph $\mathcal{G}$ by representing each clique by a single node in $\mathcal{G}_1$. An example is given in Fig. 1. The nodes are here denoted by g_i, with i the node index, while the cliques are denoted by h_j, with j the index of the clique. Notice that graph $\mathcal{G}_1$ is a compressed version of the original graph $\mathcal{G}$, since all nodes in a clique are represented by a single node in $\mathcal{G}_1$ and multiple edges between two neighboring cliques are compressed into a single edge in $\mathcal{G}_1$. In the compressed graph $\mathcal{G}_1$, two cliques are said to be neighbors if there is at least one direct link joining them. If more than one such links exist, only one of them is activated by random selection. The end nodes of active links are called gateway nodes. To eliminate the convergence error of gossip-based distributed algorithms and reduce the computational complexity of the CbD algorithm, the compressed graph $\mathcal{G}_1$ is further pruned to a spanning tree. Many approaches were proposed to define and compute spanning trees, see e.g., [20]. Let $\mathcal{G}_t$ denote a tree graph which is pruned from the compressed graph $\mathcal{G}_1$, let L denote the total number of levels of the graph $\mathcal{G}_t$ and let C^l denote the number of cliques in the lth level of $\mathcal{G}_t$. We assume that each node i in $\mathcal{G}$ has an initial value $g_i(0)$. In the current case of estimating $N\mathcal{E}^1_{\mathrm{ave}}$, $g_i(0)$ is given by $g_i(0) = \mathcal{E}^1_{i,0}(k) = \sum_{m\in M_i}\mathbf{r}_m Y_m$. Based on the tree graph $\mathcal{G}_t$, the CbD algorithm is described in Table 1. For the given initialization, this will finally lead to $N\mathcal{E}^1_{\mathrm{ave}}$. In a similar way $N^2\mathcal{E}^2_{\mathrm{ave}}$ can be estimated by initializing $g_i(0)$ as $g_i(0) = \sum_{m\in M_i} Y^*_m[N\mathcal{E}^1_{\mathrm{ave}}(k)]_m$.

It is worth pointing out that the CbD algorithm can reach the summation of all initial node values in the network in a distributed manner. Thus, using the CbD algorithm, exact values for $N\mathcal{E}^1_{\mathrm{ave}}$ and $N^2\mathcal{E}^2_{\mathrm{ave}}$ in (10) and (11), respectively, are obtained, while the randomized gossip approach described in the previous section will always have a (small) convergence error. In combination with the Sherman–Morrison formula (6), the inverse correlation matrix can be obtained in a distributed manner in a similar way as with (12). We refer to this distributed algorithm as a clique-based distributed estimation of the correlation matrix (CbDECM).

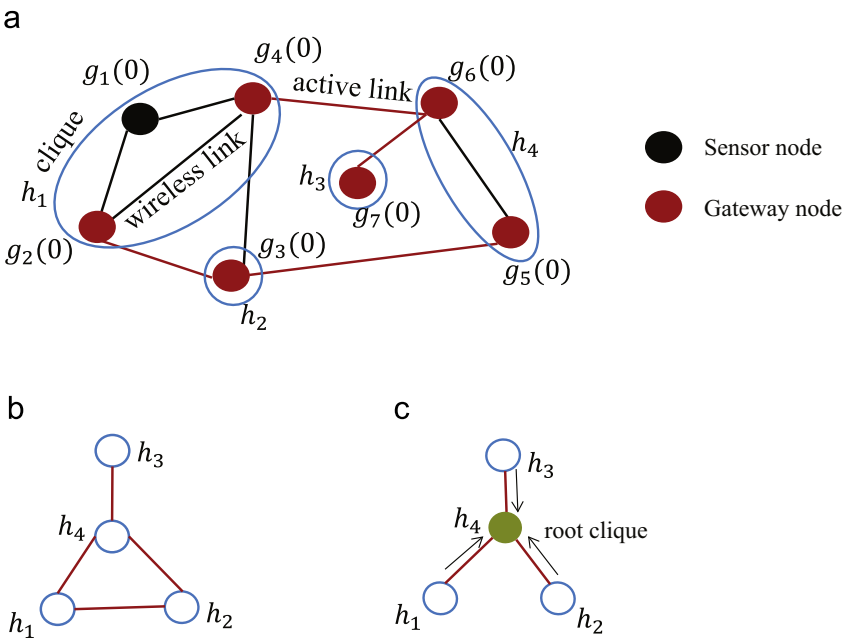

Fig. 1. (a) The original graph $\mathcal{G}$. (b) The compressed graph $\mathcal{G}_1$. (c) The tree graph $\mathcal{G}_t$.

Table 1
CbD algorithm.

1. Initialize the level index $l=1$ and $g_i = g_i(0)$, where g_i denotes the current value of node i
2. Each clique c^l, where c^l denote a clique in the lth level, updates its estimation as $h_{c^l} = \sum_{i \in c^l} g_i$. To do so, each node i in the clique c^l broadcasts its value g_i to all other nodes in the clique
3. Each clique c^l sends its estimates h_{c^l} to its neighboring clique c^{l+1} via gateway nodes one level up in the tree. The gateway node $j \in c^{l+1}$ updates its estimation as $g_j = g_j(0) + h_{c^l}$
4. $l \rightarrow l+1$
5. Return to step 2 until $l=L$
6. The root clique c^L at the top level updates its estimation as $h_{c^L} = \sum_{i \in c^L} g_i$ and sends the updated estimation h_{c^L} back to all other nodes in the lower levels of $\mathcal{G}_t$

7.2. Transmission cost analysis

This subsection discusses the required number of transmissions of the CbD algorithm. We assume that one transmission is the sending of a scalar value from one node to another. Given a connected network $\mathcal{G}$, the required number of transmissions of the CbD algorithm will be explained below and is given by $T_d = \sum_{l=1}^{L} t^l$ with

$$t^l = \begin{cases} \sum_{c^l=1}^{\hat{C}^l} K_{c^l} + C^l & 1 \le l < L, \\ K_{c^L} + 2(C-1) & l = L, \end{cases} \tag{18}$$

where t^l is the required number of transmissions of the CbD algorithm in the lth level of $\mathcal{G}_t$, K_{c^l} is the number of nodes in a clique c^l, and $\hat{C}^l$ is the number of non-overlapping cliques which consist of more than one node. Notice that $\hat{C}^l \le C^l$, since C^l is the total number of non-overlapping cliques in the lth level of $\mathcal{G}_t$.

In $\mathcal{G}$, C non-overlapping cliques can be determined and represent C nodes in $\mathcal{G}_1$. The compressed graph $\mathcal{G}_1$ is then pruned to a tree graph $\mathcal{G}_t$ and the C cliques are distributed in the different levels of $\mathcal{G}_t$. For the CbD algorithm, we go through the tree $\mathcal{G}_t$ from the lowest level, $l=1$ up to the root L. When l is smaller than L, each node i in a clique c^l broadcasts their data to all other nodes in the clique. The required number of transmissions for all nodes in the lth level is $\sum_{c^l=1}^{\hat{C}^l} K_{c^l}$. In addition, each clique c^l needs one transmission to send the updated estimates to its neighboring clique c^{l+1} in the $(l+1)$ th level. Thus, the required number of transmissions in lth $(l<L)$ level is equal to $\sum_{c^l=1}^{\hat{C}^l} K_{c^l} + C^l$.

At the top level of $\mathcal{G}_t$, there is only one clique which is the root clique of $\mathcal{G}_t$. Similar to the other cliques, each node i in the root clique broadcasts their data to all other nodes in the clique. On the other hand, the root clique has to send the updated estimates to all other nodes in the lower levels. Since the number of remaining cliques in $\mathcal{G}_t$ is $C-1$ and each clique requires one transmission to receive data from its neighboring clique and one transmission to synchronize all nodes in this clique with the updated data,

the required number of transmissions for sending data from the root clique to all other nodes is $2(C-1)$. Therefore, at the top level of $\mathcal{G}_t$, the total required number of transmissions is $K_{c^t}+2(C-1)$.

Notice that t^l is the upper bound of the required number of transmissions, since there are some cliques that might consist of only one node. Moreover, the CbD algorithm costs one transmission less when a gateway node has two neighboring gateway nodes in two different cliques.

7.3. Performance comparison

This subsection analyzes the relative performance between the proposed CbDECM algorithm and a centralized way for matrix inverse estimation in terms of data transmissions. This comparison is done in the given scenario where the source of interest (by means of the steering vector) is considered to be private. In such a scenario, each user is in need of the estimated inverse correlation matrix, i.e., $\hat{\mathbf{R}}_Y^{-1}$, as well as $\hat{\mathbf{R}}_Y^{-1}\mathbf{Y}$. Given knowledge of these two quantities, in combination with the locally known steering vector $\mathbf{d}_i$, every user (node) can estimate his source of interest by constructing the MVDR beamformer (by means of $\hat{S}_i=\mathbf{d}_i^H\hat{\mathbf{R}}_Y^{-1}\mathbf{Y}/\mathbf{d}_i^H\hat{\mathbf{R}}_Y^{-1}\mathbf{d}_i$). Although the proposed algorithm also inherently delivers an estimate of $\hat{\mathbf{R}}_Y^{-1}\mathbf{Y}$, we concentrate this comparison solely on the number of data transmissions required to compute the matrix inverse at each user's processor. For the centralized approach, which is our reference method, we therefore first gather all data at a single processor (which is assumed to be one of the nodes), after which the inverse correlation matrix is computed in a centralized fashion according to (7) and transmitted back to all users such that it can subsequently be used to estimate their signal of interest. Similarly, for the proposed approach, where (7) is computed in a distributed way based on the CbD algorithm presented in this section.

Consider a WASN with given size, the number of data transmissions for both the CbDECM and the centralized algorithm depends on the network topology. However, in practice, the exact network topology is unknown, complicating a comparison using analytic expressions in a randomly connected network. We therefore compare the CbDECM with the centralized algorithm in terms of the number of data transmissions in a fully connected network and a string connected network. In general, the fully connected topology has the best connectivity, while the string connected network has the worst connectivity. To do the performance comparison with analytic expressions, we assume that each clique in the network consists of K nodes and each node i has $M_i=u$ microphones. This means that there are $N=KC$ nodes and $M=Nu$ microphones in the network and the inverse correlation matrix is an $M\times M$ dimensional matrix. This assumption is only made for ease of analytical analysis, but not required in practice.

In a fully connected network, M–u data transmissions are needed to gather all observed signals in the central processor and then $M^2/2+M/2$ data transmissions are needed to send the lower or upper triangle of the estimated inverse correlation matrix back to all other nodes. Thus, the required number of data transmissions $T_{C,F}$ is given by

$$T_{C,F}=\frac{M^2}{2}+\frac{3M}{2}-u, \tag{19}$$

where the subscripts C and F indicate the centralized algorithm and a fully connected network, respectively. Further, we use subscripts D and S to indicate the CbDECM algorithm and a string connected network, respectively.

The CbDECM algorithm requires two rounds of processing in order to compute the sums in (8) and (9). As a fully connected network can be seen as a single clique, the compressed graph $\mathcal{G}_1$ will consist of just one node and tree pruning is not necessary. Thus, the number of levels is given by $L=1$. Based on (18), it then follows that for a scalar value it requires $T_d=K_{c^1}$ transmissions. As the number of cliques in a fully connected network is $C=1$, and the number of nodes in that clique is $K=N=M/u$, it requires $K=M/u$ transmissions for one scalar value. The distributed estimation of the matrix inverse requires an estimate of a vector of length M (by means of (8)) and a scalar value (by means of (9)), which then leads to M^2/u and M/u data transmissions, respectively. Thus, the total required number of data transmissions $T_{D,F}$ is given by

$$T_{D,F}=\frac{M^2+M}{u}. \tag{20}$$

To compare the computational cost of the CbDECM with the centralized algorithm in a fully connected network, their required number of data transmissions can be compared as

$$T_{C,F}-T_{D,F}=\frac{M^2u+3Mu-2M^2-2M}{2u}-u. \tag{21}$$

We assume that $M>2$. From (21), we then have that $T_{C,F}<T_{D,F}$ for $u=1$ and $T_{C,F}>T_{D,F}$ for $u\geq 2$. This indicates that the computational cost of the centralized algorithm is larger than those of the CbDECM algorithm if there is more than one microphone per node. Table 1 gives a numerical comparison between $T_{C,F}$ and $T_{D,F}$ for $M=500$ and $C=10$ and various combinations of K and u. This shows that for a fully connected network, and when the number of microphones per node is $u>1$, the proposed CbDECM algorithm always requires fewer data transmissions than the centralized approach.

In a string connected network where all C cliques are connected as a string, we assume that one node in the center of the string serves as the fusion center. The data is transmitted from both sides to this fusion center. The number of transmissions depends on C being odd or even. For compactness we only consider the case that C is even. For odd C a similar analysis can be carried out. When C is even, one side consists of $C/2$ cliques and the other side consists of $C/2-1$ cliques. The required number of data transmissions for both sides sending data to the gateway nodes in the central clique is $2Ku\sum_{c=1}^{C/2}c+2Ku\sum_{c=1}^{C/2-1}c$. Next, the required number of data transmissions for the central clique sending the data to the central node is

$Ku(C-1)+(K-1)u$. The central node updates the estimates of the inverse correlation matrix and sends the estimates back to all other nodes in $\mathcal{G}$. Since the correlation matrix is a symmetric matrix, the central node only needs to transmit the upper or lower triangle of the correlation matrix, which consists of $M^2/2+M/2$ variables. The required number of transmissions for all nodes to receive this information is $2(C-1)+1$, since the central node needs one transmission to broadcast the estimates to all other nodes in the central clique and $2(C-1)$ transmissions to send the data to the nodes in the remaining $C-1$ cliques. Here, each clique requires one transmission to send data to its neighboring clique and one transmission to synchronize all nodes in this clique with the updated estimates. Therefore, the required number of data transmissions of the centralized algorithm $T_{C,S}$ is given by

$$T_{C,S}=\left(C-\frac{1}{2}\right)M^2+\left(\frac{3}{2}C+\frac{1}{2}\right)M-u. \tag{22}$$

For the CbDECM algorithm, the data transmissions can be obtained using the transmission analysis given in Section 7.2. In a string connected network, there are $L=C$ levels of graph $\mathcal{G}_t$, and each level consists of only one clique (by means of $C^l=\hat{C}^l=1$). Since we assume that each clique consists of K nodes, we have $K_{c^l}=K$, $\forall l$. Thus, the required number of transmissions for the level $l<L$ is $K+1$. For the top level of $\mathcal{G}_t$, the required number of transmissions is $K+2(C-1)$. Combining the transmissions with the fact that an M-dimensional vector is transmitted per transmission in the first round and a scalar value is transmitted per transmission in the second round, the required number of data transmissions $T_{D,S}$ is given by

$$T_{D,S}=(M+1)\{(K+1)(C-1)+K+2(C-1)\}=(M+1)(N+3C-3). \tag{23}$$

We compare $T_{C,S}$ and $T_{D,S}$ for a given size network with $M=500$ and $C=10$. The required number of data transmissions of both the CbDECM and the centralized algorithm are given in Table 2 for various combinations of u and K, such that $M/C=Ku=50$.

This shows that the number of data transmissions for the proposed CbDECM algorithm is always smaller than for the centralized algorithm for the string connected network under the given parameter setting. It should be noted that this comparison in terms of required data transmissions is under the privacy preserving scenario where each user (processing node) is in need of the matrix inverse. If the constraints on the privacy preserving aspect are loosened, and the centralized processor directly estimates the target signal followed by transmission of the estimated target signal, the required transmissions are significantly reduced.

Table 2
The required number of data transmissions of the CbDECM algorithm and the centralized algorithm.

K, u	$T_{C,F}$	$T_{D,F}$	$T_{C,S}$	$T_{D,S}$
$K=1, u=50$	125 700	5010	2 382 700	18 537
$K=2, u=25$	125 725	10 020	2 382 725	23 547
$K=5, u=10$	125 740	25 050	2 3827 40	38 577
$K=10, u=5$	125 745	50 100	2 382 745	63 627
$K=25, u=2$	125 748	125 250	2 382 748	138 777
$K=50, u=1$	125 759	250 500	2 382 749	264 027

8. Simulations

In this section, we first illustrate the performance of the gossip-based algorithm and the CbDECM algorithm when estimating the inverse correlation matrix in a simulated WASN. Secondly, we demonstrate the use of the proposed CbDECM algorithm in combination with the MVDR.

8.1. Simulation environment

We simulate a wireless network with 6 cliques ($C=6$), where each clique consists of 3 acoustic sensor nodes ($K=3$ and $N=18$) and where each acoustic node consists of 2 microphones ($u=2$). The distance between the two microphones is 2 cm. The 6 cliques are (wirelessly) connected as depicted in the network in Fig. 2. A network like this could be obtained from a randomly connected network by first compressing it into a compressed network by finding the non-overlapping cliques in the network. Subsequently, the network can be pruned into a spanning tree. We consider a scenario where all sensor nodes, three speech sources and a noise source are placed in a 10 m × 8 m rectangular area. The overall node positions relative to the target signal are shown in Fig. 3. Furthermore, the speech sources consist of 30 s speech signals sampled at 16 kHz originating from the Timit database [21], and the noise source is a White Gaussian noise signal. To model the microphone-self noise, an independent additive white noise source is added to each microphone signal at the SNR of 40 dB measured at the microphone that is furthest away from the target source. In this paper, the simulation environment is assumed to be free of reverberation. The steering vectors $\mathbf{d}_i$ per user can then be calculated by gain and delay values as $\mathbf{d}_i=[a_{i,1}e^{-j\omega_f\tau_{i,1}},\ldots,a_{i,M}e^{-j\omega_f\tau_{i,M}}]^T$, where $a_{i,m}=1/l_{i,m}$ is the damping coefficient, and $\tau_{i,m}=(l_{i,m}/c)f_s$ is the delay in number of samples with $l_{i,m}$ being the distance between microphone m and the desired speech source and $c=340$ m/s being the speed of sound. The smoothing constant λ, see e.g., (5), is set at $\lambda=0.997$. All nodes process the signals in the frequency domain using frame-based processing, with a frame length of 32 ms and a 50%-overlapping Hann window. Notice that all simulations in the following subsections are performed according to the environment given in this subsection.

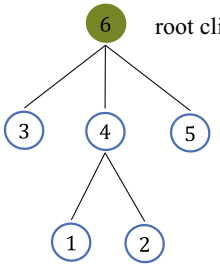

Fig. 2. Example of a network with tree topology with six cliques.

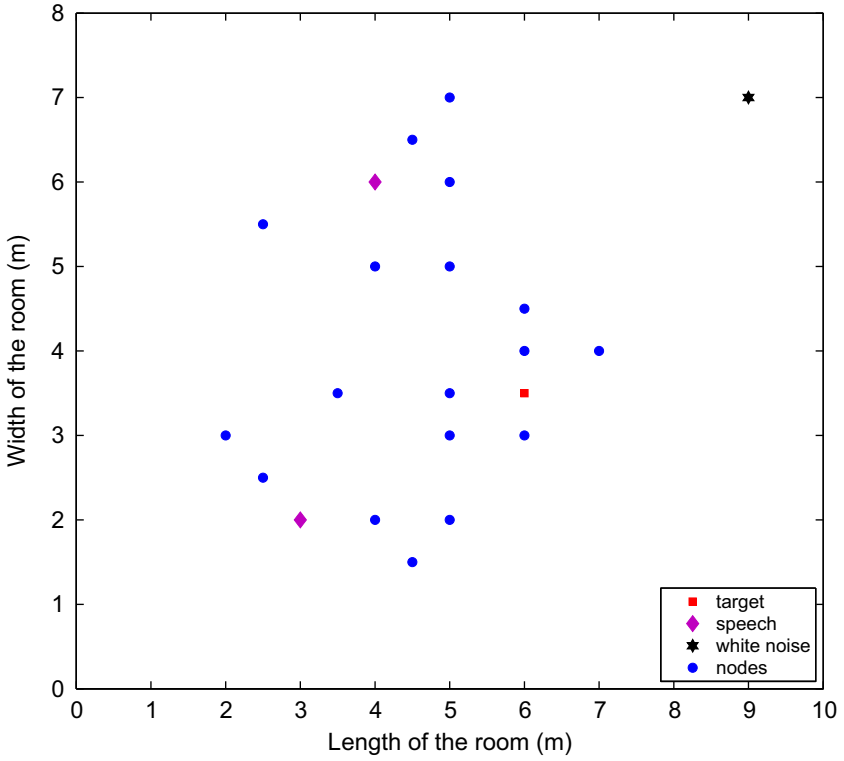

Fig. 3. WASN with 18 nodes, 3 speech sources and a noise source.

8.2. Estimation of $\mathbf{R}_{\mathbf{Y}}^{-1}$

In this subsection, we compare the two methods presented in Sections 6 and 7, that are, the gossip-based algorithm and the CbDECM algorithm for distributed estimation of the inverse correlation matrix, with a centralized estimator for the inverse correlation matrix.

The gossip operations in the gossip-based distributed processing algorithm are based on the clique-based algorithm presented in [19], which performs randomized gossip [11] across non-overlapping cliques.

To quantify the performance of the distributed estimation algorithms, we define the error between the estimated $\mathbf{R}_{\mathbf{Y}}^{-1}$ from the distributed algorithms and the estimation results from the centralized algorithm as

$$\mathrm{Err}_i(k) = 20\log_{10}\frac{1}{F}\sum_{f=1}^{F}\frac{\|\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(f,k)-\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(f,k)\|_{\mathrm{fro}}}{\|\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(f,k)\|_{\mathrm{fro}}}. \tag{24}$$

where $\mathrm{Err}_i(k)$ is the normalized square error at node i and frame k, and F denotes the number of frequency bins, and $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$ and $\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(k)$ denote the estimated inverse correlation matrix obtained from one of the distributed algorithms and the centralized algorithm, respectively. To aggregate information across frequency and time, $\mathrm{Err}_i(k)$ is averaged across time, that is

$$\mathrm{ME}_i = \frac{1}{\tilde{K}}\sum_{k=1}^{\tilde{K}}\mathrm{Err}_i(k), \tag{25}$$

with $\tilde{K}$ the number of time-frames.

Fig. 4 shows the error Err_1 between the gossip-based estimated inverse correlation matrix and the centralized estimated inverse correlation matrix for various numbers of randomized gossip iterations, and the error Err_1 between the inverse correlation matrix estimated using the proposed CbDECM method and a centralized estimate. It is observed that the error Err_1 when using the gossip-based distributed algorithm increases across time. This is in line with the convergence analysis given in Section 6.2. As expected, the error Err_1 is decreased by increasing the number of iterations. In addition, as the number of iterations is increased, the increase of the error across time slows down. Moreover, we observe that the CbDECM algorithm converges to a very accurate estimate of the inverse correlation matrix after an initial increase of the error (which is due to the floating-point relative accuracy of the Matlab).

Fig. 5 depicts the estimation performance of the gossip-based distributed estimation algorithm and the proposed CbDECM algorithm as a function of the number of data transmissions per time frame. From Fig. 5, we see that the error ME_1 of the gossip-based distributed estimation algorithm is decreased by increasing the number of data transmissions per time frame. In addition, we observe that both the transmission costs and the error of the gossip-based processing for distributed estimation of the inverse correlation matrix are higher than that for the proposed CbDECM algorithm.

8.3. Estimation of a target signal

This section discusses the performance when the proposed framework for distributed matrix inverse estimation

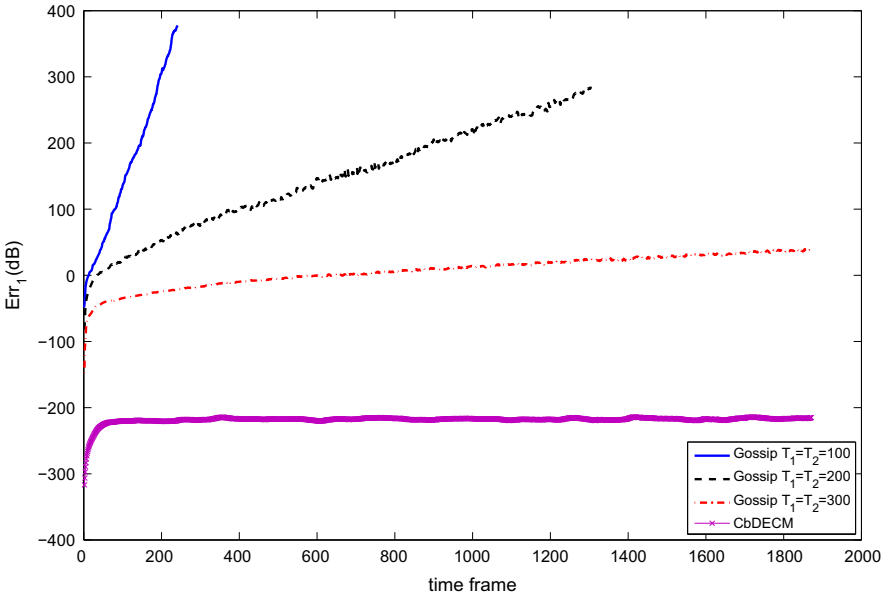

Fig. 4. The error Err of node 1 with 1 dB input SNR versus time frame.

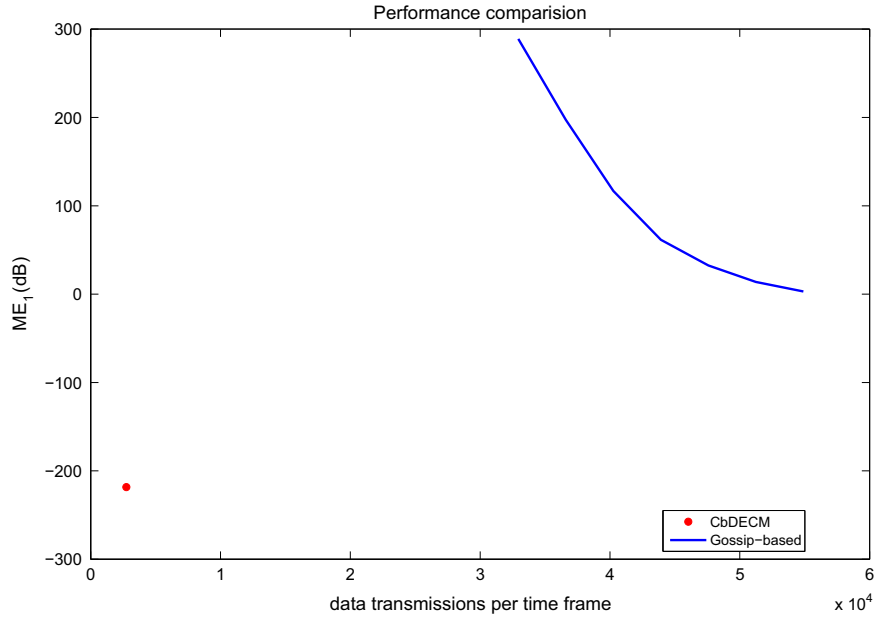

Fig. 5. The mean error ME of node 1 with 1 dB input SNR versus data transmissions per time frame.

is used in combination with an MVDR beamformer. The scenario in this experiment considers the case where the source of interest for a node i is private and selected from one of the available sources in the environment depicted in Fig. 3. The steering vector $\mathbf{d}_i$ from the desired source of node i to all microphones in the network is assumed to be known only locally. For comparison, we use the distributed delay and sum beamformer (DDSB) presented in [19]. This DDSB is essentially an MVDR defined as in (3) where $\mathbf{R_V}$ is assumed to be diagonal. This DDSB thus does not need the full correlation matrix, but is only in need of the diagonal elements, i.e., the noise PSD per microphone. The distributed MVDR beamformer with the CbDECM algorithm and the DDSB is based on similar setup where a clique-based graph is used and the steering vectors are assumed to be known a priori. To compare the distributed beamformers with their centralized versions and evaluate their performance, we also compare to the centralized MVDR (CMVDR) beamformer and the centralized delay and sum beamformer (CDSB) in this experiment. For the DDSB and the CDSB, the noise power spectral density (PSD) tracking algorithm in [22] is used to estimated the noise PSD. Moreover, for a fair comparison we set the number of data transmissions in the DDSB such that it is

equal to the number of required data transmissions in the CbDECM.

To compare the performance of the proposed CbDECM algorithm and the centralized estimation algorithm for estimating the inverse correlation matrix, we employ the estimated inverse correlation matrices of the CbDECM algorithm and the centralized algorithm in the MVDR beamformer as

$$\hat{S}_i(k)=\frac{\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)}{\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{d}_i}, \tag{26}$$

where $\hat{S}_i(k)$ is the frequency domain DFT coefficient of the beamformer output, and $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}$ is the estimated inverse correlation matrix. Notice that we explicitly mention here the dependency on time-frame k for clarity. Since each node in the network has the estimates of $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$, $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ and $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$ at time frame k, an alternative approach to incorporate the estimated inverse correlation matrix from frame k instead of $k-1$ as in (26) is given by

$$\hat{S}_i(k)=\frac{\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)(\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1))^{-1}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)}{\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)\mathbf{d}_i}. \tag{27}$$

Obviously, the computational complexity of (27) is higher, since (27) requires in addition to compute $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)(\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1))^{-1}$. To distinguish the estimation methods in (26) and (27), we denote the algorithm in (26) by CbDECM$_1$, and the algorithm in (27) by CbDECM$_2$.

To evaluate the performance of the proposed algorithm we compute the mean square-error (MSE) between the estimated desired signal and the clean speech signal. The MSE for node i is averaged over all time frames and frequency bins, and is give by

$$\mathrm{MSE}_i=\frac{1}{\tilde{K}F}\sum_{k=1}^{\tilde{K}}\sum_{f=1}^{F}|\hat{S}_i(f,k)-S_i(f,k)|^2, \tag{28}$$

where $S_i(f,k)$ is the frequency domain DFT coefficient of the desired speech signal of node i. In addition, we qualify the speech quality and the speech intelligibility of the estimated desired signal in terms of the segmental SNR and the short-time objective intelligibility measure (STOI) [23], respectively. The segmental SNR for node i is defined as

$$\mathrm{SNR}_{i,\mathrm{seg}}=\frac{1}{\tilde{K}}\sum_{k=1}^{\tilde{K}}10\log_{10}\frac{\sum_{f=1}^{F}|S_i(f,k)|^2}{\sum_{f=1}^{F}|\hat{S}_i(f,k)-S_i(f,k)|^2}. \tag{29}$$

Fig. 6(a) and (b) shows the noise reduction performance of the distributed beamformers and their centralized versions in terms of the segmental SNR and the MSE, respectively, while Fig. 6(c) show the instrumental speech intelligibility of the beamformer output. In Fig. 6(a) and (b), we observe that the speech quality of the CbDECM$_1$ is very close to the CMVDR and the CbDECM$_2$. More specifically, the difference between the segmental

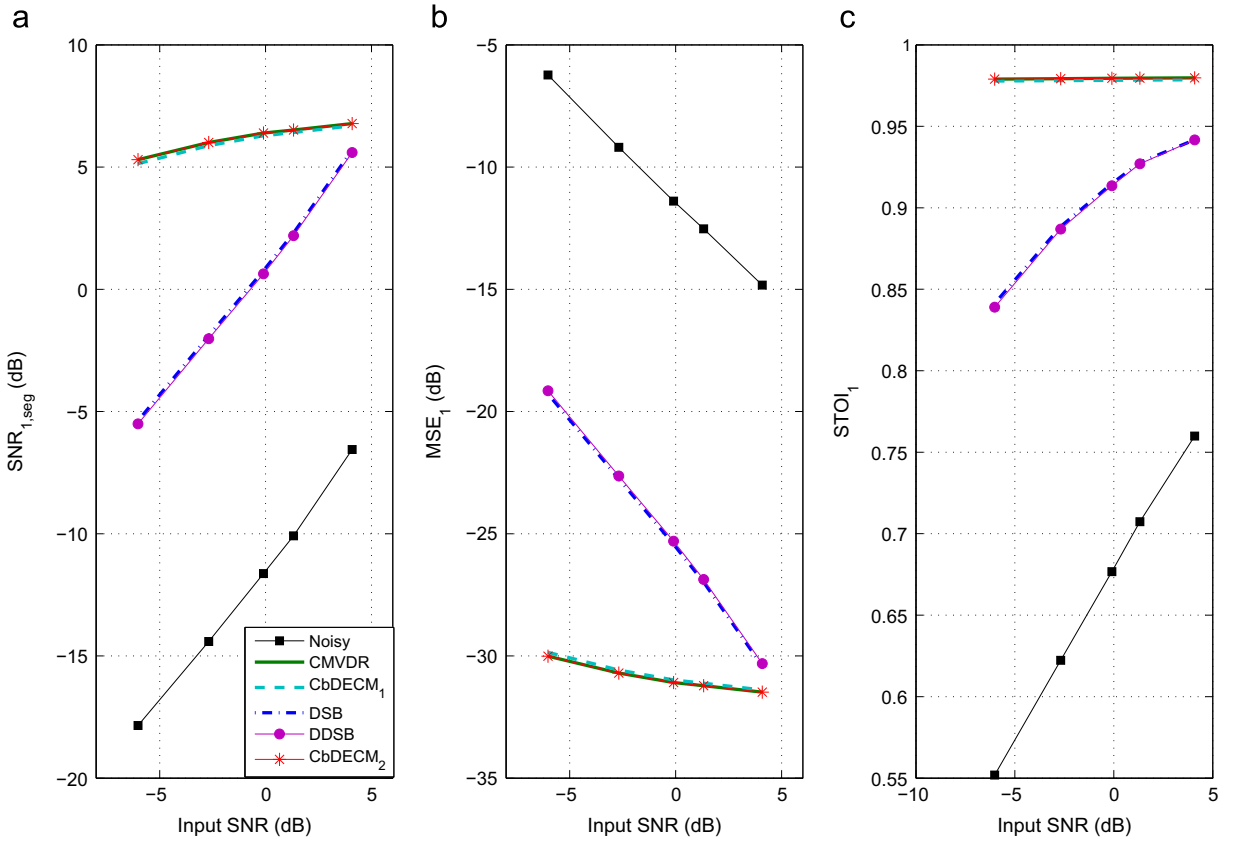

Fig. 6. (a) The segmental SNR of node 1 versus the global input SNR. (b) The MSE of node 1 versus the global input SNR. (c) The STOI of node 1 versus the global input SNR.

SNR of the CbDECM$_1$ algorithm and the CMVDR in Fig. 6(a) is decreased from 0.3 dB to 0.1 dB with increasing SNR of the noise+target input signal. Similarly, the difference between the MSE of the CbDECM$_1$ algorithm and the CMVDR in Fig. 6(b) decreases from 0.3 dB to 0.1 dB. This is reasonable since the CMVDR uses the estimated inverse correlation matrix of the current time-frame k to estimate the desired signal, while the CbDECM$_1$ algorithm uses the estimated inverse correlation matrix of the previous time frame $k-1$. Fig. 6(c) shows that the speech intelligibility in terms of STOI of the CbDECM$_1$ algorithm is identical to the CMVDR and the CbDECM$_2$ algorithm. All three figures in Fig. 6 show that both the speech quality and the speech intelligibility of the CbDECM$_2$ reach the same performance as the CMVDR. This can be explained by the fact that the estimated inverse correlation matrix of both CbDECM based algorithms converge to the estimated inverse correlation matrix of the centralized estimation algorithm. This is consistent with the experiment results given in Figs. 4 and 5. In addition, all three figures in Fig. 6 show that the improvement of the MVDR beamformers (CMVDR, CbDECM$_1$ and CbDECM$_2$) over the DDSB and the CDSB is increased from 1 to 10 dB in terms of the segmental SNR with increasing input SNR. A similar performance improvement is shown in terms of MSE and STOI. This should not come as a surprise, since the potential improvement of the MVDR beamformer is obtained by taking noise correlation into account. Moreover, all distributed beamformers (DDSB, CbDECM$_1$ and CbDECM$_2$) converge to their centralized versions with sufficient data transmissions per time frame.

9. Conclusions

In this paper, we proposed a framework for distributed estimation of the inverse correlation matrix in a randomly connected network. The proposed framework is based on the fact that using recursive exponential smoothing in combination with the Sherman–Morrison formula, the estimation of the inverse correlation matrix can be structured as two rounds of averaging consensus problems. We first investigated the use of gossip processing for distributed estimation of the inverse correlation matrix, since gossip processing is an well known approach for solving averaging consensus problems. However, due to the fact that the inverse correlation matrix is updated recursively across time, the convergence error between the gossip-based estimated correlation matrix and the centralized estimated correlation matrix accumulates across time. In addition, we therefore also proposed a clique-based distributed algorithm to eliminate this convergence error. This algorithm is referred to as clique-based distributed estimation of the inverse correlation matrix (CbDECM).

The proposed CbDECM is analyzed and compared with a centralized estimator in terms of transmission costs. The comparison is done in a scenario, where the source of interest of a user in the network is considered to be privacy preserving by hiding the information of the steering vectors. In such a privacy preserving scenario, each user in the network is assumed to know the steering vectors locally, and thus each user requires an estimate of the inverse correlation matrix.

Simulation results with the gossip-based estimation approach showed that the convergence error of the estimated inverse correlation matrix increases across time, while the CbDECM algorithm converges to the same estimate as the centralized estimator. Moreover, experiments on the comparisons between the proposed CbDECM algorithm and the distributed delay and sum beamformer in referenced literature illustrated the performance improvement of the proposed CbDECM algorithm by incorporating noise correlation. Compared with other distributed adaptive beamformers, the proposed distributed MVDR beamformer in this paper make use of prior knowledge on the steering vectors. For future research it will be interesting to investigate how to estimate the steering vectors in a distributed way while still preserving the users' privacy with respect to his source of interest.

References

[1] R.C. Hendriks, T. Gerkmann, J. Jensen, DFT-Domain Based Single-Microphone Noise Reduction for Speech Enhancement: A Survey of the State of the Art, Morgan & Claypool, 2013.
[2] J. Benesty, M.M. Sondhi, Y. Huang (Eds.), Springer Handbook of Speech Processing, Springer, Berlin, 2008.
[3] M. Brandstein, D. Ward (Eds.), Microphone Arrays: Signal Processing Techniques and Applications, Springer, New York, 2001.
[4] A. Bertrand, M. Moonen, Distributed node-specific LCMV beamforming in wireless sensor networks, IEEE Trans. Signal Process. 60 (1) (2012) 233–246.
[5] S. Markovich-Golan, S. Gannot, I. Cohen, Distributed GSC beamforming using the relative transfer function, in: EURASIP European Signal Processing Conference (EUSIPCO), 2012, pp. 1274–1278.
[6] S. Markovich-Golan, S. Gannot, I. Cohen, Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks, IEEE Trans. Audio Speech Lang. Process. 21 (2) (2013) 343–356.
[7] Y. Zeng, R.C. Hendriks, Distributed delay and sum beamformer for speech enhancement in wireless sensor networks via randomized gossip, in: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2012, pp. 4037–4040.
[8] R.C. Hendriks, Z. Erkin, T. Gerkmann, Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain, in: International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Canada, 2013, pp. 7005–7009.
[9] R.C. Hendriks, Z. Erkin, T. Gerkmann, Privacy preserving distributed beamforming based on homomorphic encryption, in: Proceeding of the European Signal Processing Conference, EUSIPCO , Marrakesh, Morocco, 2013, pp. 7005–7009.
[10] C. Fontaine, F. Galand, A survey of homomorphic encryption for nonspecialists, EURASIP J. Inf. Secur. 2007 (2007) 1–10.
[11] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah, Randomized gossip algorithms, IEEE Trans. Inf. Theory 52 (6) (2006) 2508–2530.
[12] R. Heusdens, G. Zhang, R. Hendriks, Y. Zeng, W. Kleijn, Distributed MVDR beamforming for (wireless) microphone networks using message passing, in: International Workshop on Acoustic Signal Enhancement (IWAENC), 2012, pp. 1–4.
[13] G. Zhang, R. Heusdens, Linear coordinate-descent message-passing for quadratic optimization, in: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2012, pp. 2005–2008.
[14] J. Sherman, W.J. Morrison, Adjustment of an inverse matrix corresponding to a change in one element of a given matrix, Ann. Math. Stat. 21 (1) (1950) 124–127.
[15] S. Gannot, D. Burshtein, E. Weinstein, Signal enhancement using beamforming and nonstationarity with applications to speech, IEEE Trans. Signal Process. 49 (8) (2001) 1614–1626.
[16] J. Liu, S. Mou, A.S. Morse, B.D. Anderson, C. Yu, Deterministic gossiping, Proc. IEEE 99 (9) (2011) 1505–1524.
[17] H.L. van Trees, Optimum Array Processing, Wiley, New York, 2002.

[18] R. Bovenkamp, F. Kuipers, P.V. Mieghem, Gossip-based counting in dynamic networks, Networking 2012 (2012) 404–417.
[19] Y. Zeng, R.C. Hendriks, R. Heusdens, Clique-based distributed beamforming for speech enhancement in wireless sensor networks, in: Proceedings of the European Signal Processing Conference, EUSIPCSO, Marrakesh, Morocco, 2013, pp. 1–5.
[20] H. Chen, A. Campbell, B. Thomas, A. Tamir, Minimax flow tree problems, Networks 54 (2009) 117–129.
[21] J.S. Garofolo, DARPA TIMIT Acoustic-Phonetic Speech Database, National Institute of Standards and Technology (NIST).
[22] R.C. Hendriks, R. Heusdens, J. Jensen, MMSE based noise PSD tracking with low complexity, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2010, pp. 4266–4269.
[23] C.H. Taal, R.C. Hendriks, R. Heusdens, J. Jensen, An algorithm for intelligibility prediction of time-frequency weighted noisy speech, IEEE Trans. Audio Speech Lang. Process. 19 (7) (2011) 2125–2136.