Distributed signal estimation in sensor networks where nodes have different interests

Alexander Bertrand, Marc Moonen

ESAT-SCD / IBBT Future Health Department
KU Leuven, University of Leuven
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
E-mail: alexander.bertrand@esat.kuleuven.be
marc.moonen@esat.kuleuven.be
Phone: +32 16 321899, Fax: +32 16 321970

Abstract—In this paper, we consider distributed signal estimation in sensor networks where the nodes exchange compressed sensor signal observations to estimate different node-specific signals. In particular, we revisit the so-called distributed adaptive node-specific signal estimation (DANSE) algorithm, which applies to the case where the nodes share a so-called ‘common interest’, and cast it in the more general setting where the nodes have ‘different interests’. We prove existence of an equilibrium state for such a setting by using a result from fixed point theory. By establishing a link between the DANSE algorithm and game theory, we point out that any equilibrium of the DANSE algorithm is a Nash equilibrium of the corresponding game. This provides an intuitive interpretation to the resulting signal estimators. The equilibrium state existence proof also reveals a problem with discontinuities in the DANSE update function, which may result in non-convergence of the algorithm. However, since these discontinuities are identifiable, they can easily be avoided by applying a minor heuristic modification to the algorithm. We demonstrate the effectiveness of this modification by means of numerical examples.

Index Terms—Wireless sensor networks (WSNs), distributed signal estimation, distributed adaptive beamforming, adaptive estimation, game theory

I. INTRODUCTION

In this paper, we consider distributed signal estimation in sensor networks where the nodes exchange compressed sensor signal observations to estimate different node-specific signals. In particular, we revisit the so-called distributed adaptive node-specific signal estimation (DANSE) algorithm, which operates in fully connected wireless sensor networks (WSNs) [1]–[4] or WSNs with a tree topology [5]. Each node acts as a data sink and fuses its local sensor signal observations with (compressed) signal observations obtained from other

nodes, based on a local linear minimum mean squared error (MMSE) estimator. The term ‘node-specific’ refers to the fact that each node in the network has a different objective, i.e., each node aims to obtain a local estimate of a desired signal, which is different for each node. If these node-specific desired signals share a common latent signal subspace, and if sufficient bandwidth is available, the behavior of the DANSE algorithm is well-understood. This is referred to here as the case where nodes share a ‘common interest’. For such settings, different versions of the algorithm have been proven to converge to an equilibrium state with different node-specific distributed estimators, and all of them achieve the same mean square error (MSE) as the corresponding optimal centralized estimator [1]–[3], [5]–[8].

In practice, the above assumption is often not (or only partially) satisfied, i.e., the nodes often have (partially) ‘different interests’. In this case, it is not clear whether there exists an equilibrium state for the DANSE algorithm, and if so, whether the corresponding distributed estimators are optimal in any sense. Experiments in [1], [3], [8] on different types of signals have indicated that the DANSE algorithm generally converges to an equilibrium state for such settings. In this paper, we prove that such an equilibrium state indeed exists, by using a result from fixed point theory. We also establish a link between the DANSE algorithm and game theory, which will reveal that any equilibrium state of the DANSE algorithm is a Nash equilibrium [9]. Nash equilibria are known to be suboptimal in general due to non-cooperative behavior of the players in the game.

The equilibrium state existence proof also reveals a problem with discontinuities in the DANSE update function. Although the probability that the DANSE algorithm encounters such discontinuities is zero, the actual equilibrium state may be close to such a discontinuity. If such ill-conditioned equilibria exist, the algorithm may not converge due to estimation errors in the signal statistics and/or numerical problems. However, since these ill-conditioned settings are identifiable, they can easily be avoided by applying a minor heuristic modification to the DANSE algorithm. We demonstrate the effectiveness of this modification by means of numerical examples.

Alexander Bertrand is supported by a Postdoctoral Fellowship of the Research Foundation - Flanders (FWO). This research work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Research Council CoE EF/05/006 Optimization in Engineering (OPTEC) and PFV/10/002 (OPTEC), 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), Research Project IBBT 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.

A. Relationship with previous work on distributed estimation

The literature on distributed estimation in WSNs is very versatile, and considers many different problem statements and data models (see, e.g., [10] for an overview and classifications). Some approaches require a so-called fusion center (e.g., [11]–[16]) that gathers data from all sensors, whereas other algorithms are fully distributed where all processing happens inside the network (e.g., [1]–[7], [17]–[24]). The estimation literature can also be classified into signal estimation (or beam-forming) and parameter estimation techniques [10]. Signal estimation focuses on the design of fusion rules to combine raw or pre-processed sensor signal observations, to estimate a hidden *signal*, e.g. for denoising [1]–[7], [14], [15], [21], [25], [26]. Note that a new estimation variable, i.e., a sample of the hidden signal, is then introduced in each sampling interval. Parameter estimation, on the other hand, refers to estimation and/or tracking of a parameter vector with fixed dimension, which is extracted from the sensor signal observations (e.g., by locally solving a linear regression problem [17]–[20], [24]). In this case, the number of estimation variables is fixed over time, i.e., it does not increase whenever new sensor observations are acquired. Although both terms (parameter and signal estimation) are often used interchangeably, they have very different problem statements, and need to be tackled in different ways [10]. In addition, there is also a large amount of related work on coding in WSNs, i.e., distributed compression or source coding [27]–[30], joint estimation and coding [4], [16], [26], [31], and compressed sensing [16], [32].

Following the above classification, the DANSE algorithm belongs to the class of signal estimation algorithms. This means that it aims to define a set of fusion rules (in this case a linear filter-and-sum estimator), exploiting correlation between signals in different sensors, to estimate a hidden desired signal, while reducing noise and interference. In contrast to, e.g., [14], [15], we consider a fully distributed approach without fusion center, i.e., all processing happens inside the network. Furthermore, each node acts as a data sink, i.e., each node aims to estimate a node-specific desired signal. Although we do not focus on compression or source coding, we explicitly limit the number of signals that can be shared between nodes, i.e., multi-sensor observations are fused to obtain a signal with fewer channels than the original number of sensors. The effect of additional (lossy) compression of the signals that are shared between nodes in the DANSE algorithm is analyzed in [4].

B. Outline

The outline of this paper is as follows. In section II, we state the problem of distributed node-specific signal estimation, and we briefly review the DANSE algorithm and its assumed data model ('common interest' case). In Section III, we explain that this data model is often not or only partially satisfied in practice ('different interest' case), we explain how this case can be viewed from a game theoretical perspective, and we show that any equilibrium state of the DANSE algorithm then corresponds to a Nash equilibrium. In section IV, we prove that such an equilibrium state always exists, except in some contrived cases where the DANSE algorithm updates to

discontinuous points of its update function. In section V, we give examples of scenarios where such discontinuities leads to convergence problems, and we explain how the algorithm can be modified to avoid these discontinuities or ill-conditioned settings. In Section VI, we provide simulations, demonstrating the stabilizing effect of the modified algorithm. Conclusions are drawn in Section VII.

II. DISTRIBUTED NODE-SPECIFIC SIGNAL ESTIMATION

In this section, we provide a brief review of distributed node-specific signal estimation and the DANSE algorithm in a fully connected sensor network. For a more detailed explanation and analysis, we refer to [1].

A. Problem formulation and notation

Consider a fully connected¹ sensor network with the set of nodes $\mathcal{N} = \{1, \dots, N\}$ containing N nodes. Node $n \in \mathcal{N}$ collects observations $\mathbf{y}_n[t]$ (at sample time t) of a complex-valued² M_n -channel signal $\mathbf{y}_n$ (the different channels of $\mathbf{y}_n$ usually correspond to different sensors at node n). For conciseness, we will omit the time index t in the sequel. We define $\mathbf{y}$ as the M -channel signal in which all $\mathbf{y}_n$ are stacked, where $M = \sum_{n \in \mathcal{N}} M_n$. The objective for node n is to estimate a K -channel node-specific signal $\mathbf{d}_n$, referred to as the desired signal³. It is noted that each node may have a different desired signal, such that possibly $\mathbf{d}_n \neq \mathbf{d}_q$ if $n \neq q$.

We first consider the centralized estimation problem, i.e., we assume that each node has access to observations of the entire M -channel signal $\mathbf{y}$. Node n uses a linear estimator $\mathbf{W}_n$ to estimate $\mathbf{d}_n$ as $\hat{\mathbf{d}}_n = \mathbf{W}_n^H \mathbf{y}$ where $\mathbf{W}_n$ is a complex valued $M \times K$ matrix, and where superscript H denotes the conjugate transpose operator. We consider linear MMSE estimation with a node-specific optimal estimator $\hat{\mathbf{W}}_n$ that minimizes the cost function

$$J_n(\mathbf{W}_n) = E\{\|\mathbf{d}_n - \mathbf{W}_n^H \mathbf{y}\|^2\}, \quad (1)$$

with $E\{\cdot\}$ denoting the expected value operator. Let $\mathbf{R}_{yy} = E\{\mathbf{y}\mathbf{y}^H\}$, which can be estimated by temporal averaging of observations of $\mathbf{y}$, and let $\mathbf{R}_{yd_n} = E\{\mathbf{y}\mathbf{d}_n^H\}$, which can be estimated directly (e.g., with training sequences) or indirectly (see, e.g., [1]). Assuming that the correlation matrix $\mathbf{R}_{yy}$ has full rank⁴, the unique minimizer of (1) is

$$\hat{\mathbf{W}}_n = \mathbf{R}_{yy}^{-1} \mathbf{R}_{yd_n}. \quad (2)$$

B. The DANSE_K algorithm

For any integer K , the DANSE_K algorithm linearly compresses $\mathbf{y}_n$ to a K -channel signal $\mathbf{z}_n = \mathbf{C}_n^H \mathbf{y}_n$, where $\mathbf{C}_n$ is

¹For the sake of an easy exposition, the DANSE algorithm is explained here for the case of fully connected WSN's. However, it can also be formulated for tree topology networks [5].

²Throughout this paper, all signals are assumed to be complex valued to permit frequency-domain descriptions, e.g., in the short-time Fourier transform (STFT) domain.

³ $\mathbf{d}_n$ is assumed to be a K -channel signal for each node $n \in \mathcal{N}$, which may require the inclusion of auxiliary signals, see [1].

⁴This is usually satisfied in practice due to uncorrelated sensor noise. If the matrix is not full rank, the minimum norm solution can be selected, based on the Moore-Penrose pseudoinverse [33].

a $K \times M_n$ compression matrix⁵ that will be defined later (see formula (8)). Observations of the compressed signal $\mathbf{z}_n$ are then broadcast to the remaining $N - 1$ nodes. This compresses the data to be sent by node n by a factor of $\frac{M_n}{K}$. If the desired signals $\mathbf{d}_n$ all share a common Q -dimensional latent signal subspace, we will point out in section II-C that the DANSE_K algorithm achieves the optimal estimators whenever $K \geq Q$.

Since the compression matrices $\mathbf{C}_n$ and the local estimators will be iteratively updated by the DANSE_K algorithm, we introduce the iteration index i in the sequel. We define the $(N - 1)K$ -channel signal $\mathbf{z}_{-n}^i = [\mathbf{z}_1^{iT} \dots \mathbf{z}_{n-1}^{iT} \mathbf{z}_{n+1}^{iT} \dots \mathbf{z}_N^{iT}]^T$, and a partitioning of the estimator matrix $\mathbf{W}_n^i$ as $\mathbf{W}_n^i = [\mathbf{W}_{n1}^{iT} \dots \mathbf{W}_{nN}^{iT}]^T$, where $\mathbf{W}_{nq}^i$ is the part of $\mathbf{W}_n^i$ that is applied to $\mathbf{y}_q$. Node n then collects observations of its own signal $\mathbf{y}_n$ and the signals in $\mathbf{z}_{-n}^i$ obtained from the other nodes in the network, and uses these to update its local linear MMSE estimator, i.e.,

$$\begin{aligned} \begin{bmatrix} \mathbf{W}_{nn}^{i+1} \\ \mathbf{G}_{n,-n}^{i+1} \end{bmatrix} = \\ \arg \min_{\mathbf{W}_{nn}, \mathbf{G}_{n,-n}} E \left\{ \left\| \mathbf{d}_n - \begin{bmatrix} \mathbf{W}_{nn}^H & \mathbf{G}_{n,-n}^H \end{bmatrix} \begin{bmatrix} \mathbf{y}_n \\ \mathbf{z}_{-n}^i \end{bmatrix} \right\|^2 \right\}, \end{aligned} \quad (3)$$

where $\mathbf{W}_{nn}$ is the part of the estimator that is applied to $\mathbf{y}_n$, and where $\mathbf{G}_{n,-n} = [\mathbf{G}_{n,1}^T \dots \mathbf{G}_{n,n-1}^T \mathbf{G}_{n,n+1}^T \dots \mathbf{G}_{n,N}^T]^T$, with $\mathbf{G}_{nq}$ denoting the part of the estimator that is applied to $\mathbf{z}_q^i$. Let

$$\tilde{\mathbf{y}}_n^i = \begin{bmatrix} \mathbf{y}_n \\ \mathbf{z}_{-n}^i \end{bmatrix}. \quad (4)$$

The solution of (3) is then given by

$$\begin{bmatrix} \mathbf{W}_{nn}^{i+1} \\ \mathbf{G}_{n,-n}^{i+1} \end{bmatrix} = (\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i)^{-1} \mathbf{R}_{\tilde{\mathbf{y}}_n \mathbf{d}_n}^i, \quad (5)$$

with

$$\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i = E\{\tilde{\mathbf{y}}_n^i \tilde{\mathbf{y}}_n^{iH}\} \quad (6)$$

$$\mathbf{R}_{\tilde{\mathbf{y}}_n \mathbf{d}_n}^i = E\{\tilde{\mathbf{y}}_n^i \mathbf{d}_n^H\}. \quad (7)$$

The compression rule to generate the signal $\mathbf{z}_n^i$ that is broadcast by node n at iteration i is defined by the linear operator

$$\mathbf{z}_n^i = \mathbf{W}_{nn}^{iH} \mathbf{y}_n. \quad (8)$$

The DANSE algorithm is illustrated in Fig. 1, for a network with $N = 3$ nodes. It is noted that $\mathbf{W}_{nn}$ both acts as a compressor and as a part of the estimator $\mathbf{W}_n$. From Fig. 1, it is easy to see that the parametrization of the $\mathbf{W}_n$ effectively applied at node n is

$$\tilde{\mathbf{W}}_n = \begin{bmatrix} \mathbf{W}_{11} \mathbf{G}_{n1} \\ \vdots \\ \mathbf{W}_{NN} \mathbf{G}_{nN} \end{bmatrix}, \quad (9)$$

where $\mathbf{G}_{nn} = \mathbf{I}_K$, with $\mathbf{I}_K$ denoting the $K \times K$ identity matrix. Expression (9) defines a solution space for all $\mathbf{W}_n$, $n \in \mathcal{N}$, simultaneously, where node n can only control the

⁵If $M_n \leq K$, then $\mathbf{C}_n$ is chosen as the $M_n \times M_n$ identity matrix instead, in which case no compression is obtained.

Fig. 1. The DANSE_K algorithm with 3 nodes ($|K| = 3$). Each node k estimates a signal $\mathbf{d}_k$ using observations of its own M_k -channel sensor signal, and 2 K -channel signals broadcast by the other two nodes.

parameters $\mathbf{W}_{nn}$ and $\mathbf{G}_{n,-n}$. We use a tilde to indicate that the estimator is parametrized according to (9).

The DANSE_K algorithm iteratively updates the parameters in (9), by letting each node n compute (5), $\forall n \in \mathcal{N}$, in a sequential round robin fashion. It is noted that each node then essentially performs a similar task as in a centralized computation, but on a smaller scale, i.e., with fewer signals. The complete DANSE_K algorithm is given in Table I.

Remark I: The iterative nature of the DANSE_K algorithm may suggest that the same sensor signal observations are compressed and broadcast multiple times, i.e., once after every iteration. However, in practical applications, iterations are spread over time, which means that successive updates of the estimators use different sensor signal observations. By exploiting the stationarity assumption, updated estimators are only used for new (future) observations (as it is the case in adaptive filtering or adaptive beamforming [34], [35]). In other words, if $\mathbf{W}_{nn}^i$ is updated to $\mathbf{W}_{nn}^{i+1}$ at time t_0 , this is only used to produce observations of $\mathbf{z}_n^i$ for $t > t_0$, while previous observations for $t \leq t_0$, are neither recompressed nor retransmitted. Effectively, each sensor signal observation is compressed and transmitted only once. For a detailed non-batch description of the algorithm, we refer to [1].

Remark II: In the DANSE_K algorithm, the nodes update in a sequential round robin fashion, i.e., at each iteration i , a different node updates its local parameters. Alternative procedures where nodes update simultaneously are provided in [2].

C. Convergence and optimality of the DANSE_K algorithm for nodes with a common interest

Assume that the node-specific desired signals of all the nodes share a common latent signal subspace of dimension Q , i.e.,

$$\mathbf{d}_n = \mathbf{A}_n \mathbf{d}, \quad \forall n \in \mathcal{N}, \quad (13)$$

TABLE I
THE DANSE_K ALGORITHM

The DANSE_K Algorithm

- 1) • Initialize $\mathbf{W}_{qq}^0$ and $\mathbf{G}_{q,-q}^0$, $\forall q \in \mathcal{N}$, as random matrices.
• $i \leftarrow 0$.
• $n \leftarrow 1$.
- 2) • All nodes $q \in \mathcal{N}$ transmit observations of $\mathbf{z}_q^i = \mathbf{W}_{qq}^{iH} \mathbf{y}_q$ to the other nodes.
• Node n updates its local parameters $\mathbf{W}_{nn}^i$ and $\mathbf{G}_{n,-n}^i$ by minimizing its local MSE criterion, based on observations of its own inputs sensor signal $\mathbf{y}_n$ and of the compressed signals $\mathbf{z}_{-n}^i$ that it receives from nodes $q \in \mathcal{N} \setminus \{n\}$:

$$\begin{bmatrix} \mathbf{W}_{nn}^{i+1} \\ \mathbf{G}_{n,-n}^{i+1} \end{bmatrix} = (\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i)^{-1} \mathbf{R}_{\tilde{\mathbf{y}}_n \mathbf{d}_n}^i. \quad (10)$$

If $\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i$ is rank deficient, the Moore-Penrose pseudoinverse [33] is used instead.

- All other nodes do not change their variables:

$$\forall q \in \mathcal{N} \setminus \{n\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i, \mathbf{G}_{q,-q}^{i+1} = \mathbf{G}_{q,-q}^i. \quad (11)$$

- 3) For the newly observed samples, each node $q \in \mathcal{N}$ generates an estimate of its desired signal:

$$\bar{\mathbf{d}}_q = \mathbf{W}_{qq}^{i+1H} \mathbf{y}_q + \mathbf{G}_{q,-q}^{i+1H} \mathbf{z}_{-q}^i. \quad (12)$$

- 4) • $i \leftarrow i + 1$.
• $n \leftarrow (n \bmod N) + 1$.
- 5) Return to step 2

with $\mathbf{A}_n$ a fixed (but unknown) $K \times Q$ matrix and where $E\{\mathbf{d}\mathbf{d}^H\}$ is a full-rank matrix. This is referred to as the case where nodes have a common interest, and represents the assumed data model for the DANSE_K algorithm in [1]–[3], [5]–[8], [25]. In [1], it is explained that the optimal centralized estimators (2) are in the solution space of the DANSE_K algorithm, as defined in (9), if (13) is satisfied with $K \geq Q$ and if each $\mathbf{A}_n$ matrix has rank Q . The following theorem guarantees convergence of the DANSE_K algorithm to the optimal centralized estimators in the case⁶ where $K = Q$ [1]:

Theorem II.1. *If the sensor signal correlation matrix $\mathbf{R}_{yy}$ has full rank, and if there exists a latent K -channel signal $\mathbf{d}$ such that $\mathbf{d}_n = \mathbf{A}_n \mathbf{d}$, $\forall n \in \mathcal{N}$, with $\mathbf{A}_n$ a full rank $K \times K$ matrix, then the DANSE_K algorithm converges for any initialization of its parameters to the optimal centralized estimators (2) for all $n \in \mathcal{N}$.*

III. DANSE_K ALGORITHM FOR NODES WITH DIFFERENT INTERESTS

Convergence and optimality of the DANSE_K algorithm is obtained under data model (13) with $K = Q$. However, this assumption may not be satisfied, for example when $K < Q$ due to underestimation of Q (or upper bounds on K due to bandwidth constraints), or when there is no correlation between the different node-specific desired signals to begin with. Furthermore, even in cases where K and Q match, the model may only be approximately satisfied. A common

example is the case of convolutive mixtures. Here, the data model is satisfied when evaluated in the frequency domain, since convolutive mixtures are transformed into instantaneous mixtures at each frequency. However, since finite discrete Fourier transforms (DFT) are used in practice, there is inevitable frequency leakage between frequency bins. The data model is then violated in each frequency bin, especially so for small DFT sizes.

In such cases where the desired signals $\mathbf{d}_n$ do not fully capture the Q -dimensional latent signal subspace ($K < Q$ or $\mathbf{A}_n$ is not full rank), the optimal centralized estimators (2) are not in the solution space of DANSE_K as defined by (9), and so they cannot be reached. Still, it is observed in extensive simulations that the DANSE_K algorithm mostly converges to an equilibrium state in these cases⁷. Insights on the existence and interpretation of this equilibrium can be found in game theory, more specifically in the notion of the Nash equilibrium [9].

A. Game theory and Nash equilibria

Consider a game between N players, where the goal for player n is to maximize a certain utility- or payoff function $u_n(\mathbf{s}_1, \dots, \mathbf{s}_N)$, and where $\mathcal{N}$ is the set of players. Player n can choose a specific strategy vector $\mathbf{s}_n$ from its strategy set $\mathcal{S}_n$. Notice that the value of the utility u_n of player n depends on its own strategy $\mathbf{s}_n$ and the strategies chosen by the other $(N - 1)$ players. Let $\mathbf{s} = [\mathbf{s}_1^T, \dots, \mathbf{s}_N^T]^T$ denote

⁶We only address the case where $K = Q$ and where $\mathbf{R}_{yy}$ has full rank. If $K > Q$, the DANSE_K algorithm still converges to the optimal centralized estimators, but this is based on simulations, without a formal proof [1].

⁷This is stated here as an observation, since a formal convergence proof with necessary conditions is not available. The statement is even not true in general, since there exist some contrived examples for which the DANSE_K algorithm cannot converge (see Appendix).

the joint strategy, i.e., the stacked vector of all strategies, and let $\mathcal{S}$ denote the set of all possible joint strategies, i.e., $\mathcal{S}$ is the Cartesian product of the strategy sets of all players, $\mathcal{S} = \times_{n \in \mathcal{N}} \mathcal{S}_n$. Let $\mathbf{s}_{-n}$ denote the joint strategy $\mathbf{s}$ with the strategy of player n removed.

Definition III.1 (Nash equilibrium). *Strategy $\mathbf{s} \in \mathcal{S}$ is a Nash equilibrium if $u_n(\mathbf{s}) \geq u_n(\hat{\mathbf{s}}_n, \mathbf{s}_{-n}) \forall \hat{\mathbf{s}}_n \in \mathcal{S}_n, \forall n \in \mathcal{N}$.*

In other words, the Nash equilibrium describes a setting where none of the players can improve its utility function, given the current strategies chosen by the other players. Therefore, the definition of a Nash equilibrium can be reformulated by use of the best-reply correspondence [36]:

Definition III.2 (best-reply correspondence). *The best-reply correspondence for player n is a point-to-set mapping that maps each strategy $\mathbf{s} \in \mathcal{S}$ to a set $\mathcal{B}_n(\mathbf{s}) \subseteq \mathcal{S}_n$ that satisfies $\mathcal{B}_n(\mathbf{s}) = \{\arg \max_{\hat{\mathbf{s}}_n \in \mathcal{S}_n} u_n(\hat{\mathbf{s}}_n, \mathbf{s}_{-n})\}$. The best-reply set for the game is then defined as $\mathcal{B}(\mathbf{s}) = \times_{n \in \mathcal{N}} \mathcal{B}_n(\mathbf{s})$.*

Now we can reformulate the definition of Nash equilibrium:

Corollary III.3. *Strategy $\mathbf{s} \in \mathcal{S}$ is a Nash equilibrium if and only if $\mathbf{s} \in \mathcal{B}(\mathbf{s})$.*

Nash equilibria are known to be suboptimal in general, i.e., there often exist other joint strategies that yield a higher utility for all or the majority of the players (or a higher total utility, when summed over all players). In game theory this is often referred to as ‘the price of anarchy’ [37], which is a measure that indicates the loss in utility due to non-cooperative behavior of the players.

B. DANSE_K algorithm in a game-theoretic framework

It is not difficult to see the correspondence between the DANSE_K algorithm and a game as described above. Indeed, the set of nodes corresponds to the set of players, and the utility function u_n that player n aims to maximize corresponds to the MSE cost function, i.e.,

$$u_n = -\tilde{J}_n(\mathbf{W}, \mathbf{G}_{n,-n}) = -J_n(\tilde{\mathbf{W}}_n), \quad (14)$$

where $\mathbf{W} = [\mathbf{W}_{11}^T \mathbf{W}_{22}^T \dots \mathbf{W}_{NN}^T]^T$ and where $\tilde{\mathbf{W}}_n$ is defined by (9). The estimators $\{\mathbf{W}_{nn}, \mathbf{G}_{n,-n}\}$ that are used by node n correspond to the strategy vector $\mathbf{s}_n$ of player n .

Theorem III.4. *Every equilibrium state of the DANSE_K algorithm is a Nash equilibrium of the game as described above.*

Proof: One iteration in the DANSE_K algorithm, where node n updates its local estimators, is a best-reply correspondence to the current estimators/strategies applied by the other nodes/players. Because of Corollary III.3, any equilibrium state of the DANSE_K algorithm must be a Nash equilibrium. $\square$

The above correspondence between game theory and the DANSE_K algorithm, and the fact that its equilibrium states correspond to Nash equilibria, explains that the equilibrium states are generally suboptimal, i.e., other distributed estimators may exist that are also parametrized by (9) and that

improve the overall estimation performance. This is of course due to the implicit non-cooperative behavior of the nodes. Indeed, the update rule based on (3) makes nodes act selfishly, i.e., each node updates its parameters to minimize its own cost function, ignoring possible increases in the cost of other nodes due to this update.

The proof in [1] of Theorem II.1 shows that the DANSE_K algorithm is actually behaving cooperatively when the assumptions of the theorem are satisfied, despite the non-cooperative update rule. This implicit cooperative behavior is due to the fact that all nodes have $\mathbf{d}_n$'s that are linear combinations of a common $\mathbf{d}$. In other words, there is a common latent interest between the different nodes in the network, which in this case reduces the ‘price of anarchy’ to zero.

IV. EXISTENCE OF AN EQUILIBRIUM STATE

In Subsection IV-A, we first prove that there always exists an equilibrium state for the DANSE_K algorithm⁸, i.e., a state that is invariant under the DANSE_K update rules, even when $K < Q$ in (13). Furthermore, the proof will reveal a problem with discontinuities in the DANSE_K update function. In Section V, we will explain the practical consequences of these discontinuities, and how their identification can be used to increase the robustness of the DANSE_K algorithm in ill-conditioned scenarios.

A. Proof of existence

From Corollary III.3, we know that a Nash equilibrium satisfies $\mathbf{s} \in \mathcal{B}(\mathbf{s})$. In fixed point theory, this means that $\mathbf{s}$ is a so-called fixed point of the point-to-set function $\mathcal{B}(\mathbf{s})$. Indeed, Nash used Kakutani's fixed point theorem to prove existence of Nash equilibria in strategic form games with convex strategy spaces [9]. However, the game-theoretic framework used by Nash is probabilistic, i.e., it uses mixed strategies which are probability distributions over a set of pure strategies. The DANSE_K algorithm, however, is deterministic, and the function $\mathcal{B}(\mathbf{s})$ is a point-to-point function (given by (5), when applied to all $n \in \mathcal{N}$ simultaneously) instead of a point-to-set function. This allows us to use another (simpler) fixed point theorem:

Theorem IV.1 (Brouwer's fixed point theorem [38]). *Every continuous function F from the closed unit ball D^m to itself has at least one fixed point.*

The closed unit ball D^m is the set of all points in Euclidean m -space that are at a maximum distance of 1 from the origin. A fixed point of a function $F : D^m \rightarrow D^m$ is a point $\mathbf{x}$ in D^m such that $F(\mathbf{x}) = \mathbf{x}$. Because the properties involved (‘continuity’, ‘being a fixed point’) are invariant under homeomorphisms, the theorem equally applies if the domain is not the closed unit ball itself but some set homeomorphic to it. Since a compact (i.e., bounded and closed) convex set is

⁸We only prove existence of an equilibrium state. Uniqueness and stability of the equilibrium, as well as convergence to this equilibrium, are still open questions for the general case. However, extensive simulations appear to confirm that these properties indeed hold, except in some contrived examples (see Appendix).

homeomorphic to the closed unit ball, this yields the following slightly more general theorem [38]:

Theorem IV.2. *Every continuous function F from a compact convex subset $\mathcal{C}$ of a Euclidean space to itself has at least one fixed point.*

We will use this theorem to prove the existence of an equilibrium state for the DANSE_K algorithm. However, we will first make a slight modification to the DANSE_K algorithm. The actual update formula (5) solves the local optimization problem given by (3). In what follows, we replace the unconstrained optimization problem (3) with the following constrained optimization problem

$$\begin{aligned} \left[\begin{array}{c} \mathbf{W}_{nn}^{i+1} \\ \vdots \\ \mathbf{G}_{n,-n}^{i+1} \end{array} \right] = \\ \arg \min_{\mathbf{W}_{nn}, \mathbf{G}_{n,-n}} E \left\{ \left\| \mathbf{d}_n - \left[\begin{array}{c} \mathbf{W}_{nn}^H \\ \vdots \\ \mathbf{G}_{n,-n}^H \end{array} \right] \left[\begin{array}{c} \mathbf{y}_n \\ \vdots \\ \mathbf{z}_{-n} \end{array} \right] \right\|^2 \right\}, \\ \text{s.t. } \|\mathbf{G}_{n,-n}\| \leq T, \end{aligned} \quad (15)$$

where T is a pre-defined positive number, i.e., we impose a norm constraint on the $\mathbf{G}_{n,-n}$ matrix (this can be any matrix norm). We do this to obtain a continuous update function to be able to use Theorem IV.2, as will be explained later. However, it is noted that this constrained optimization problem is merely introduced for theoretical completeness of the below theorem, to remove some mathematical artifacts. In practice, the DANSE_K algorithm always uses the unconstrained optimization problem given by (3), so we do not elaborate on solving (15). If (15) has multiple solutions, it is again assumed that the minimum norm solution is selected.

Theorem IV.3 (Existence of an equilibrium state for the DANSE_K algorithm). *Assume that the update formula (3) of node n in the DANSE_K algorithm is replaced by (15) with $0 < T < \infty$. If the correlation matrix $\mathbf{R}_{yy}$ has full rank, then there exists an equilibrium state for the DANSE_K algorithm, i.e., a state which is invariant under the DANSE_K update rules.*

Proof: Let F_n denote the function that performs the DANSE_K update of the elements in $\mathbf{W}_{nn}$, i.e., $\mathbf{W}_{nn}^{i+1} = F_n(\mathbf{W}^i)$, where $\mathbf{W}_{nn}^{i+1}$ satisfies (15). Notice that the update of $\mathbf{W}_{nn}$ only depends on $\mathbf{W}$ (through $\mathbf{z}_{-n}$), and is independent of $\mathbf{G}_{q,-q}$, $\forall q \in \mathcal{N}$. The point $\mathbf{W}^* = [\mathbf{W}_{11}^{*T} \dots \mathbf{W}_{NN}^{*T}]^T$ is an equilibrium state for the DANSE_K algorithm if and only if $F_n(\mathbf{W}^*) = \mathbf{W}_{nn}^*$, $\forall n \in \mathcal{N}$. In other words, $\mathbf{W}^*$ is an equilibrium state if $\mathbf{W}^*$ is a fixed point of the function

$$F(\mathbf{W}) = \left[\begin{array}{c} F_1(\mathbf{W}) \\ \vdots \\ F_N(\mathbf{W}) \end{array} \right]. \quad (16)$$

We will prove existence of a fixed point of the function F , by applying Theorem IV.2. Therefore, we have to prove that the function F is continuous over a compact convex set $\mathcal{C}$, and that it maps $\mathcal{C}$ into itself.

1) Continuity of F

We first explain that the discontinuities of the function F

can be removed by choosing $T < \infty$. Assume that $T = \infty$ in (15), i.e., that there is no norm constraint on $\mathbf{G}_{n,-n}$, and therefore the function F_n , $\forall n \in \mathcal{N}$, is defined by the update (5). Function $F(\mathbf{W})$ is then continuous in all points $\mathbf{W}$, except for points $\mathbf{W}$ for which there exists a $n \in \mathcal{N}$ such that $\mathbf{W}_{nn}$ is rank deficient. To see this, assume that in the i -th iteration we have a point $\mathbf{W}^i$ for which $\mathbf{W}_{nn}^i$ is full rank $\forall n \in \mathcal{N}$. Since each $\mathbf{W}_{nn}^i$ is full rank, the matrix $\mathbf{R}_{\tilde{y}_n \tilde{y}_n}^i$ is also full rank, since it is a compressed version of $\mathbf{R}_{yy}$ based on linearly independent combinations. Continuity of F then immediately follows from continuity of the inverse of full-rank matrices (see (5)), i.e., the update $F(\mathbf{W}^i)$ changes smoothly for small perturbations in $\mathbf{W}^i$. However, if there exists a node $q \in \mathcal{N}$ for which $\mathbf{W}_{qq}^i$ is rank deficient, $\mathbf{z}_q$ does not span an M_n -dimensional signal subspace, and the matrices $\mathbf{R}_{\tilde{y}_n \tilde{y}_n}^i$ for $n \neq q$ then become rank deficient so that F_n , and hence F , is discontinuous.

It is noted that the update function F_n is invariant under a non-zero scaling of its argument, i.e., $F_n(\mathbf{W}^i) = F_n(\delta \mathbf{W}^i)$, $\forall \delta \neq 0$, since a scaling of $\mathbf{W}^i$ with δ is implicitly compensated inside F_n by a scaling of $\mathbf{G}_{n,-n}$ with $1/\delta$. Due to this inverse scaling, the elements of $\mathbf{G}_{n,-n}$ grow infinitely large when δ approaches zero, i.e., $\|\mathbf{G}_{n,-n}^{i+1}\| \rightarrow \infty$ when $\delta \rightarrow 0$. However, if δ is exactly equal to zero, this means that all $\delta \mathbf{W}_{nn}^i$ are zero (i.e., rank deficient) and that there is no communication between nodes, i.e., $\mathbf{z}_n = 0$, $\forall n \in \mathcal{N}$, and nodes can then only rely on their own sensor signal observations. In other words:

$$F_n(\delta \mathbf{W}^i) = \begin{cases} \mathbf{W}_{nn}^{i+1} & \text{if } \delta \neq 0 \\ \mathbf{W}_{nn}^{\text{local}} & \text{if } \delta = 0 \end{cases}, \quad (17)$$

with $\mathbf{W}_{nn}^{\text{local}} \neq \mathbf{W}_{nn}^{i+1}$, which means that a discontinuity indeed appears at $\mathbf{W}^i = 0$. Similarly, such a discontinuity appears at any point $\mathbf{W}$ for which there exists a $q \in \mathcal{N}$ such that $\mathbf{W}_{qq}$ is rank deficient. Indeed, in this case, an infinitesimal perturbation on $\mathbf{W}_{qq}$ can generate a new infinitesimally small linearly independent signal component in $\mathbf{z}_q$ that can be exploited by node n if it makes $\mathbf{G}_{nq}$ infinitely large.

We have explained that $\|\mathbf{G}_{nq}^{i+1}\| \rightarrow \infty$ if $\mathbf{W}_{qq}^i$ gets closer to rank deficiency. Therefore, adding a norm constraint⁹ $\|\mathbf{G}_{n,-n}^{i+1}\| \leq T < \infty$, $\forall n \in \mathcal{N}$, as defined in (15), will remove these discontinuities from the update function F . Indeed, when evaluating F in a point $\mathbf{W}$ which contains a rank deficient $\mathbf{W}_{nn}$, an infinitesimal perturbation $\Delta \mathbf{W}$ on $\mathbf{W}$ will also result in an infinitesimal change in the update, i.e., $\|F(\mathbf{W}) - F(\mathbf{W} + \Delta \mathbf{W})\|$ is infinitesimally small. This is because the $\mathbf{G}_{nq}$'s are bounded, and hence cannot generate a (finite) contribution from the new linearly independent (but infinitesimally small) signal component. Therefore, F will have a smooth transition in the neighborhood of rank-deficient points.

2) Mapping of a convex compact set $\mathcal{C}$ into itself

⁹It is noted that the discontinuities can also be removed by adding a fixed regularization term to (5), e.g., by inverting $\mathbf{R}_{\tilde{y}_n \tilde{y}_n}^i + \epsilon \mathbf{I}$ instead of $\mathbf{R}_{\tilde{y}_n \tilde{y}_n}^i$, where ϵ is a small positive number and $\mathbf{I}$ the identity matrix with appropriate dimensions. However, this will always have an influence on the dynamics and the equilibria of the DANSE_K algorithm. Adding the norm constraint as defined in (15) is less intrusive, as it only has an influence on the algorithm if the constraint becomes active.

We choose an arbitrary node $n \in \mathcal{N}$. Define the sub-level set $\mathcal{C}_n$ as

$$\mathcal{C}_n = \{\mathbf{W}_n : J_n(\mathbf{W}_n) \leq J_n^0\}, \quad (18)$$

with

$$J_n^0 = J_n(\mathbf{O}_{M \times K}), \quad (19)$$

and where $\mathbf{O}_{M \times K}$ denotes an all-zero $M \times K$ matrix. It is noted that, due to the implicit optimization (3) in the DANSE_K algorithm,

$$\widetilde{\mathbf{W}}_n^i \in \mathcal{C}_n, \forall i \in \mathcal{I}, \quad (20)$$

where $\widetilde{\mathbf{W}}_n^i$ is defined similar to (9), and where $\mathcal{I}_n$ denotes the set of iteration indices corresponding to iterations where node n updates its parameters. Since J_n is a convex quadratic function, its sub-level sets are compact convex sets, and therefore $\mathcal{C}_n$ is a compact convex set. We define the set $\mathcal{C}_{nn}$ as

$$\mathcal{C}_{nn} = \{\mathbf{W}_{nn} | \exists \mathbf{W}_n \in \mathcal{C}_n : \mathbf{W}_{nn} = \mathbf{E}_n \mathbf{W}_n\}, \quad (21)$$

with

$$\mathbf{E}_n = \left[\mathbf{O}_{M_n \times \sum_{l=1}^{n-1} M_l} \mid \mathbf{I}_{M_n} \mid \mathbf{O}_{M_n \times \sum_{l=n+1}^N M_l} \right], \quad (22)$$

i.e., $\mathcal{C}_{nn}$ is the projection of $\mathcal{C}_n$ onto the coordinates corresponding to $\mathbf{W}_{nn}$. Therefore, $\mathcal{C}_{nn}$ is also a compact convex set since compactness and convexity are invariant under affine transformations. Notice that, due to (20), and the fact that $\mathbf{G}_{nn} = \mathbf{I}_K$,

$$F_n(\mathbf{W}) \in \mathcal{C}_{nn}, \forall \mathbf{W}. \quad (23)$$

Now consider the Cartesian product set $\mathcal{C} = \times_{n \in \mathcal{N}} \mathcal{C}_{nn}$, which is again a compact convex set. Because of (23), we find that the function $F(\mathbf{W})$ maps the complete space to the set $\mathcal{C}$. Therefore, the set $\mathcal{C}$ is mapped into itself. $\square$

Remark: Since the update function F , which is used in the above proof, is the same for the DANSE_K algorithm with sequential node updating [1] and for the DANSE_K algorithm with simultaneous node updating [2], the theorem also holds for the latter. Furthermore, similar results can be obtained for the tree-DANSE algorithm [5], which operates in sensor networks with a tree topology.

V. CONVERGENCE

Since T in the constrained optimization problem (15) can be chosen arbitrary large, the norm constraint can be omitted in practice, i.e., nodes can still solve unconstrained MMSE problems as specified in (3). On the other hand, this norm constraint is required in the existence proof of an equilibrium state for the DANSE_K algorithm. The question is then whether the inclusion of the constraint is purely theoretical, or truly has implications in practice?

A. Convergence problems due to discontinuities

In practical cases, it is observed that the discontinuous points described in the proof of Theorem IV.3 are never chosen by the DANSE_K algorithm, and that the algorithm iterates away from these points when used as an initial setting. In these cases, the norm constraint never becomes active and hence it

has no impact on the algorithm (if T is chosen sufficiently large). Therefore, in practical cases, the equilibrium state exists even when the norm constraint is omitted.

However, it is possible to construct contrived examples where the DANSE_K algorithm indeed chooses discontinuous points. These examples sometimes do not have an equilibrium state, and the DANSE_K algorithm with (3) then gets stuck in a limit cycle instead. To display the underlying mechanism for this limit cycle behaviour, we provide a detailed example in the Appendix. If there is no equilibrium state, an artificial equilibrium state is generated by the additional norm constraint on the $\mathbf{G}_{n,-n}$'s, which depends on the choice of T . This artificial equilibrium state is usually unstable, which means that the norm constraint does not change the fact that the algorithm does not converge.

It is possible to construct many similar examples, where the limit cycle behavior always has the same cause, namely that a $\mathbf{W}_{nn}^i$ becomes rank deficient. Although these examples are contrived and artificial, they should not be treated as merely mathematical artifacts or curiosities. Even though in practice there is a zero probability that a $\mathbf{W}_{nn}^i$ actually becomes rank deficient, it may indeed become ill-conditioned. In the proof of Theorem IV.3, we have explained that the norm of $\mathbf{G}_{qn}$, $\forall q \in \mathcal{N}$, may become very large when the partial estimator $\mathbf{W}_{nn}$ gets close to rank deficiency, i.e., when it is ill-conditioned. If the equilibrium state contains such an ill-conditioned $\mathbf{W}_{nn}$, inevitable small perturbations due to numerical noise and estimation errors in the updates of $\mathbf{W}_{nn}$ will be amplified by those large $\mathbf{G}_{qn}$'s in the other nodes $q \neq n$. Due to this amplification, this noise will have a significant effect and it will ripple through the entire network, affecting the updates in all the nodes, and hence affecting the dynamics of the DANSE_K algorithm. As a result, the DANSE_K algorithm will fail to converge.

The above mentioned ill-conditioned equilibrium states can indeed occur in practice. One possibility is when the channels of $\mathbf{d}_n$ (at a particular node n) are highly correlated, and therefore the centralized MMSE solution $\widetilde{\mathbf{W}}_n$ (and its distributed version) will be close to rank deficiency. Low SNR nodes can also cause ill-conditioned estimators. Indeed, when the desired signal of node n is not or only weakly observed by the sensors at node n itself, its sensor signals often become useless for its local estimation problem (although they can be useful for other nodes that estimate a different desired signal). Node n will then set the corresponding column in $\mathbf{W}_{nn}$ close to zero. The fact that these two scenarios cause problems was already addressed in [3], but the actual underlying problem, i.e., the ill-conditioned nature of the equilibrium state, was not identified.

B. Avoiding discontinuities

In the previous subsection it is concluded that, although the theoretical discontinuities are encountered with zero probability in practical scenarios, it is possible to have an equilibrium state that is close to one of these discontinuities. In this case, the algorithm may fail to converge. In [3], a more robust version of the DANSE algorithm has been introduced, referred

to as the robust-DANSE or R-DANSE algorithm, which does not affect the optimality of the equilibrium state. However, the R-DANSE algorithm is quite intrusive since it changes the desired signals at the different nodes to create a scenario that is better conditioned.

In practice, it is often undesired or impossible to change the desired signals in certain nodes. Since we have identified the underlying problem, i.e., the discontinuities in the update function, we can enforce convergence by applying a heuristic safety measure¹⁰ instead of applying the R-DANSE algorithm. This is described in Table II, and we refer to this modified algorithm as the DANSE_K^* algorithm.

In the DANSE_K^* algorithm, the estimator $\mathbf{W}_{nn}$ at node n is not necessarily equal to the compression matrix $\mathbf{C}_n$ anymore (as it was the case in the DANSE_K algorithm). The DANSE_K^* algorithm freezes the coefficients of the compression matrix $\mathbf{C}_n$ when the estimator $\mathbf{W}_{nn}^{i+1}$ comes close to rank deficiency. Basically, this means that the dynamics of node n are eliminated from the network-wide problem, such that the remaining nodes can obtain an equilibrium. In this case, the algorithm will behave nicely, and it will converge to an equilibrium state, as will be demonstrated in the next section.

VI. SIMULATION RESULTS

A. Experiment 1: DANSE^* in a scenario with limit cycle behavior

We first demonstrate the effectiveness of the DANSE_1^* algorithm on the contrived example in the Appendix. Fig. 2 shows the MSE cost of node 3 over the different iterations, with the DANSE_1 and DANSE_1^* algorithm¹¹. For the DANSE_1 algorithm we observe a limit cycle of length 6, whereas the DANSE_1^* algorithm converges. It is noted that the non-convergence of the DANSE_1 algorithm is not due to numerical issues or estimation errors in the correlation matrices, as it is the case in the (more practical) scenario in the next subsection. Even with infinite precision, the algorithm will not converge in this example due to the contrived choice of input signals.

B. Experiment 2: DANSE^* in an ill-conditioned scenario

We now demonstrate the effectiveness of the DANSE_1^* algorithm in a simulation of the scenario that is schematically visualized in Fig. 3. Consider a two-channel latent process $\mathbf{d} = [d_1 \ d_2]$ containing two uncorrelated white source signals d_1 and d_2 ($Q = 2$), and a network with 4 nodes where nodes 1 and 2 estimate d_1 , and nodes 3 and 4 estimate d_2 ($K = 1$). All nodes are equipped with 2 sensors ($M_n = 2$). The noise at each sensor is white and uncorrelated with the noise at all other sensors. Node 1 and node 4 are in the neighborhood of source 1, and therefore collect noisy observations of d_1 . Node 1 observes d_1 with an SNR of 3 dB and node 4 is closer to the source, yielding an SNR of 12 dB. Node 1 and node 4 are assumed to be far away from source 2 such that the received

Fig. 2. MSE cost of node 3 versus number of iterations, for the example depicted in Fig. 7.

Fig. 3. Schematic visualisation of the ill-conditioned scenario, as simulated in Section VI-B. It visualizes the source-sensor distances, and it shows which signal each node aims to estimate. Note that node 2 and 4 observe their desired signal at a very low SNR due to the long distance to their target source, and short distance to the interfering source.

power of d_2 is 20 dB lower, i.e., -17 dB. The sensor signals at node 2 and 3 are similarly constructed, but with d_1 and d_2 swapped, i.e., they are close to source 2 and far away from source 1.

An important observation is that node 2 and node 4 observe their desired signal at a very low SNR, and will therefore not use their own sensor signals in their node-specific estimation process (i.e., $\mathbf{W}_{22}$ and $\mathbf{W}_{44}$ will have a small norm). However, since these are high SNR nodes, their sensor signals are very useful for node 3 and node 1 respectively. Therefore, the corresponding equilibrium state is close to a discontinuity of the DANSE_1 update function. This has a significant impact on the convergence of the DANSE_1 algorithm, as explained in Subsection V-A.

To model estimation errors on the signal statistics, we have added a random number to each entry of the correlation matrices $\mathbf{R}_{yy}$ and $\mathbf{R}_{yd_n}$ in each iteration. These numbers are drawn from a zero-mean Gaussian distribution with a standard deviation equal to one percent of the magnitude of the considered entry. The results of the DANSE_1 algorithm applied to the above scenario are shown in Fig. 4. It is observed that the DANSE_1 algorithm behaves randomly, which is due to the estimation errors that have a large effect because of the ill-conditioned nature of the scenario. If the entries of

¹⁰This safety measure lets the algorithm converge, but to a different equilibrium state than the (ill-conditioned) equilibrium state.

¹¹In both algorithms, the $\mathbf{G}_n$ variables are initialized with zeros, which explains the fact that the MSE at node 3 does not change in the first 2 iterations, in which the other two nodes perform an update.

TABLE II
THE DANSE_K^{*} ALGORITHM

The DANSE_K^{*} Algorithm

- 1) • Initialize $\mathbf{W}_{qq}^0$ and $\mathbf{G}_{q,-q}^0$, $\forall q \in \mathcal{N}$, as (well-conditioned) random matrices.
 • Initialize $\mathbf{C}_q^0 = \mathbf{W}_{qq}^0$.
 • $i \leftarrow 0$.
 • $n \leftarrow 1$.
- 2) • All nodes $q \in \mathcal{N}$ transmit observations of $\mathbf{z}_q^i = \mathbf{C}_q^{iH} \mathbf{y}_q$ to the other nodes.
 • Node n updates its local parameters $\mathbf{W}_{nn}^i$ and $\mathbf{G}_{n,-n}^i$ by computing (5), i.e.,

$$\begin{bmatrix} \mathbf{W}_{nn}^{i+1} \\ \mathbf{G}_{n,-n}^{i+1} \end{bmatrix} = (\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i)^{-1} \mathbf{R}_{\tilde{\mathbf{y}}_n d_n}^i. \quad (24)$$

If $\mathbf{R}_{\tilde{\mathbf{y}}_n \tilde{\mathbf{y}}_n}^i$ is rank deficient, the Moore-Penrose pseudo-inverse [33] is used instead.

- Let $\sigma_{n,\min}^{i+1}$ denote the K -th singular value of $\mathbf{W}_{nn}^{i+1}$ (ordered from large to small). Perform the following update:

$$\mathbf{C}_n^{i+1} = \begin{cases} \mathbf{W}_{nn}^{i+1} & \text{if } \sigma_{n,\min}^{i+1} \geq P \\ \mathbf{C}_n^i & \text{if } \sigma_{n,\min}^{i+1} < P \end{cases} \quad (25)$$

where P is a user-defined small positive number.

- All other nodes do not change their variables:

$$\forall q \in \mathcal{N} \setminus \{n\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i, \mathbf{C}_q^{i+1} = \mathbf{C}_q^i, \mathbf{G}_{q,-q}^{i+1} = \mathbf{G}_{q,-q}^i. \quad (26)$$

- 3) For the newly observed samples, each node $q \in \mathcal{N}$ generates an estimate of its desired signal:

$$\bar{\mathbf{d}}_q = \mathbf{W}_{qq}^{i+1H} \mathbf{y}_q + \mathbf{G}_{q,-q}^{i+1H} \mathbf{z}_{-q}^i. \quad (27)$$

- 4) • $i \leftarrow i + 1$.
 • $n \leftarrow (n \bmod N) + 1$.
- 5) Return to step 2

$\mathbf{C}_n$ are frozen whenever $\|\mathbf{W}_{nn}^{i+1}\| < 0.05$ (i.e., $P = 0.05$ in the DANSE₁^{*} algorithm), the algorithm behaves better and converges to an equilibrium. However, due to the random estimation errors in the correlation matrices, the MSE still has some minor fluctuations. The MSE of the centralized algorithm (and without estimation errors in the correlation matrix) is given as a reference. Note that DANSE₁ theoretically cannot achieve this lower bound due to the fact that $K < Q$, i.e., the assumptions in Theorem II.1 are not satisfied.

C. Experiment 3: large-scale scenario and influence of parameter P

In a last example, we simulate the DANSE₁ and DANSE₁^{*} algorithm in a large-scale WSN with $N = 30$ nodes, each having 2 sensors ($M_n = 2$). We again consider a two-channel latent process $\mathbf{d} = [d_1 \ d_2]$ containing two uncorrelated white source signals d_1 and d_2 ($Q = 2$). All nodes have the same noise power, but the signal power with which d_1 and d_2 are observed differs between the nodes, based on their distance from these sources. These distances are uniformly distributed, i.e., we used steering vectors drawn from a uniform distribution on the interval $[-0.5, 0.5]$ (negative entries correspond to 180 degree phase jumps, e.g., due to a polarity switch in the sensor). In general, this uniformity in the sensor signals means that there are no ‘outlier’ nodes that cause strange behavior in the DANSE₁ algorithm. Fig. 5 shows the output

Fig. 4. MSE cost of node 4 versus number of iterations for DANSE₁ and DANSE₁^{*} in an ill-conditioned scenario.

MSE of the DANSE₁ algorithm. We can indeed observe that the DANSE₁ algorithm behaves nicely, and converges to an equilibrium state (with some minor fluctuations due to the estimation errors on the correlation matrices that are introduced in each iteration). The figure also gives the output

Fig. 5. MSE cost of node 30 versus number of iterations for DANSE_1 and DANSE_1^* (with different values of P) in a scenario with uniformity in the sensor signals.

Fig. 6. MSE cost of node 30 versus number of iterations for DANSE_1 and DANSE_1^* (with different values of P) in an ill-conditioned scenario.

MSE for the DANSE_1^* algorithm for different values of P . The MSE is higher for DANSE_1^* due to the fact that the compression matrix C_n is sometimes frozen, which happens more frequently when P is large. This shows that the value P yields a trade-off between robustness and output MSE, and should be carefully chosen.

In the next simulation, we used the same set-up, but now also introduce 5 ‘outlier’ nodes, i.e., for which the signals are generated similarly to the signals in experiment 2 (i.e., they either observe d_1 or d_2 with an extremely high SNR). Note that the other 25 nodes have the same sensor signals as in the experiment shown in Fig. 5. The results are shown in Fig. 6, again demonstrating that the DANSE_1^* behaves nicely (if P is chosen large enough, i.e., $P \geq 0.05$), whereas the DANSE_1 algorithm has very random behavior due to numerical issues due to the fact that the equilibrium estimator of the outlier nodes is close to a discontinuity of the DANSE updating function. This unstable behavior in the outlier nodes also affects all other nodes in the network.

VII. CONCLUSIONS

In this paper, we have considered distributed signal estimation in sensor networks where the nodes exchange compressed sensor signal observations to estimate different node-specific signals. We have revisited the so-called distributed adaptive node-specific signal estimation (DANSE) algorithm, which applies to the case where the nodes share a common interest, and we have cast it in the more general setting where the nodes have different interests. By establishing a link between the DANSE algorithm and game theory, we have pointed out that any equilibrium of the DANSE algorithm is a Nash equilibrium of the associated game between nodes. This provides an intuitive interpretation to the resulting signal estimators. By using a result from fixed point theory, we have proven the existence of an equilibrium state for the DANSE algorithm, even if the common interest data model is not or

Fig. 7. A scenario where DANSE_1 does not converge, but instead gets stuck in a limit cycle.

only approximately satisfied. The equilibrium state existence proof has also revealed a problem with discontinuities in the DANSE update function. Although in practice these discontinuities are encountered with zero probability, it is possible to have an ill-conditioned equilibrium state, i.e., an equilibrium state that is close to one of these discontinuities, which may result in non-convergence of the algorithm. However, since the discontinuities are identifiable, they can easily be avoided by applying a minor heuristic modification to the algorithm, yielding the so-called DANSE^* algorithm. We have demonstrated the effectiveness of the DANSE^* algorithm by means of numerical examples.

TABLE III
THE SIX ITERATIONS OF THE LIMIT CYCLE BEHAVIOR OF DANSE₁ IN THE SCENARIO DESCRIBED IN FIG. 7.

	node 1	node 2	node 3
iteration 1	$z_1 = \eta_1 (d_1 + d_2 + d_3)$ $\mathbf{w}_{11} = [\eta_1 \ 0]^T$ $g_{12} = \gamma_1, g_{13} = \gamma_2$	$z_2 = \eta_2 (d_2 + n_2)$ $\mathbf{w}_{22} = [0 \ \eta_2]^T$ $g_{21} = 0, g_{23} = 0$	$z_3 = \eta_3 (d_3 + n_3)$ $\mathbf{w}_{33} = [0 \ \eta_3]^T$ $g_{31} = 0, g_{32} = 0$
iteration 2	$z_1 = \eta_1 (d_1 + d_2 + d_3)$ $\mathbf{w}_{11} = [\eta_1 \ 0]^T$ $g_{12} = 0, g_{13} = 0$	$z_2 = -(d_1 + d_3)$ $\mathbf{w}_{22} = [-1 \ 0]^T$ $g_{21} = 1/\eta_1, g_{23} = 0$	$z_3 = \eta_3 (d_3 + n_3)$ $\mathbf{w}_{33} = [0 \ \eta_3]^T$ $g_{31} = 0, g_{32} = 0$
iteration 3	$z_1 = \eta_1 (d_1 + d_2 + d_3)$ $\mathbf{w}_{11} = [\eta_1 \ 0]^T$ $g_{12} = 0, g_{13} = 0$	$z_2 = -(d_1 + d_3)$ $\mathbf{w}_{22} = [-1 \ 0]^T$ $g_{21} = 1/\eta_1, g_{23} = 0$	$z_3 = -d_1$ $\mathbf{w}_{33} = [-1 \ 0]^T$ $g_{31} = 0, g_{32} = -1$
iteration 4	$z_1 = 0$ $\mathbf{w}_{11} = [0 \ 0]^T$ $g_{12} = 0, g_{13} = -1$	$z_2 = -(d_1 + d_3)$ $\mathbf{w}_{22} = [-1 \ 0]^T$ $g_{21} = 1/\eta_1, g_{23} = 0$	$z_3 = -d_1$ $\mathbf{w}_{33} = [-1 \ 0]^T$ $g_{31} = 0, g_{32} = -1$
iteration 5	$z_1 = 0$ $\mathbf{w}_{11} = [0 \ 0]^T$ $g_{12} = 0, g_{13} = -1$	$z_2 = \eta_2 (d_2 + n_2)$ $\mathbf{w}_{22} = [0 \ \eta_2]^T$ $g_{21} = 0, g_{23} = 0$	$z_3 = -d_1$ $\mathbf{w}_{33} = [-1 \ 0]^T$ $g_{31} = 0, g_{32} = -1$
iteration 6	$z_1 = 0$ $\mathbf{w}_{11} = [0 \ 0]^T$ $g_{12} = 0, g_{13} = -1$	$z_2 = \eta_2 (d_2 + n_2)$ $\mathbf{w}_{22} = [0 \ \eta_2]^T$ $g_{21} = 0, g_{23} = 0$	$z_3 = \eta_3 (d_3 + n_3)$ $\mathbf{w}_{33} = [0 \ \eta_3]^T$ $g_{31} = 0, g_{32} = 0$

Appendix: DANSE algorithm with limit cycles

Consider a network with 3 nodes ($\mathcal{N} = \{1, 2, 3\}$), where node j estimates the desired signal d_j . It is assumed that the three desired signals d_1 , d_2 and d_3 are uncorrelated, and therefore $Q = 3$. We set $K = 1$, and since $K < Q$, data model (13) is not satisfied. Each node has two sensors, collecting the signals as shown in Fig. 7, where n_1 , n_2 and n_3 are noise signals that are neither correlated to each other nor to any d_j , $\forall j \in \{1, 2, 3\}$. Since $K = 1$, the $\mathbf{W}$'s and $\mathbf{G}$'s are replaced by their vector and scalar equivalents $\mathbf{w}$ and g . The sensor signals in this example are well-chosen such that the DANSE₁ algorithm has a limit cycle behavior due to updates to points for which the update function is discontinuous, as explained in the proof of Theorem IV.3. Assume that we initialize nodes 2 and 3 according to the first row of Table III, and that node 1 performs the first update of its estimators, i.e., $\mathbf{w}_{11}$, g_{12} and g_{13} . Since the second sensor signal of node 1 is uncorrelated to any of the other signals, node 1 will make a linear combination of only three signals, i.e., its first sensor signal, z_2 and z_3 . This is also shown in the first row of Table III. The other rows of this table show the changes of the parameters over the different iterations, where the nodes update in a sequential round robin fashion. The result of the seventh iteration is exactly the same as the result of the first iteration, i.e., the DANSE₁ algorithm does not converge, but gets stuck in a limit cycle of length 6.

Iteration 4 is the crucial iteration here, since node 1 updates its $\mathbf{w}_{11}$ to an all-zero vector. This is because it receives the signal $z_3 = -d_1$, and therefore, it can perfectly estimate its desired signal without using its own sensor signals. This corresponds to a discontinuous point of the update function F of the DANSE algorithm, as described in the proof of Theorem IV.3. Since z_1 is now a zero-signal, node 2 cannot use any information anymore from node 1 and needs to switch

to another estimator, and this effect ripples through until we arrive in the initial state again. It is not difficult to see that the algorithm will get stuck in this limit cycle for every possible initialization, i.e., the algorithm never converges. This means that there is no equilibrium state. Applying a norm constraint on the g 's will generate an artificial equilibrium state that depends on the choice of T in (15). Even then, the algorithm will still converge to this limit cycle, so the artificial equilibrium state is seen to be unstable.

REFERENCES

- [1] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: sequential node updating," *IEEE Transactions on Signal Processing*, vol. 58, pp. 5277–5291, 2010.
- [2] —, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part II: simultaneous & asynchronous node updating," *IEEE Transactions on Signal Processing*, vol. 58, pp. 5292–5306, 2010.
- [3] —, "Robust distributed noise reduction in hearing aids with external acoustic sensor nodes," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 530435, 14 pages, 2009. doi:10.1155/2009/530435.
- [4] T. C. Lawin-Ore and S. Doclo, "Analysis of rate constraints for MWF-based noise reduction in acoustic sensor networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. pp. 269–272.
- [5] A. Bertrand and M. Moonen, "Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology," *IEEE Trans. Signal Processing*, vol. 59, no. 5, pp. 2196–2210, May 2011.
- [6] S. Markovich Golan, S. Gannot, and I. Cohen, "A reduced bandwidth binaural MVDR beamformer," in *Proc. of the International Workshop on Acoustic Echo and Noise Control (IWAENC)*, Tel-Aviv, Israel, Aug. 2010.
- [7] A. Bertrand and M. Moonen, "Distributed node-specific LCMV beamforming in wireless sensor networks," *IEEE Transactions on Signal Processing*, 2012.
- [8] 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 and Language Processing*, vol. 17, pp. 38–51, Jan. 2009.

- [9] J. Nash, "Non-cooperative games," *The Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951.
- [10] A. Bertrand, "Signal processing algorithms for wireless acoustic sensor networks," Ph.D. dissertation, Katholieke Universiteit Leuven, Leuven, Belgium, May 2011.
- [11] I. D. Schizas, G. B. Giannakis, and Z.-Q. Luo, "Distributed estimation using reduced-dimensionality sensor observations," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4284–4299, Aug. 2007.
- [12] Y. Zhu, E. Song, J. Zhou, and Z. You, "Optimal dimensionality reduction of sensor data in multisensor estimation fusion," *IEEE Transactions on Signal Processing*, vol. 53, no. 5, pp. 1631–1639, May 2005.
- [13] J. Fang and H. Li, "Optimal/near-optimal dimensionality reduction for distributed estimation in homogeneous and certain inhomogeneous scenarios," *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4339–4353, 2010.
- [14] H. Ochiai, P. Mitran, H. Poor, and V. Tarokh, "Collaborative beamforming for distributed wireless ad hoc sensor networks," *IEEE Transactions on Signal Processing*, vol. 53, no. 11, pp. 4110–4124, 2005.
- [15] M. F. A. Ahmed and S. A. Vorobyov, "Collaborative beamforming for wireless sensor networks with gaussian distributed sensor nodes," *IEEE Trans. Wireless. Comm.*, vol. 8, pp. 638–643, February 2009.
- [16] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Joint source-channel communication for distributed estimation in sensor networks," *IEEE Trans. Information Theory*, vol. 53, no. 10, pp. 3629–3653, oct. 2007.
- [17] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.
- [18] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, pp. 1035–1048, March 2010.
- [19] A. Bertrand, M. Moonen, and A. H. Sayed, "Diffusion bias-compensated RLS estimation over adaptive networks," *IEEE Trans. Signal Processing*, Nov. 2011.
- [20] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Performance analysis of the consensus-based distributed LMS algorithm," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 981030, 19 pages, 2009. doi:10.1155/2009/981030.
- [21] A. Speranzon, C. Fischione, K. Johansson, and A. Sangiovanni-Vincentelli, "A distributed minimum variance estimator for sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 609–621, May 2008.
- [22] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.
- [23] P. Braca, S. Marano, and V. Matta, "Running consensus in wireless sensor networks," in *Proc. Int. Conf. on Information Fusion*, July 2008, pp. 1–6.
- [24] A. Bertrand and M. Moonen, "Consensus-based distributed total least squares estimation in ad hoc wireless sensor networks," *IEEE Trans. Signal Processing*, vol. 59, no. 5, pp. 2320–2330, May 2011.
- [25] A. Bertrand, J. Callebaut, and M. Moonen, "Adaptive distributed noise reduction for speech enhancement in wireless acoustic sensor networks," in *Proc. Int. Workshop on Acoustic Echo and Noise Control (IWAENC)*, Tel Aviv, Israel, Aug. 2010.
- [26] S. Srinivasan and A. C. Den Brinker, "Rate-constrained beamforming in binaural hearing aids," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 257197, 14 pages, 2009.
- [27] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, Jul. 1973.
- [28] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): design and construction," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 626–643, March 2003.
- [29] J. Chou, D. Petrovic, and K. Ramchandran, "A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks," in *Proc. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, March 2003, pp. 1054–1062.
- [30] M. Gastpar, P. Dragotti, and M. Vetterli, "The distributed Karhunen-Loève transform," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5177–5196, 2006.
- [31] O. Roy and M. Vetterli, "Rate-constrained collaborative noise reduction for wireless hearing aids," *IEEE Transactions on Signal Processing*, vol. 57, no. 2, pp. 645–657, 2009.
- [32] M. Duarte, S. Sarvotham, D. Baron, M. Wakin, and R. Baraniuk, "Distributed compressed sensing of jointly sparse signals," in *In Asilomar Conf. Signals, Sys., Comput*, 2005, pp. 1537–1541.
- [33] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. Baltimore: The Johns Hopkins University Press, 1996.
- [34] A. H. Sayed, *Adaptive Filters*. NJ: John Wiley & Sons, 2008.
- [35] O. Hoshuyama, A. Sugiyama, and A. Hirano, "A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filters," *IEEE Trans. on Signal Processing*, vol. 47, no. 10, pp. 2677–2684, oct 1999.
- [36] A. B. MacKenzie and L. A. DaSilva, *Game Theory for Wireless Engineers (Synthesis Lectures on Communications)*. Morgan & Claypool Publishers, 2006.
- [37] E. Koutsoupias and C. Papadimitriou, "Worst-case equilibria," *In STACS99: 16th International Symposium on Theoretical Aspects of Computer Science*, pp. 404–413, 1999.
- [38] V. I. Istratescu, *Fixed Point Theory, An Introduction*, ser. Mathematics and Its Applications, Vol. 7. Dordrecht, Holland: Reidel. Publishing Co., 1981.