Greedy Distributed Node Selection for Node-Specific Signal Estimation in Wireless Sensor Networks

J. Szurley[a,1,*], A. Bertrand[a,1,2], P. Ruckebusch[b], I. Moerman[b], M. Moonen[a,1]

[a]*ESAT-SCD (SISTA) / iMinds - Future Health Department, KU Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium*
[b]*Ghent University - iMinds, Dept. of Information Technology (INTEC), Gaston Crommenlaan 8 Bus 201, 9050 Ghent, Belgium*

Abstract

A wireless sensor network is envisaged that performs signal estimation by means of the distributed adaptive node-specific signal estimation (DANSE) algorithm. This wireless sensor network has constraints such that only a subset of the nodes are used for the estimation of a signal. While an optimal node selection strategy is NP-hard due to its combinatorial nature, we propose a greedy procedure that can add or remove nodes in an iterative fashion until the constraints are satisfied based on their *utility*. With the proposed definition of utility, a centralized algorithm can efficiently compute each nodes's utility at hardly any additional computational cost. Unfortunately, in a distributed scenario this approach becomes intractable. However by using the convergence and optimality properties of the DANSE algorithm, it is shown that for node removal, each node can efficiently compute a utility upper bound such that the MMSE increase after removal will never exceed this value. In the case of node addition, each node can determine a utility lower bound such that the MMSE decrease will always exceed this value once added. The greedy node selection procedure can then use these upper and lower bounds to facilitate distributed node selection.

Keywords: Wireless sensor networks; distributed signal estimation; node selection

1. Introduction

A wireless sensor network (WSN) utilizes a collection of sensor nodes to observe a physical phenomenon where collected sensor observations may be used to monitor or estimate a parameter or signal. There are many key benefits of using a WSN over a single sensor, e.g., to collect a wider range of spatial and temporal information and to ensure redundancy in case of sensor failure, which accounts for the rapid proliferation of their use in many applications [1, 2, 3, 4].

*Corresponding author
Email addresses: joseph.szurley@esat.kuleuven.be (J. Szurley), alexander.bertrand@esat.kuleuven.be (A. Bertrand), peter.ruckebusch@intec.ugent.be (P. Ruckebusch), ingrid.moerman@intec.ugent.be (I. Moerman), marc.moonen@esat.kuleuven.be (M. Moonen)

[1]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 P7/ 'Dynamical systems, control and optimization' (DYSCO) 2012-2017, Research Project iMinds, Research Project FWO nr. nr. G.0763.12 'Wireless Acoustic Sensor Networks for Extended Auditory Communication'.

[2]A. Bertrand is supported by a Postdoctoral Fellowship of the Research Foundation Flanders (FWO).

Many sensor networks are posed with the task of estimating a network-wide desired signal or parameter by means of cooperative communication, i.e., every node contributes to a global estimation problem. This framework may be modified to the case where each node tries to estimate its own node-specific desired signals while again using its local signal observations and those provided by the other nodes in the network. In this case, each node could estimate the source signals as they are observed by the node's own local sensors. This may be important if spatial information needs to be retained in the estimates, such as noise reduction algorithms for cooperative hearing aids, which require node-specific signal estimates to not loose the spatial cues for directional hearing, i.e., the signals have to be estimated as they impinge at the two ears [5, 6, 7].

In a centralized WSN, the nodes relay their observations to a main base station or fusion center (FC) where all information is aggregated and processed in order to estimate a set of desired signals. This type of WSN is susceptible to a single point failure, i.e., if the FC fails the network is no longer able to process the collected information. Furthermore, transmitting all the raw sensor signals to the FC may require a significant communication bandwidth. Therefore instead of requiring that each node transmits its observations to a FC it is beneficial to have a distributed WSN framework where the computational load may be divided among the nodes in the network while still being able to reach the same solution as in the centralized case. Ideally this distributed WSN should also be able to perform the same functions of a centralized WSN, preferably with less communication bandwidth compared to a centralized, FC-based, approach.

Therefore, in this paper, the envisaged distributed WSN performs signal estimation by means of the distributed adaptive node-specific signal estimation (DANSE) algorithm [8]. The DANSE algorithm performs a linear minimum mean square error (MMSE) estimation of a set of node-specific desired signals at each node, based on the iterative computation of a set of distributed spatial filters. It has been used for such applications as acoustic beamforming and distributed noise reduction in hearing aids or wireless acoustic sensor networks [5]. A benefit of using the DANSE algorithm is that it can reduce the overall communication bandwidth consumption of the system while still converging to the full-bandwidth solution, i.e., when each node transmits each of its uncompressed sensor signals to all other nodes.

While previous implementations of the DANSE algorithm have relied on fully-connected networks [8] or tree topologies [9], it has not explicitly taken network constraints into account. Due to the nodes being deployed over large distances or in hostile environments as well as their limited battery life, it is often desired to limit the number of active nodes at any given time. Indeed, if the WSN is densely deployed, many sensors record redundant data and may be placed in an inactive or sleep mode in order to preserve the network lifetime. Therefore, the number of total active nodes in the system, K, should be reduced to a smaller subset, N. This is an inherent combinatorial optimization problem with $\binom{K}{N}$ combinations. As the number of nodes K increases, the computational time required to find the optimal subset becomes infeasible.

The proposed selection strategy in this paper allows each node to determine its effect on the MMSE in a computationally efficient manner without relying on a FC while estimating a node-specific desired signal. There are however

several methods that have been proposed in order to perform node selection in a WSN [10] where each of these methods differ in one or more ways from the techniques proposed in this paper. Joshi *et. al.* have proposed a formulation that relies on a MAP estimator and uses a convex cone in order to measure the impact of removing a sensor, which relies on knowing the underlying statistics of the system [11]. Other methods are able to evaluate performance bounds compared to the optimal solution [12], using MAP or maximum likelihood (ML) estimators. In using either an MAP or ML estimator it is assumed that the WSN is performing *parameter* estimation of the underlying statistics of the system, whereas the method proposed in this paper aims to perform *signal* estimation. In [13] a distributed strategy has been proposed that is compared to a centralized approach but is cast as a multi-armed bandit problem which looks to activate a certain subset of sensors to aid in estimation in consensus based averaging which again aims to estimate a desired *parameter* of the system. Thatte *et. al.* have proposed placing bounds on the MMSE under various network topologies but rely on a FC to perform the estimation [14] whereas the proposed method looks to accomplish this in a completely distributed fashion.

In order to select nodes from a given set, we introduce the concept of *utility*, which is assigned to a node as a way to determine its importance to the signal estimation problem at hand [15, 16, 17]. It is defined as the increase or decrease of the MMSE after removing or adding the respective node and re-optimizing the estimators. Since each node is assumed to have multiple sensors, a new utility computation algorithm is developed which efficiently computes the utility of a set of sensors at once, which then corresponds to the utility of the node.

Due to the distributed computation of the proposed utility as well as the combinatorial nature of node selection we devise a distributed algorithm that uses a greedy procedure to add and remove nodes from the network as in [18]. For node removal, at each iteration, the greedy procedure in this paper will remove the node with the lowest utility. Likewise, for node addition, this greedy procedure will add the node with the highest utility. Similar greedy techniques have been applied to radar arrays [19, 20] where the change of the MMSE is used for target detection. It should also be noted that the utility proposed in our framework differs from other definitions such as [12, 13, 21, 22] which rely on the concept of submodularity which the proposed utility does not have.

Although the exact utility cannot be computed in the DANSE framework, it can be shown that we can compute upper and lower bounds on the utility, i.e., the increase or decrease of the MMSE when removing or adding nodes respectively. By using the convergence and optimality properties of the DANSE algorithm we show that the nodes can independently decide whether to stay active in the current network based on their local utility estimation. The network therefore does not need to rely on a FC in order to facilitate node selection. However since the DANSE algorithm allows each node to estimate a node-specific signal, each sensor will also have a different utility for each individual estimation problem. Therefore, the computed utilities (and their bounds) are referenced to a common network-wide utility measure to circumvent this problem.

The organization of the paper is as follows : In Section 2 the data model of the signals is provided along with the MMSE-based spatial filtering procedure that each node uses in order to estimate a node-specific desired signal. In Section 3 the utility is described in a centralized scenario with a greedy node selection procedure, and we also define a

network-wide utility measure that is common for all the node-specific estimation problems. In Section 4 the DANSE algorithm is reviewed along with its convergence properties. In Section 5 the utility is described in a distributed scenario where the DANSE algorithm is in place and it is shown how it can be used in the greedy node selection as an upper and lower bound with respect to the increase or decrease to the MMSE. Simulations are performed in Section 6 where the centralized and distributed scenarios are compared. An adaptive scenario is also simulated that shows the use of the utility and the greedy node selection procedure in a real-time environment.

2. Data Model

Consider a WSN with K nodes. We assume that the nodes communicate with one another in a fully-connected fashion in a synchronous setting where there is an ideal communication link between nodes. We assume that each node, $k \in \{1 \dots K\}$, observes M_k complex[3] sensor signals where the total number of signals in the network is given by $M = \sum_{k=1}^{K} M_k$. The M_k sensor signals may be provided by different sensors at node k, or from remote sensors that forward their observations to node k. The received signal of sensor (or channel) m of node k is given as

$$y_{km} = x_{km} + v_{km}, \ m = 1, \dots, M_k \tag{1}$$

where x_{km} is a desired signal component (the signal model for x_{km} will be defined later, see (4)), v_{km} is an additive noise component which may be correlated to the noise in other sensors or nodes. It is assumed that the desired signal and noise components are stationary, ergodic and statistically independent.[4] The goal of each node is to estimate one or more node-specific versions of the desired signals x_{km}, as will be explained later.

The received signals at node k are stacked in an M_k-dimensional vector as

$$\mathbf{y}_k = [y_{k1} \dots y_{kM_k}]^T \tag{2}$$

and the vectors $\mathbf{x}_k$ and $\mathbf{v}_k$ are defined similarly such that

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k. \tag{3}$$

The desired signal components of node k are assumed to be linear mixtures of Q source signals given as

$$\mathbf{x}_k = \mathbf{A}_k \mathbf{s} \tag{4}$$

where $\mathbf{A}_k$ is an $M_K \times Q$-dimensional complex-valued steering matrix and $\mathbf{s}$ is a Q-dimensional stochastic vector variable containing the Q source signals. The objective of node k is to estimate an unobservable J-channel node-specific desired signal, $\bar{\mathbf{x}}_k$, defined by the desired signal component x_{km} in J local reference sensors. Without loss of

[3] We assume that all signals are complex valued in order to allow for a frequency domain representation.

[4] In practice, e.g., for speech processing, this stationarity and ergodicity assumption can be relaxed to short-term stationarity and ergodicity provided that the finite signal segments behave in this fashion.

generality (w.l.o.g.), we assume that the first J channels of $\mathbf{y}_k$ correspond to these reference sensors, i.e.,

$$\bar{\mathbf{x}}_k = [\mathbf{I}|\mathbf{0}]\mathbf{x}_k \tag{5}$$

where $\mathbf{I}$ is a J-dimensional identity matrix and $\mathbf{0}$ is a $J \times (M_k - J)$-dimensional matrix with all entries equal to 0. The node specific $\bar{\mathbf{x}}_k$ can also be represented in terms of its node specific steering matrix, $\bar{\mathbf{A}}_k$, as

$$\begin{aligned}\bar{\mathbf{x}}_k &= [\mathbf{I}|\mathbf{0}]\mathbf{A}_k\mathbf{s} \\ &= \bar{\mathbf{A}}_k\mathbf{s}.\end{aligned} \tag{6}$$

It is noted that we do not aim to obtain the original source signals in $\mathbf{s}$, i.e., we do not aim to unmix the signals in $\bar{\mathbf{x}}_k$ or to equalize for the filtering due to the steering matrix $\bar{\mathbf{A}}_k$. Instead, we want to estimate the desired signal components as they are locally observed in the J reference sensors at node k. This is important if spatial information must be retained in the signal estimates, which when needed, requires a node-specific estimator. In the sequel, we assume that the dimension of the node specific $\bar{\mathbf{x}}_k$ is equal to that of the dimension of the source signal space,[5] i.e., $J = Q$, where we assume that $\bar{\mathbf{A}}_k$ is invertible $\forall k \in \{1, \ldots, K\}$ and that $Q \leq M_k$.

We first assume that each node has access to all M signals, where all $\mathbf{y}_k$, $\mathbf{x}_k$, $\mathbf{v}_k$ vectors are stacked into M-dimensional vectors $\mathbf{y}$, $\mathbf{x}$, $\mathbf{v}$ respectively, and we refer to this case as the centralized estimation. We consider a linear MSE cost function based on the node-specific linear estimator, $\mathbf{W}_k$, given as

$$J_k(\mathbf{W}_k) = E\{\|\bar{\mathbf{x}}_k - \mathbf{W}_k^H\mathbf{y}\|_2^2\} \tag{7}$$

where $E\{.\}$ is the expected value operator, $\|.\|_2^2$ is the l_2 norm squared, and H is the complex conjugate operator. The linear MMSE estimator that minimizes (7) is given by [23]

$$\hat{\mathbf{W}}_k = \mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} \tag{8}$$

where $\mathbf{R}_{\mathbf{yy}} = E\{\mathbf{y}\mathbf{y}^H\}$ is the sensor signal correlation matrix and $\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} = E\{\mathbf{y}\bar{\mathbf{x}}_k^H\}$ is a cross-correlation matrix between the sensor signals and the desired signal components at node k. Although $\bar{\mathbf{x}}_k$ is unobservable, due to the independence with the additive noise, there are several strategies that can be used to estimate the cross-correlation matrix depending on the application [8, 24, 25, 26]. In Section 6.1 a method to estimate the correlation matrices will be discussed.

Using the optimal estimator (8) the minimum cost is given as

$$\begin{aligned}J_k(\hat{\mathbf{W}}_k) &= E\{\|\bar{\mathbf{x}}_k - \hat{\mathbf{W}}_k^H\mathbf{y}\|_2^2\} \\ &= \sum_{j=1}^{J} P_{k_{xj}} - \mathbf{r}_{\mathbf{y}\bar{\mathbf{x}}_{kj}}^H\hat{\mathbf{w}}_{kj}\end{aligned} \tag{9}$$

where $\hat{\mathbf{w}}_{kj}$ and $\mathbf{r}_{\mathbf{y}\bar{\mathbf{x}}_{kj}}$ represent the jth column of $\hat{\mathbf{W}}_k$ and $\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k}$ respectively and $P_{k_{xj}} = E\{\|x_{kj}\|_2^2\}$ represents the desired signal power in the jth channel.

[5] In Section 4 this assumption is used to ensure the convergence of the DANSE algorithm to that of a centralized scenario.

We define the $Q \times Q$-dimensional MMSE cost matrix

$$\hat{\mathbf{J}}_k \triangleq \mathbf{R}_{\bar{\mathbf{x}}_k \bar{\mathbf{x}}_k} - \mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k}^H \hat{\mathbf{W}}_k \tag{10}$$

where $\mathbf{R}_{\bar{\mathbf{x}}_k \bar{\mathbf{x}}_k} = E\{\bar{\mathbf{x}}_k \bar{\mathbf{x}}_k^H\}$ is the desired signal correlation matrix. The minimum cost can then be compactly represented as the trace of (10), i.e.,

$$J_k(\hat{\mathbf{W}}_k) = \mathrm{Tr}\{\hat{\mathbf{J}}_k\}. \tag{11}$$

We now consider the desired signal components belonging to another node q, $\bar{\mathbf{x}}_q$, defined similarly as in (4). Since we assume that the desired signal components of each node are linear mixtures of the same Q source signals in $\mathbf{s}$, it can be shown that the MMSE cost matrices and MMSE estimators between nodes are related to one another by their steering matrices. The MMSE cost matrices between node k and node q are related as shown in Appendix A

$$\hat{\mathbf{J}}_k = \bar{\mathbf{A}}_{qk}^{-H} \hat{\mathbf{J}}_q \bar{\mathbf{A}}_{qk}^{-1} \tag{12}$$

where $\bar{\mathbf{A}}_{qk} = \bar{\mathbf{A}}_k^{-H} \bar{\mathbf{A}}_q^H$. This is a direct result of $\bar{\mathbf{x}}_k$ and $\bar{\mathbf{x}}_q$ being related by their steering matrices $\bar{\mathbf{A}}_k$ and $\bar{\mathbf{A}}_q$, respectively. Likewise the optimal estimators of node k and node q are related to one another by a product of their steering matrices,

$$\begin{aligned} \hat{\mathbf{W}}_k &= \hat{\mathbf{W}}_q \bar{\mathbf{A}}_q^{-H} \bar{\mathbf{A}}_k^H \\ &= \hat{\mathbf{W}}_q \bar{\mathbf{A}}_{qk}^{-1}. \end{aligned} \tag{13}$$

3. Utility

Suppose that each node in the network has calculated its own $M \times Q$ optimal estimator (8) and, due to constraints imposed on the system and the possibility of new nodes becoming available, we would like to remove or add nodes to the network while controlling the effect on the MMSE at each node. This problem is inherently combinatorial in nature so we will therefore fall back on the use of greedy heuristics. In order to determine the effect of removing or adding a node, a utility measure is introduced that quantifies how much a node contributes to the current estimation.

The utility of a node is defined as the difference in the MMSE (9), when one node is added or removed from the estimation after re-optimizing the estimator $\mathbf{W}_k$. In [15] it is described how the utility of a sensor can be computed with a relatively small computational complexity when compared to a naive approach which would be to remove one sensor from the system and then to recalculate the minimum cost to check its contribution and to do this for all sensors.

While in the centralized approach computational complexity may not be a concern, in the distributed case nodes often have a smaller processing capability. In this section, we explain how the techniques in [15] can be generalized to find an expression that computes the utility of a group of sensors (e.g., the M_k sensors corresponding to node k) instead of a single sensor. This expression will then later be used for node selection in the distributed scenario.

3.1. Utility for node removal

The utility, $U_{k_{-q}}$, of node q's sensors, $\mathbf{y}_q$, with respect to node k's estimation problem is defined as the increase in MMSE when $\mathbf{y}_q$ is removed from node k's estimation problem, i.e.,

$$U_{k_{-q}} = J_{k_{-q}}(\hat{\mathbf{W}}_{k_{-q}}) - J_k(\hat{\mathbf{W}}_k) \tag{14}$$

where the subscript $-q$ indicates that node q's sensors are removed from the function and $\hat{\mathbf{W}}_{k_{-q}}$ is referred to as the optimal fall-back estimator when node q is removed. Note that $\hat{\mathbf{W}}_{k_{-q}}$ is *not* equal to $\hat{\mathbf{W}}_k$ with M_q rows removed but it is equal to the re-optimized MMSE estimator that minimizes $J_{k_{-q}}$ in which the sensors of node q are removed. Using (11), the utility of node q with respect to node k's estimation problem is given by

$$\begin{aligned} U_{k_{-q}} &= \mathrm{Tr}\{\hat{\mathbf{J}}_{k_{-q}} - \hat{\mathbf{J}}_k\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{-q}}\}. \end{aligned} \tag{15}$$

In order to calculate $\hat{\mathbf{W}}_{k_{-q}}$ without having to take a full inverse as given in (8), we first partition the $M \times M$ sensor signal correlation matrix, $\mathbf{R}_{\mathbf{yy}}$, as follows

$$\mathbf{R}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} & \mathbf{R}_{\mathbf{y}_q\mathbf{y}_{-q}} \\ \mathbf{R}^H_{\mathbf{y}_q\mathbf{y}_{-q}} & \mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} \end{bmatrix} \tag{16}$$

where, for the sake of an easy exposition but w.l.o.g., it is assumed that the first $M_q \times M_q$ elements of the matrix correspond to node q's sensors (i.e., $q = 1$). Notice that to remove another nodes's sensors, the indices would need to be shifted accordingly which does not affect generality of the utility computation described in the sequel.

The calculation of $\hat{\mathbf{W}}_{k_{-q}}$, using the definition given in (8), requires the inverse of only a portion of the sensor signal correlation matrix, namely $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, which is currently unknown. In order to calculate the inverse without having to first remove the corresponding rows and columns that pertain to the node q's sensors and calculate $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, we block partition the current inverse $\mathbf{R}^{-1}_{\mathbf{yy}}$ as

$$\mathbf{R}^{-1}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C} \end{bmatrix} \tag{17}$$

where $\mathbf{S}$ is an invertible $M_q \times M_q$-dimensional matrix, $\mathbf{V}$ is an $M_q \times (M - M_q)$-dimensional matrix and $\mathbf{C}$ is an $(M - M_q) \times (M - M_q)$-dimensional matrix. It is noted that this matrix inverse (including all of its block components) is already known from the computation of the current MMSE estimator at node k. Using the block form of the matrix inversion lemma [27], $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ may be calculated using only the known values in the current $\mathbf{R}^{-1}_{\mathbf{yy}}$ matrix as

$$\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} = \mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V} \tag{18}$$

which is the Schur complement of $\mathbf{S}$ in $\mathbf{R}^{-1}_{\mathbf{yy}}$.

The current optimal estimator $\hat{\mathbf{W}}_k$ is also block partitioned as

$$\hat{\mathbf{W}}_k = \begin{bmatrix} \hat{\mathbf{W}}_{k_q} \\ \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} \end{bmatrix} \tag{19}$$

where $\hat{\mathbf{W}}_{k_q}$ is an $M_q \times Q$-dimensional matrix that represents the estimator values applied to node q's sensors and $\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}$ is an $(M - M_q) \times Q$-dimensional matrix that represents the estimator values applied to the other sensors. In Appendix B, it is shown that the optimal fall-back estimator is given as

$$\hat{\mathbf{W}}_{k_{-q}} = \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H \mathbf{S}^{-1} \hat{\mathbf{W}}_{k_q} \tag{20}$$

which are all known values of the current estimate. Again shown in Appendix B, using (20) the utility is given as

$$\boxed{U_{k_{-q}} = \mathrm{Tr}\{\hat{\mathbf{W}}_{k_q}^H \mathbf{S}^{-1} \hat{\mathbf{W}}_{k_q}\}} \tag{21}$$

$$= \mathrm{Tr}\{\mathbf{U}_{k_{-q}}\} \tag{22}$$

where

$$\mathbf{U}_{k_{-q}} \triangleq \hat{\mathbf{W}}_{k_q}^H \mathbf{S}^{-1} \hat{\mathbf{W}}_{k_q} \tag{23}$$

which only relies on an inversion[6] of an $M_q \times M_q$ matrix $\mathbf{S}$, and the current estimator values. In order to find the impact of any other node to the current estimation of node k, $U_{k_{-i}}, i \in \{1 \dots J\}$, the estimator coefficients and partial inverse of (21) can be changed to the corresponding indices.

Instead of using (21), the utility could be found naively by removing a nodes signals and calculating the new estimator $\hat{\mathbf{W}}_{k_{-q}}$ which relies on the inverse of $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ having a worst case scenario $O(M - M_q)^3$ computational complexity. Figure 1 shows the computational complexity of finding $U_{k_{-q}}$ using this naive approach compared to using (21) where all nodes are assumed to have 4 sensor signals. The dashed line at the bottom indicates the computational complexity of calculating the utility using (21) which is constant due to the fact of only relying on the inverse of always the same, in this case, 4×4-dimensional matrix. The total computational complexity is given for different matrix inversion algorithms [28] which have an increasingly large number of calculations for an increasing number of nodes. In [15] a less generalized expression compared to (21) was derived which finds the utility of a single sensor at computational cost of $O(M)$. The utility computed using (21) could instead be found in an iterative fashion based on this single sensor utility where in each iteration, a single sensor of node q is removed and this process is repeated until all of the sensors from node q are removed. The utility of node q with respect to node k's estimation problem in this iterative approach is then given as

$$U_{k_{-q}} = \sum_{m=1}^{M_q} U_{k_{-q_m}} \tag{24}$$

where m is the sensor index of node q defined similarly as in (1). In the iterative approach, a new optimal fall-back estimator must be calculated after each sensor removal which was found to have a computationally complexity of $O((M - m)^2)$ [15]. The overall computational complexity therefore would be $\sum_{m=1}^{M_q} O((M - m)^2)$. However using (21) the utility can be found with a single matrix inversion with a maximum computational complexity of $O(M_q^3)$. Since often $M >> M_q$, using this iterative approach is usually more computationally intensive than using (21).

[6] It should be noted that when a single channel is considered for removal then this becomes a scalar inversion, and (21) reduces to the formulation presented in [15].

3.2. Utility for node addition

For node addition, the utility of node q with respect to node k's estimation is defined as the decrease in MMSE when all sensors of node q are added to node k's estimation problem. We assume that the current estimator without node q, $\hat{\mathbf{W}}_{k_{-q}}$, is known which also implies that the current sensor signal correlation matrix, $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, and its inverse, $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, are known. The cross-correlation matrix between the sensor signals and the desired signal components, $\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k}$, is partitioned as

$$\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} = \begin{bmatrix} \mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} \\ \mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} \end{bmatrix} \tag{25}$$

where $\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} = E\{\mathbf{y}_q\bar{\mathbf{x}}_k^H\}$ is not included in the estimation problem and $\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} = E\{\mathbf{y}_{-q}\bar{\mathbf{x}}_k^H\}$ represents the current cross-correlation matrix.

The utility is defined identically as in the case of node removal, which is repeated here for convenience, as

$$\begin{aligned} U_{k_{-q}} &= J_{k_{-q}}(\hat{\mathbf{W}}_{k_{-q}}) - J_k(\hat{\mathbf{W}}_k) \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{-q}}\} \end{aligned} \tag{26}$$

which relies on the new estimator with the addition of node q, $\hat{\mathbf{W}}_k$. Unlike the node removal case, the statistics to estimate the contribution of node q are unknown at node k because no information is sent when the node is not connected to the network. To circumvent this limitation we presuppose that node q periodically sends part of its observations to node k from which the required statistics can be measured, but these are not included in the estimation at node k hence only an $(M - M_q) \times (M - M_q)$-dimensional inverse is taken. Notice that this makes the calculation of the utility when a node is added to the estimation substantially different than for the node removal case.

With the above mentioned strategy of periodically sending node q's statistics to node k, the sensor signal correlation matrix is partitioned the same as given in (16) however we would like to find a computationally efficient manner in calculating the utility without having to take the full inverse of $\mathbf{R}_{\mathbf{yy}}$ to compute $\hat{\mathbf{W}}_k$.

For the sake of an easy exposition we define two intermediate variables

$$\mathbf{\Gamma} = \mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_q} \tag{27}$$

$$\mathbf{\Sigma} = \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{y}_q}\mathbf{\Gamma} \tag{28}$$

that incorporate the statistics of the current connected nodes and those of node q. In Appendix C it is shown that the utility, when node q is added to node k's estimation using these two intermediate variables along with the previously defined notation, is given as

$$\boxed{U_{k_{-q}} = \mathrm{Tr}\{(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k})^H\mathbf{\Sigma}^{-1}(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k})\}} \tag{29}$$

where

$$\mathbf{U}_{k_{-q}} \triangleq (\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k})^H\mathbf{\Sigma}^{-1}(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}) \tag{30}$$

which is again a generalization of the work presented in [15]. Since $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ is already known from the current estimation, the computational complexity of finding $\mathbf{\Gamma}$ is $O((M - M_q)^2 M_q)$ and $\mathbf{\Sigma}$ relies on the inverse of an $M_q \times M_q$ matrix. Therefore the overall computationally complexity of finding the utility will be $O((M - M_q)^2 M_q + M_q^3)$. If we again compare this to a naive approach of calculating the utility, i.e., including node q's signals in the current estimation and taking a worst case scenario $O(M^3)$ to find the change in the MMSE, we see that (29) offers a substantial decrease in computational complexity for large M.

3.3. Definition of a common network-wide utility measure

The utility calculated with (21) and (29) gives the difference in the MMSE when a node is removed from or added to the network. However the utilities calculated at an individual node are biased to that nodes desired signals, i.e., the utility calculated for node k's signals at node k, $U_{k_{-k}}$, may differ significantly for another node, $U_{q_{-k}}$. This conflict in the utilities stems from the fact that each node estimates its own node-specific desired signals. This makes it difficult to quantify the network-wide utility of a node's sensor signals, i.e., a single utility measure that incorporates every node's estimation problem.

One approach could be to use the sum, $\sum_k U_{k_{-q}}$, to define the network-wide impact of node q's signals. First of all, this would require the computation of K^2 utility values to evaluate the network-wide utility of each node. Secondly, and more importantly, this measure is heavily biased towards estimation problems at nodes with a large signal power, as the MMSE directly depends on the signal power of the desired signal. The utility values corresponding to these estimation problems will dominate the summation.

We therefore propose scaling the utilities by means of (12) in which the utilities are now in terms of a virtual node s. This modifies the utilities of the nodes as if they were estimating the dry source signals which effectively removes the bias toward a single nodes desired signals resulting in a common utility-reference. The intuition behind this approach is that a reliable estimate of the dry source signal(s) also allows each node to compute a reliable estimate of their locally observed source signal(s), i.e., if a node's sensor signals have a large utility with respect to this dry source estimation problem, they will be important for every node-specific estimation problem too. This is because each node actually estimates node-specific scaled versions of the dry source signals.

For ease of exposition we assume that each node will scale its utilities as if it were estimating the unobservable dry source signals in $\mathbf{s}$. Note that the estimation of $\mathbf{s}$ is not possible in practice, as the cross-correlation matrix in (8) cannot be computed from the local sensor signals at node k for the case where $\bar{\mathbf{x}}_k = \mathbf{s}$. However, the remarkable aspect of this is that information about the dry source signals is not needed to calculate a node's utility with respect to it. We define the desired dry source signals in terms of a $Q \times Q$-dimensional identity matrix $\bar{\mathbf{A}}_s = \mathbf{I}_{Q\times Q}$ for a virtual node s so that

$$\begin{aligned} \bar{\mathbf{x}}_s &= \bar{\mathbf{A}}_s \mathbf{s} \\ &= \mathbf{s}. \end{aligned} \tag{31}$$

The MMSE cost at this virtual node is given as

$$\begin{aligned} J_s(\hat{\mathbf{W}}_s) &= E\{\|\bar{\mathbf{x}}_s - \hat{\mathbf{W}}_s^H \tilde{\mathbf{y}}\|^2\} \\ &= \mathrm{Tr}\{\mathbf{R}_{\bar{\mathbf{x}}_s\bar{\mathbf{x}}_s} - \mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_s}\hat{\mathbf{W}}_s\} \end{aligned} \tag{32}$$

where $\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_s} = E\{\mathbf{y}\bar{\mathbf{x}}_s^H\}$. For the sake of an easy exposition, but w.l.o.g, we assume that the dry source signals have also been power normalized to unity which, relying on the assumption that the signals are statistically independent, gives $\mathbf{R}_{\bar{\mathbf{x}}_s\bar{\mathbf{x}}_s} = E\{\mathbf{s}\mathbf{s}^H\} = \mathbf{I}$.

The utility is defined similarly to (14) where the utility of node k's signals with respect to node s's estimation problem is given by

$$U_{s_{-k}} = J_{s_{-k}}(\hat{\mathbf{W}}_{s_{-k}}) - J_s(\hat{\mathbf{W}}_s). \tag{33}$$

Using the relationship between steering matrices and the MMSE cost matrices, as given in (12), we have

$$\begin{aligned} U_{s_{-k}} &= \mathrm{Tr}\{\hat{\mathbf{J}}_{s_{-k}}\} - \mathrm{Tr}\{\hat{\mathbf{J}}_s\} \\ &= \mathrm{Tr}\{\bar{\mathbf{A}}_{ks}^{-H}(\hat{\mathbf{J}}_{k_{-k}} - \hat{\mathbf{J}}_k)\bar{\mathbf{A}}_{ks}^{-1}\} \\ &= \mathrm{Tr}\{(\hat{\mathbf{J}}_{k_{-k}} - \hat{\mathbf{J}}_k)\bar{\mathbf{A}}_{ks}^{-1}\bar{\mathbf{A}}_{ks}^{-H}\} \end{aligned} \tag{34}$$

where $\bar{\mathbf{A}}_{ks} = \mathbf{I}_k\bar{\mathbf{A}}_k^H$. Using this and the fact that $\mathbf{R}_{\bar{\mathbf{x}}_k\bar{\mathbf{x}}_k} = \bar{\mathbf{A}}_k\bar{\mathbf{A}}_k^H$, it is then shown that

$$\boxed{U_{s_{-k}} = \mathrm{Tr}\{\mathbf{U}_{k_{-k}}\mathbf{R}^{-1}_{\bar{\mathbf{x}}_k\bar{\mathbf{x}}_k}\}} \tag{35}$$

which can then be applied to both (23), and (30). Note that $\mathbf{R}_{\bar{\mathbf{x}}_k\bar{\mathbf{x}}_k}$ is a submatrix of $\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k}$ in (8) if the desired signal and the noise are uncorrelated (see also Section 6.1).

This definition of a common network-wide utility measure allows each node to track the network-wide utility of its own sensor signals. Furthermore, it provides a common reference such that the utilities computed at the different nodes can be easily compared with each other, or with a common threshold (see also Section 3.4).

3.4. Greedy centralized node selection

To allow maintaining a minimal network-wide estimation performance, we only remove a node if this removal does not result in a network-wide MMSE increase[7] of more that η, where η is a user-defined threshold, which can be adapted depending on the current MMSE. Similarly, we only add a node if this addition can guarantee a minimal increase in the network-wide MMSE, i.e., larger than η. It is noted that, since we now use a common network-wide utility measure, each node can use the same value for η. As an example in terms of a constraint, if the number of nodes in the network were to be limited so that 50% were removed, this η could be adjusted until this constraint was met.

[7]This is the MMSE with respect to the (virtual) estimation of the dry source signals.

To facilitate a distributed node selection algorithm, we would also like each node to independently decide whether it should add itself or remove itself from the network which requires that each node calculates the network-wide utility of its own signals. To this end, in the case of node addition, instead of node k broadcasting its signal periodically to the other nodes to measure its utility, we can assume that it does not transmit its signal in order to conserve energy but that it can still receive the $M - M_k$ broadcast signals from the other connected nodes in the network. In this case node k uses its own $\mathbf{y}_k$ sensor signals as well as the $M - M_k$ signals from the other nodes to compute its utility, $U_{k_{-k}}$, for node addition by means of (21) instead of (29) since it is able to calculate the full optimal estimator $\hat{\mathbf{W}}_k$.

In the node removal case, the selection process picks the node with the lowest utility below η and removes this from the network. In the case there are multiple nodes for which the utility value is smaller than η, a greedy choice is made, i.e., the node with the smallest utility is removed. The network continues this selection process until there are no more nodes whose utility fall below η. Likewise if a node is not connected to the network and its utility is greater than the threshold value, it is added to the network where again a greedy choice is made in case multiple nodes exceed the threshold.

Since the selection process is greedy we do not make any claims on optimality but argue that because of the prohibitive computational complexity of an exhaustive search, the utility based approach offers a safe bound on the impact on the network-wide performance (in terms of the MMSE of the dry source signal estimation) while being computationally efficient. The greedy centralized node selection is summarized in Table 1.

Remark 1. *Notice also that once a node is removed or added from or to the network the inverse sensor signal correlation matrix (17) must be recalculated which effects the utilities of the nodes, therefore we cannot predict future utility values.*

4. Distributed Adaptive Node-Specific Signal Estimation (DANSE)

In Sections 2 and 3 it was assumed that each node has access to all M signals of $\mathbf{y}$ to compute the optimal $\hat{\mathbf{W}}_k$ for estimating its node-specific desired signal components. In the distributed scenario, the goal of each node is to estimate its desired signal components as good as in the centralized scenario without each node having to broadcast all of its M_k signals to the other nodes. This can be accomplished by using the distributed adaptive node-specific signal estimation (DANSE) algorithm. In this section we provide a brief outline of the DANSE algorithm and the reader is referred to [8] for a more detailed discussion as well as convergence proofs.

In DANSE, node k broadcasts a compressed version of its sensor signals $\mathbf{z}_k = \mathbf{C}_k^H \mathbf{y}_k$ to the other nodes where $\mathbf{C}_k$ is an $M_k \times Q$ compression matrix which will be defined later (see (37)). This compresses the data transmitted from the individual nodes by a factor of $\frac{M_k}{Q}$. Note that the number of channels in $\mathbf{z}_k$ are chosen to equal Q, i.e., the dimension of $\bar{\mathbf{x}}_k$ which is required for DANSE to converge to the optimal estimators [8]. The DANSE algorithm updates the compression matrix $\mathbf{C}_k$ of each node in an iterative round-robin fashion. We introduce the index i to indicate the current iteration of the algorithm.

The estimator matrix $\mathbf{W}_k^i$ is partitioned to

$$\mathbf{W}_k^i = [\mathbf{W}_{k1}^{iT} \dots \mathbf{W}_{kK}^{iT}]^T \tag{36}$$

where $\mathbf{W}_{kk}^i$ is the partial estimator that node k applies to its own sensor signals, $\mathbf{y}_k$. This $\mathbf{W}_{kk}^i$ is then used as the compression matrix $\mathbf{C}_k$ to generate the $\mathbf{z}_k^i$ signal, i.e.,

$$\mathbf{z}_k^i = \mathbf{W}_{kk}^{iH} \mathbf{y}_k. \tag{37}$$

Note that $\mathbf{W}_{kk}^i$ is used as a partial estimator as well as a compression matrix.

In the DANSE algorithm node k has access to its own sensor signals, $\mathbf{y}_k$, and the $Q(K-1)$ broadcast signals, from other nodes given as

$$\mathbf{z}_{-k}^i = [\mathbf{z}_1^{iT} \dots \mathbf{z}_{k-1}^{iT} \dots \mathbf{z}_{k+1}^{iT} \dots \mathbf{z}_K^{iT}]^T \tag{38}$$

where the $-k$ subscript indicates that the broadcast signal, $\mathbf{z}_k^i$, of node k itself is not included.

Instead of decompressing each $\mathbf{z}_q$ (as received from node q) in (38), node k applies a $Q \times Q$ transformation matrix $\mathbf{G}_{kq}$ to each received signal, i.e., it effectively applies an estimation matrix in the form

$$\tilde{\mathbf{W}}_k^i = [(\mathbf{W}_{11}^i \mathbf{G}_{k1}^i)^T \dots (\mathbf{W}_{kK}^i \mathbf{G}_{kK}^i)^T]^T \tag{39}$$

where the $\mathbf{G}_{kq}^i$'s are stacked together in a matrix of the form $\mathbf{G}_k^i = [\mathbf{G}_{k1}^{iT} \dots \mathbf{G}_{kK}^{iT}]^T$. Note that since node k has access to its uncompressed sensor signals $\mathbf{y}_k$ it does not need to apply a transformation matrix as it does to the received $\mathbf{z}_{-k}$ signals. Since $\mathbf{G}_{kk}^i$ is then not explicitly defined for node k it can be set to an identity matrix so that (39) is

$$\tilde{\mathbf{W}}_k^i = [(\mathbf{W}_{11}^i \mathbf{G}_{k1}^i)^T \dots (\mathbf{W}_{kk}^i \mathbf{I}_{Q\times Q})^T \dots (\mathbf{W}_{kK}^i \mathbf{G}_{kK}^i)^T]^T. \tag{40}$$

The DANSE algorithm now performs an MMSE estimation at each node in a round robin fashion given as

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \hline \mathbf{G}_{-k}^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk}, \mathbf{G}_{-k}}{\arg\min} E\left\{ \left\| \bar{\mathbf{x}}_k - \begin{bmatrix} \mathbf{W}_{kk} \\ \hline \mathbf{G}_{-k} \end{bmatrix}^H \tilde{\mathbf{y}}_k^i \right\|_2^2 \right\} \tag{41}$$

where

$$\tilde{\mathbf{y}}_k^i = \begin{bmatrix} \mathbf{y}_k \\ \hline \mathbf{z}_{-k}^i \end{bmatrix} \tag{42}$$

and $\mathbf{G}_{-k}^{i+1}$ is $\mathbf{G}_k^{i+1}$ without $\mathbf{G}_{kk}^{i+1}$. The solution of (41) is given as

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \hline \mathbf{G}_{-k}^{i+1} \end{bmatrix} = (\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^i)^{-1} \mathbf{R}_{\tilde{\mathbf{y}}_k \bar{\mathbf{x}}_k}^i \tag{43}$$

where

$$\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^i = E\{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k^H\}, \quad \mathbf{R}_{\tilde{\mathbf{y}}_k \bar{\mathbf{x}}_k} = E\{\tilde{\mathbf{y}}_k \bar{\mathbf{x}}_k^H\}. \tag{44}$$

The MMSE estimate at node k is then given as the filtered combination of the nodes own sensor signals together with the received signals from other nodes $\mathbf{z}_{-k}$,

$$\tilde{\mathbf{x}}_k = \hat{\mathbf{W}}_{kk}^H \mathbf{y}_k + \sum_{l=1,l\neq k}^{K} \mathbf{G}_{kl}^H \mathbf{z}_l. \tag{45}$$

We define a block length B that represents the number of observations collected between two increments of the DANSE algorithm. The DANSE algorithm is summarized in Table 2 and Figure 2 gives a depiction of the DANSE algorithm in a network with three nodes, $K = 3$, and two broadcast signals per node composing $\mathbf{z}_k$ ($Q = 2$).

If $\mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$ is full rank, and $\bar{\mathbf{A}}_k$ is a full-rank $Q \times Q$, $\forall k \in K$, matrix then the DANSE algorithm converges for any initialization of its parameters to the centralized solution given in (8) [8]. After convergence, $i = \infty$, the estimator coefficients between node k and node q are related by

$$\tilde{\mathbf{W}}_k^\infty = \tilde{\mathbf{W}}_q^\infty (\mathbf{G}_{qk}^\infty)^{-1} \tag{46}$$

with

$$\begin{aligned} \mathbf{G}_{qk}^\infty &= \bar{\mathbf{A}}_k^{-H} \bar{\mathbf{A}}_q^H \\ &= \bar{\mathbf{A}}_{qk}. \end{aligned} \tag{47}$$

Remark 2. *Due to the iterative nature of the DANSE algorithm it may appear that the same sensor signals are broadcast multiple times. However, the iterations are spread out over time which means that different compressed versions of observations are broadcast at successive iterations in the algorithm. Therefore the nodes do not need to recompress and re-broadcast the same observations and so the processing in the different iterations is performed on different blocks of data. In Table 2 each iteration of DANSE uses different observations (the sample index is incremented based on the DANSE iteration index i).*

5. Distributed Computation of Utility Bounds

In the distributed scenario, nodes only have access to their own sensor signals and linearly compressed sensor signals from the other nodes, e.g., node k only has access to its own M_k sensor signals, $\mathbf{y}_k$, and the $Q(K-1)$ broadcast signals from the other nodes, $\mathbf{z}_{-k}$. Using these signals we would like each node to compute its own utility locally, $U_{s_{-k}}$, to determine if it should add itself to or remove itself from the network. For node addition, we assume that while a node is not transmitting it can still receive the other $\mathbf{z}_{-k}$ broadcast signals. For the sake of easy exposition, we assume that node k computes the utility $U_{k_{-k}}$ with respect to its own estimation problem, rather than $U_{s_{-k}}$ with respect to the dry source signals (see Section 3.3). However, everything in this section can easily be extended to also compute $U_{s_{-k}}$ by using the appropriate transformation given in (35).

The utility for node deletion given in (21) relies on the availability of $\mathbf{S}$, a sub-matrix of $\mathbf{R}_{\mathbf{yy}}^{-1}$, which is never available in the distributed case. Therefore, (21) cannot be used, and we need the original definition of the utility in

(14). This relies on the ability for a node to calculate its optimal fall-back estimator, e.g., k's fall-back estimator when removing itself from the network is $\hat{\mathbf{W}}_{k_{-k}}$. However, in the distributed scenario, the optimal estimator is found by iteratively passing information from one node to the next until the system reconverges. When node k is removed from the network this not only changes the partial estimator $\mathbf{W}_{kk}$ but also $\mathbf{G}_{-k}$ which both rely on the statistics of the other nodes. Therefore if node k were removed from the network, the fall-back estimator at node k, $\tilde{\mathbf{W}}_{k_{-k}}$ is initially sub-optimal and only becomes optimal once all of the nodes in the network have converged again. To avoid the explicit computation of the K different fall-back estimators for each possible node removal, we define $\overline{U}_{k_{-k}}$ that is based on (21) where $\mathbf{R}_{\mathbf{yy}}^{-1}$ is now replaced with $(\mathbf{R}^{i}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k})^{-1}$. In the case of node removal this $\overline{U}_{k_{-k}}$ will be shown to be an upper bound on the increase in the MMSE, i.e., the actual utility $U_{k_{-k}}$.

Likewise for node addition we will show that with a similar argument removed nodes are able to calculate $\overline{U}_{k_{-k}}$ which produces a utility lower bound on the actual utility $U_{k_{-k}}$, i.e., if the node adds itself to the existing network the actual decrease in MMSE after addition will be greater than that given by the utility. Therefore even though the exact utility cannot be computed, bounds for the change in the MMSE can be found for node removal and addition which will facilitate node selection.

5.1. Node removal : utility upper bound

Assuming the DANSE algorithm has converged[8] we define the following quantity for node k with respect to its node-specific estimation problem

$$\boxed{\overline{U}^{i}_{k_{-k}} = J_{k_{-k}}(\overline{\mathbf{W}}^{i+1}_{k_{-k}}) - J_k(\hat{\mathbf{W}}_k)} \tag{48}$$

where $\overline{\mathbf{W}}^{i+1}_{k}$ is

$$\overline{\mathbf{W}}^{i+1}_{k} = \begin{bmatrix} \mathbf{W}^{i+1}_{11}\mathbf{G}^{i+1}_{k1} \\ \vdots \\ \mathbf{W}^{i+1}_{kK}\mathbf{G}^{i+1}_{kK} \end{bmatrix} \tag{49}$$

and $\overline{\mathbf{W}}^{i+1}_{k_{-k}}$ is equal to (49) with $\mathbf{W}^{i+1}_{kk}\mathbf{G}^{i+1}_{kk}$ removed.

It is noted that (48) does not represent the true utility of node k since, in principle, once the node is removed the other nodes must update their local estimator parameters until the DANSE algorithm has reconverged to the re-optimized fall-back estimator, i.e., $\overline{\mathbf{W}}^{\infty}_{k_{-k}} = \tilde{\mathbf{W}}^{\infty}_{k_{-k}} = \hat{\mathbf{W}}_{k_{-k}}$ where $\tilde{\mathbf{W}}^{\infty}_{k_{-k}}$ is equivalent to (39) after convergence without node k's signals and $\hat{\mathbf{W}}_{k_{-k}}$ represents the optimal estimator without node k's signals.

Due to the convergence of DANSE, the cost function then decreases until

$$J_{k_{-k}}(\overline{\mathbf{W}}^{\infty}_{k_{-k}}) = J_{k_{-k}}(\hat{\mathbf{W}}_{k_{-k}}). \tag{50}$$

[8] For simulations presented in Section 6 convergence is reached after each node updates its local parameters 5 times, i.e., $5\times K$ DANSE iterations.

Since $\hat{\mathbf{W}}_{k_{-k}}$ minimizes $J_{k_{-k}}$, the $\overline{U}^i_{k_{-k}}$ using the sub-optimal estimator $\overline{\mathbf{W}}^{i+1}_{k_{-k}}$ is an upper bound of the increase in MMSE, i.e.,

$$\begin{aligned} J_{k_{-k}}(\overline{\mathbf{W}}^{\infty}_{k_{-k}}) - J_k(\hat{\mathbf{W}}_k) &\leq J_{k_{-k}}(\overline{\mathbf{W}}^{i+1}_{k_{-k}}) - J_k(\hat{\mathbf{W}}_k) \\ \overline{U}^{\infty}_{k_{-k}} &\leq \overline{U}^i_{k_{-k}}. \end{aligned} \tag{51}$$

The utility upper bound, $\overline{U}^i_{k_{-k}}$, can be efficiently computed by means of (21) based on $\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$

$$\boxed{\overline{U}^i_{k_{-k}} = \mathrm{Tr}\{\hat{\mathbf{W}}^H_{kk}(\mathbf{R}^i_{\mathbf{y}_k\mathbf{y}_k})^{-1}\hat{\mathbf{W}}_{kk}\}} \tag{52}$$

where $\mathbf{R}^i_{\mathbf{y}_k\mathbf{y}_k}$ is the part of $\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$ pertaining to node k's signals only. A corresponding upper bound for the network-wide utility $\overline{U}_{s_{-k}}$ corresponding to the dry source signals can also be computed similarly to (35).

With this, node k can decide to remove itself from the network knowing the maximum impact it will have in terms of increase in MMSE. It should also be noted that $\overline{\mathbf{W}}^{i+1}_{k_{-k}}$ is not explicitly computed when calculating the utility upper bound, and only exists once the node has been removed from the network.

5.2. Node addition : utility lower bound

As in the centralized scenario, the addition of a node is substantially different from node removal due to the fact that node k does not broadcast $\mathbf{z}_k$ when not included in the network. In the distributed scenario when a node is not connected to the network we assume that it is still able to receive signals, possibly awaking periodically to judge its current importance to the network estimation.

In the sequel, we assume that node k still performs an estimation of its own desired signals using $\mathbf{z}_{-k}$ and its own $\mathbf{y}_k$ signals. In this case, we show that the node is able to determine a utility lower bound with the same formula that is used for node removal (52) without having to broadcast its $\mathbf{z}_k$ signal.

Assuming the DANSE algorithm has converged with node k not broadcasting its signals, we define the following utility for node k with respect to node k's estimation problem,

$$\boxed{\overline{U}^i_{k_{-k}} = J_{k_{-k}}(\hat{\mathbf{W}}_{k_{-k}}) - J_k(\overline{\mathbf{W}}^{i+1}_k)} \tag{53}$$

where again $\hat{\mathbf{W}}_{k_{-k}}$ represents the optimal estimator without node k's signals and $\overline{\mathbf{W}}^{i+1}_k$ is given in (49). Notice that the cost function with node k's signals removed, $J_{k_{-k}}(\hat{\mathbf{W}}_{k_{-k}})$, is currently minimized as all of the nodes in the network have performed their estimation without node k's signals using the DANSE algorithm.

Assuming that node k would include itself in the network, then due to the convergence of the DANSE algorithm, the cost decreases to

$$J_k(\hat{\mathbf{W}}_k) = J_k(\overline{\mathbf{W}}^{\infty}_k) \tag{54}$$

where again the estimators have converged to their optimal values. The $\overline{U}^i_{k_{-k}}$ using the sub-optimal estimator therefore

is a lower bound for the MMSE decrease when a node is added to the network, i.e.,

$$J_{k_-k}(\hat{\mathbf{W}}_{k_-k}) - J_k(\overline{\mathbf{W}}_k^{\infty}) \geq J_{k_-k}(\hat{\mathbf{W}}_{k_-k}) - J_k(\overline{\mathbf{W}}_k^{i+1})$$
$$\overline{U}_{k_-k}^{\infty} \geq \overline{U}_{k_-k}^{i}. \tag{55}$$

Therefore if node k was to add itself to the network, i.e., begin broadcasting its $\mathbf{z}_k$, its MMSE will decrease by at least the utility

$$\boxed{\overline{U}_{k_-k}^{i} = \mathrm{Tr}\{\overline{\mathbf{W}}_{kk}^{iH}(\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}^{i})^{-1}\overline{\mathbf{W}}_{kk}^{i}\}}. \tag{56}$$

It should be noted that since node k uses its $\mathbf{y}_k$ in its current estimation it does not need to rely on (29) to calculate its utility. Therefore by using (56), which is computationally more efficient, we limit the computational power and memory requirement of the node, but bearing in mind that the calculation of the utility from either equation would be equivalent.

5.3. Greedy distributed node selection

In the distributed scenario the same method for adding and removing nodes can be used as in the centralized case (Table 1). However instead of calculating the exact utilities, only upper and lower bounds can be computed. The greedy selection procedure in Section 3.4 is therefore modified to take the utility bounds into consideration. During estimation nodes will calculate their utility bounds based on (52) and (56) and scale them to the common dry source reference using (35). A distributed version of the node selection algorithm is given in Table 3. Since the nodes compute upper and lower bounds, we know that it is safe to remove or add nodes, i.e., without risking an MMSE increase or decrease that is larger than η.

6. Simulations

6.1. Estimation of signal statistics

For the computation of the utility bounds and the DANSE updates it was implicitly assumed that the second-order signal statistics are known throughout the estimation procedure. However in real-time applications there is normally a finite observation window where estimation of the signals statistics is done by time averaging with the collected observations and exploiting the assumed behavior of the signals such as short-term stationarity and ergodicity.

Let $\tilde{\mathbf{y}}_k[t]$, (42), denote the observations of $\tilde{\mathbf{y}}_k$ collected at time t at node k. Estimating the so-called "signal+noise" correlation matrix, $\mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$, is typically done by time averaging the collected observations with a forgetting factor $0 < \lambda < 1$ [5],[26, § 9],[29], i.e.,

$$\mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}[t] = \lambda\mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}[t-1] + (1-\lambda)\tilde{\mathbf{y}}_k[t]\tilde{\mathbf{y}}_k[t]^H. \tag{57}$$

Estimating the desired signal correlation matrix, $\mathbf{R}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}$, is not as straightforward because the desired signal components are collected with the addition of noise. If the desired signals are assumed to have on-off behavior,[9] meaning

[9] This type of behavior is typical for speech enhancement applications where there is assumed to be pauses in the speech signal.

that there are periods when only noise is present and periods when there is desired signal as well as noise, the "noise" and "signal+noise" statistics may be gathered separately.

During periods when the desired signals plus noise are present $\mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$ is computed by means of (57). Likewise during noise-only periods the received signals are placed into a "noise-only" correlation matrix given by

$$\mathbf{R}_{\tilde{\mathbf{v}}_k\tilde{\mathbf{v}}_k}[t] = \lambda\mathbf{R}_{\tilde{\mathbf{v}}_k\tilde{\mathbf{v}}_k}[t-1] + (1-\lambda)\tilde{\mathbf{v}}_k[t]\tilde{\mathbf{v}}_k[t]^H \tag{58}$$

where $\mathbf{v}_k$ is defined in (3) and $\tilde{\mathbf{v}}_k$ refers to the corresponding noise component in $\tilde{\mathbf{y}}_k$, as defined in (42).

Usually the desired signals and noise are assumed to be un-correlated and statistically independent, therefore a desired signal correlation matrix may be estimated by subtracting the "signal+noise" by the "noise-only" correlation matrix, i.e.,

$$\mathbf{R}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k} = \mathbf{R}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k} - \mathbf{R}_{\tilde{\mathbf{v}}_k\tilde{\mathbf{v}}_k}. \tag{59}$$

Subsequentially the cross correlation matrix, $\mathbf{R}_{\tilde{\mathbf{y}}_k\bar{\mathbf{x}}_k}$, can be given as

$$\mathbf{R}_{\tilde{\mathbf{y}}_k\bar{\mathbf{x}}_k} = E\{\tilde{\mathbf{y}}_k\bar{\mathbf{x}}_k^H\} \tag{60}$$

which, using the assumption that the desired signals and noise are un-correlated, may be given as

$$\mathbf{R}_{\tilde{\mathbf{y}}_k\bar{\mathbf{x}}_k} = \mathbf{R}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}\mathbf{E} \tag{61}$$

where $\mathbf{E}$ is an $M_k + Q(K-1) \times Q$-dimensional matrix that has a $Q \times Q$-dimensional identity matrix corresponding to the desired signal components and 0 otherwise, i.e.,

$$\mathbf{E} = \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}. \tag{62}$$

Note that this is just one possible strategy to estimate $\mathbf{R}_{\tilde{\mathbf{y}}_k\bar{\mathbf{x}}_k}$. Other strategies may involve using training sequences, or only considering quasi-static scenarios [8].

6.2. Batch mode

In this section we demonstrate the greedy utility based node selection process in batch mode which means that all iterations of DANSE were performed on data obtained from the entire length of the signals. In Section 6.3 an adaptive implementation with moving sources is presented where, instead of processing the entire length of the signal, shorter blocks are processed which change the utilities throughout the simulation.

Although batch mode is not a practical implementation, a batch mode simulation gives a reasonable view on the performance limits of the algorithm. The greedy node selection process used a utility bound based on (52), (56) and (35). The dimension of the desired signal space, ($J = Q$), was varied depending on the simulation, where each desired source consisted of 10000 samples generated from a uniformly distributed random process on the interval [-0.5 0.5]. The coefficients of $\mathbf{A}_k$ were generated by a uniform process on the unit interval. A single spatially located white noise

source was added where the signal was generated with a similar process as the dry source signal and scaled with a random number in (0,1] for each node. Uncorrelated white noise, representative of sensor noise, that was half the average power of the desired signals, was added to each sensor.

As each node in the network estimates a different signal, we will visualize the network-wide performance at one particular 'visualization node'. We assume[10] that $\bar{\mathbf{A}}_k = \mathbf{I}$ in (6) at this node, i.e., the node estimates the dry source signals. This allows us to compare the MMSE increase/decrease compared to the network-wide utility bounds which are also referenced to the dry source signal estimation problem (see Section 3.3). We also constrain the algorithm such that this visualization node is never removed from the network. It is also ensured that the order of removal is the same in the centralized and the distributed so that the dry source MSE can be compared between the two cases. However it should be noted that this is not always the case (see Figure 9), since due the greedy selection even after the removal of a single node, the next node removal may be different.

In the distributed scenario, the DANSE parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_{-k}$ $\forall k \in \{1 \dots K\}$, were updated in a round-robin fashion for $5 \times K$ iterations, i.e., 5 updates per node, which was deemed sufficient for network convergence, i.e., the difference in cost between iterations was below machine precision. After DANSE had reached a steady state the utility bounds pertaining to the dry source signal, $\overline{U}_{s_{-k}}$, were calculated. The bounds for the MSE threshold η were predefined before the start of the simulations and only one node at a time was allowed to be removed or added to the network, where the DANSE parameters were allowed to reconverge before the selection process resumed.

The centralized and distributed selection procedures for removal were compared as shown in Figure 3. There are $K = 20$ nodes in the system each with 5 sensors ($M = 100$), and $Q = 3$. The figure on the left is the centralized selection process, where the full matrix $\mathbf{R}_{\mathbf{yy}}^{-1}$ can be used in (21). The dashed-lines indicate how much the dry source MSE increases after the removal of a node, i.e., $J_s(\hat{\mathbf{W}}_s) + U_{s_{-k}}$.

The figure on the right is the DANSE algorithm that performs the same removal process where (52) is used and scaled by (35). The dotted lines indicate the maximum increase a node will have on the dry source MSE after removal, i.e., $J_s(\hat{\mathbf{W}}_s) + \overline{U}^i_{s_{-k}}$. Each increment on the horizontal axis in the distributed scenario corresponds to a full iteration cycle of DANSE, i.e., every node has updated its node-specific parameters ($\mathbf{W}_{kk}$ and $\mathbf{G}_{-k}$) once. It was observed in [8] that convergence of the node-specific parameters occurs after each node has updated them 5 times. Therefore each DANSE cycle in Figure 3 corresponds to each node in the network updating their node-specific parameters once ($i = 20$ DANSE iterations) with a node being removed after each node has updated its node-specific parameters 5 times ($i = 20 \times 5$ DANSE iterations).

During the node selection process the utility upper bound and centralized utility were added to the current dry source MMSE in order to observe the increase in MMSE after node removal. This is indicated by the dotted and dashed lines. Notice the large decrease in MSE in the first few iterations of the DANSE algorithm compared to the subsequent reconvergence iteration when a node is removed from the network. This is due to the random initialization

[10] Node that this only for illustrative purposes since $\bar{\mathbf{A}}_k = \mathbf{I}$ usually does not occur in practice.

of the parameters at the beginning of the DANSE algorithm. Once the DANSE algorithm has converged and a node is removed, the sub-optimal estimators lie close to the new optimal estimators as shown in the relatively small decrease in MSE after the first set of iterations.

Figure 4 shows a magnified view of the selection process between DANSE iterations 15-21 of Figure 3. The utility upper bound lies above the utility that would have been found if the optimal estimator would have been used. Again we see a relatively small decrease in MSE when compared to the original initialization of the DANSE algorithm.

The next simulation contained $K = 5$ nodes in the system each with 20 sensor signals ($M = 100$) and $Q = 3$, i.e., the network loses many more sensor signals per node removal than in the previously simulated network. This normally has a larger impact on the utility bound, as the fall-back estimators must reconverge from a larger difference. Figure 5 shows the increase in MSE with the utility bound during the selection process. Notice that the utility bounds are less tight than in the previous network ($K = 20$, $M = 100$). This is because there are many more degrees of freedom in the DANSE-parameters at the other nodes. As the utility bound does not take the future DANSE updates of these parameters at other nodes into account, there is a larger gap between the centralized utility and the utility upper bound computed in the distributed case.

Due to the node removal case calculating a utility upper bound, there are times when the node selection fails to remove nodes that would have been removed had the optimal utility been used for node selection. Using the previous simulated network the value of η was adjusted so that this type of failure in the node selection process does occur. In Figure 6, the value of η and both the centralized utility and utility bounds were added to the current MSE to observe the impact of node removal. In this scenario, the centralized utility is below η, however the utility bound falls above η and so the node is not considered for removal. While this does not have a negative effect on the estimation, i.e., the MSE stays at a lower value, it prolongs the usage of a node that would have otherwise had its transmission capabilities turned-off possibly shortening the lifetime of the network.

For node addition a network was constructed with $K = 20$ nodes each with 5 sensor signals ($M = 100$) and $Q = 3$. At the beginning of the selection process a single node broadcast its $\mathbf{z}_k$ signal and other nodes used this signal along with their local signals to determine their utility. The utility threshold, η, was set to ∞ so that all of the available nodes would eventually add themselves to the network. In the centralized case, the utility was found using (35) and was then subtracted from the current MSE to find the new MSE after node addition, i.e., $J_{s_{-k}}(\hat{\mathbf{W}}_s) - U_{s_{-k}}$ and is represented by the dashed line. In the distributed case, represented by the dotted lines the utility was calculated by means of (52) and subtracted from the current MSE, i.e., $J_{s_{-k}}(\hat{\mathbf{W}}_s) - \overline{U}^i_{s_{-k}}$. It should be noted that since the utility was subtracted from the current MMSE, the true MSE after addition will be lower than that calculated MSE which uses the sub-optimal estimator as shown in (55). In Figure 7 the utility bound is shown to provide at least a minimal decrease in the system, i.e., after convergence in the distributed scenario the MSE is lower than that given by the utility bound.

6.3. Adaptive implementation

For the adaptive implementation a simulated environment, depicted in Figure 8, is considered where there are two moving desired source signals ($Q = 2$). The desired source signals (□), which are generated from a uniformly distributed random process on the interval [-0.5 0.5], follow paths indicated by the L-shaped dashed lines. The desired source signals move at a speed of 0.3 m/s and stop for 3 seconds at each corner. After the desired source signals reach the end of the path they follow the same route to their starting point. This movement repeats until the end of the simulation. Five white Gaussian noise sources ⧫, which are generated from the same process as the desired source signals, are present and an uncorrelated white Gaussian noise that is 5% of the average power of the noise sources is added to each sensor observation. There are $K = 30$ nodes each with 5 sensors (∘) so that the total number of sensor observations are $M = 150$.

The individual sensor measurements originating from the desired signal and noise sources are attenuated and summed at each sensor. The attenuation factor is given as $\frac{1}{r}$ where r denotes the distance from the signal source to the sensor. We assume that the desired source signals and noise statistics are estimated at each node based on (57)-(58) where the correlation matrices are updated with a forgetting factor of $\lambda = 0.97$ and the sensors observe their signals at a sampling frequency of $f_s = 8$kHz.

The desired source signals are stationary for the first 5 seconds of the simulation in order to populate the necessary signal statistics and all nodes are considered active during this time. After this initialization the node selection process is started in order to remove and add nodes depending on their utility bounds when compared to the predefined threshold η. After the addition or removal of a node from the system a full DANSE cycle occurs, i.e., all nodes update once, before the selection algorithm begins again. This is done in order to allow the DANSE algorithm to reconverge after the addition or removal process.

Figure 9 compares the MSE when no nodes are removed to the centralized and distributed node selection process, as well as the number of active nodes in the system. The selection of nodes in the centralized and distributed scenario do not follow the same order due to the use of an upper and lower bound and due to the limited tracking capabilities of DANSE, which may generate errors in the utility bounds.

However the utility bound is able to limit the effect on the MSE similarly to that in the centralized scenario. There are even times that the MSE and number of active nodes in the distributed scenario are better than that of the centralized scenario which is possible due to the fact that greedy node selection is often suboptimal. The total number of active nodes at any one time during the centralized solution with no node removal as well as the centralized and distributed selection process is given in the bottom plot of Figure 9.

The scenario is shown in Figure 10 during various times (0s, 22s, 45s, 90s) of the node selection process. The active nodes are shown in blue and the nodes that are only receiving signals from the other nodes and not transmitting their $\mathbf{z}_k$ signals are shown in red. At 0 seconds no nodes are removed from the network which also indicates that the system is using the maximal amount of power. After 22 seconds there has already been a large reduction in the

amount of active nodes in the network. It should be noted that this is dependent on η and could be adjusted to fit the desired scenario.

7. Conclusions

In this paper we have introduced the utility as a means to facilitate node selection in a distributed wireless sensor network that performs node-specific signal estimation. This was accomplished by using the convergence and optimality properties of the DANSE algorithm in unison with an MSE threshold and a greedy selection process. While the distributed utility bounds were shown to be sub-optimal they were successfully used to limit the MMSE increase or decrease during node selection. The centralized and distributed node selection were compared to one another and it was shown that the utility bounds offers an efficient way to perform node selection while still allowing to control the MSE performance. Simulation results using the distributed node selection process often have a similar performance to the centralized node selection process and show that there are significant power savings in the network while only slightly effecting the MSE.

Appendix A.

The cost function of node q evaluated with the node-specific linear MMSE estimator, $\hat{\mathbf{W}}_q$, is given as

$$J_q(\hat{\mathbf{W}}_q) = E\{\|\bar{\mathbf{x}}_q - \hat{\mathbf{W}}_q^H \mathbf{y}\|_2^2\}. \tag{A.1}$$

Using this optimal estimator, the MMSE cost matrix for node q is given as

$$\hat{\mathbf{J}}_q = \mathbf{R}_{\bar{\mathbf{x}}_q \bar{\mathbf{x}}_q} - \mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_q}^H \hat{\mathbf{W}}_q \tag{A.2}$$

where $\mathbf{R}_{\bar{\mathbf{x}}_q \bar{\mathbf{x}}_q} = E\{\bar{\mathbf{x}}_q \bar{\mathbf{x}}_q^H\}$ is now the node q's desired signal correlation matrix.

The optimal estimators are related to one another by (13), which when used in (A.2) produces

$$\hat{\mathbf{J}}_q = \mathbf{R}_{\bar{\mathbf{x}}_q \bar{\mathbf{x}}_q} - \mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_q}^H \hat{\mathbf{W}}_k \bar{\mathbf{A}}_{qk}. \tag{A.3}$$

By expanding the desired components of node q into its complex-valued steering matrix and source signal vector (A.3) is then given as

$$\hat{\mathbf{J}}_q = \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{s}^H\} \bar{\mathbf{A}}_q^H - \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{y}^H\} \hat{\mathbf{W}}_k \bar{\mathbf{A}}_{qk}. \tag{A.4}$$

Now the product of $\bar{\mathbf{A}}_{qk}^{-H} \hat{\mathbf{J}}_q \bar{\mathbf{A}}_{qk}^{-1}$ is given as

$$\begin{aligned}
\bar{\mathbf{A}}_{qk}^{-H} \hat{\mathbf{J}}_q \bar{\mathbf{A}}_{qk}^{-1} &= \bar{\mathbf{A}}_{qk}^{-H} \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{s}^H\} \bar{\mathbf{A}}_q^H - \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{y}^H\} \hat{\mathbf{W}}_k \bar{\mathbf{A}}_{qk} \bar{\mathbf{A}}_{qk}^{-1} \\
&= \bar{\mathbf{A}}_k \bar{\mathbf{A}}_q^{-1} \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{s}^H\} \bar{\mathbf{A}}_q^H \bar{\mathbf{A}}_q^{-H} \bar{\mathbf{A}}_k^H - \bar{\mathbf{A}}_k \bar{\mathbf{A}}_q^{-1} \bar{\mathbf{A}}_q E\{\mathbf{s}\mathbf{y}^H\} \hat{\mathbf{W}}_k \\
&= \bar{\mathbf{A}}_k E\{\mathbf{s}\mathbf{s}^H\} \bar{\mathbf{A}}_k^H - \bar{\mathbf{A}}_k E\{\mathbf{s}\mathbf{y}^H\} \hat{\mathbf{W}}_k \\
&= \hat{\mathbf{J}}_k \qquad \text{(A.5)} \\
& \qquad \text{(A.6)}
\end{aligned}$$

which shows the equivalence stated in (12).

Appendix B.

For ease of exposition we re-iterate the block partitioning of the inverse "signal+noise" correlation matrix as

$$\mathbf{R}_{\mathbf{y}\mathbf{y}}^{-1} = \begin{bmatrix} \mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C} \end{bmatrix} \tag{B.1}$$

and of the cross-correlation matrix $\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k}$ as

$$\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} = \begin{bmatrix} \mathbf{R}_{\mathbf{y}_q \bar{\mathbf{x}}_k} \\ \mathbf{R}_{\mathbf{y}_{-q} \bar{\mathbf{x}}_k} \end{bmatrix} \tag{B.2}$$

where $\mathbf{R}_{\mathbf{y}_q \bar{\mathbf{x}}_k} = E\{\mathbf{y}_q \bar{\mathbf{x}}_k^H\}$ and $\mathbf{R}_{\mathbf{y}_{-q} \bar{\mathbf{x}}_k} = E\{\mathbf{y}_{-q} \bar{\mathbf{x}}_k^H\}$. The estimator without node q's signals is given as

$$\hat{\mathbf{W}}_{k_{-q}} = \mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}^{-1} \mathbf{R}_{\mathbf{y}_{-q} \bar{\mathbf{x}}_k}. \tag{B.3}$$

Now using the previously defined inverse correlation matrix (18)

$$\begin{aligned}\hat{\mathbf{W}}_{k_{-q}} &= (\mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V})\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} \\ &= \mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}.\end{aligned} \tag{B.4}$$

The current estimator values are given as

$$\hat{\mathbf{W}}_k = \begin{bmatrix}\hat{\mathbf{W}}_{k_q} \\ \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}\end{bmatrix} = \begin{bmatrix}\mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C}\end{bmatrix}\begin{bmatrix}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} \\ \mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\end{bmatrix}. \tag{B.5}$$

Now using (B.5) and re-arranging the expression for $\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}$ we have

$$\begin{aligned}\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} &= \mathbf{V}^H\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} + \mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} \\ \mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} &= \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k}.\end{aligned} \tag{B.6}$$

Using (B.6) into (B.4) produces

$$\hat{\mathbf{W}}_{k_{-q}} = \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}). \tag{B.7}$$

Now using the fact that $\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q} = \mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} + \mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}$, (B.7) may by represented as (20), i.e.,

$$\hat{\mathbf{W}}_{k_{-q}} = \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q}. \tag{B.8}$$

Now using the optimal fall-back estimator (20) we are able to calculate the utility given in (21). Using (B.8) and the definition of the utility (15) gives

$$\begin{aligned}U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{-q}}\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{V}^H\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q}\}.\end{aligned} \tag{B.9}$$

Block partitioning the first element in the trace of (B.9) gives

$$\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_k = \left[\begin{array}{c|c} \mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} & \mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\end{array}\right]^H\begin{bmatrix}\hat{\mathbf{W}}_{k_q} \\ \hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}\end{bmatrix} \tag{B.10}$$

which expands the utility to

$$\begin{aligned}U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_q} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{V}^H\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q}\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\hat{\mathbf{W}}_{k_q} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{V}^H\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q}\}.\end{aligned} \tag{B.11}$$

Now using (B.5) we have

$$\begin{aligned}\hat{\mathbf{W}}_{k_q} &= \mathbf{S}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} + \mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} \\ \mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k} &= \hat{\mathbf{W}}^H_{k_q}\mathbf{S}^{-1} - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{V}^H\mathbf{S}^{-1}\end{aligned} \tag{B.12}$$

which when used with the previous result gives the utility

$$U_{k_{-q}} = \mathrm{Tr}\{\hat{\mathbf{W}}^H_{k_q}\mathbf{S}^{-1}\hat{\mathbf{W}}_{k_q}\}. \tag{B.13}$$

Appendix C.

We partition the inverse correlation matrix using the Woodbury identity and use two intermediate variables, $\mathbf{\Gamma} = \mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}\mathbf{R}_{\mathbf{y}_{-\mathbf{q}}\mathbf{y}_{\mathbf{q}}}$ and $\mathbf{\Sigma} = \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} - \mathbf{R}^H_{\mathbf{y}_{-\mathbf{q}}\mathbf{y}_{\mathbf{q}}}\mathbf{\Gamma}$,

$$\mathbf{R}^{-1}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{\Sigma}^{-1} & -\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H \\ -\mathbf{\Gamma}\mathbf{\Sigma}^{-1} & \mathbf{C} \end{bmatrix}. \tag{C.1}$$

We first expand (26) using the definition of the optimal estimator (8) to

$$U_{k_{-q}} = \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\mathbf{R}^{-1}_{\mathbf{yy}}\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\}. \tag{C.2}$$

Using the previously defined intermediate variables we see that

$$\mathbf{R}^H_{\mathbf{y}\bar{\mathbf{x}}_k}\mathbf{R}^{-1}_{\mathbf{yy}}\mathbf{R}_{\mathbf{y}\bar{\mathbf{x}}_k} = \mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}. \tag{C.3}$$

Now using (18) and (C.1), $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ is given as

$$\begin{aligned} \mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} &= \mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V} \\ &= \mathbf{C} - \mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H. \end{aligned} \tag{C.4}$$

Now combining (C.3), (C.4) and (C.2) produces

$$\begin{aligned} U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{R}^H_{\mathbf{y}_q\bar{\mathbf{x}}_k}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k} - \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} + \mathbf{R}^H_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k}\} \\ &= \mathrm{Tr}\{(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k})^H\mathbf{\Sigma}^{-1}(\mathbf{R}_{\mathbf{y}_q\bar{\mathbf{x}}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\bar{\mathbf{x}}_k})\}. \end{aligned} \tag{C.5}$$

References

[1] S. Gajjar, S. Pradhan, K. Dasgupta, Wireless sensor network: Application led research perspective, in: Recent Advances in Intelligent Computational Systems (RAICS '11), Trivandarum, Kerala, India, pp. 025–030.

[2] A. Alemdar, M. Ibnkahla, Wireless sensor networks: Applications and challenges, in: 9th Int. Symp. on Signal Process. and Its Applications (ISSPA '07), Sharjah, United Arab Emirates, pp. 1–6.

[3] A. Bertrand, Applications and trends in wireless acoustic sensor networks: A signal processing perspective, in: 18th IEEE Symp. on Communications and Vehicular Technology in the Benelux (SCVT '11), Ghent, Belgium, pp. 1–6.

[4] D. Puccinelli, M. Haenggi, Wireless sensor networks: applications and challenges of ubiquitous sensing, IEEE Circuits and Systems Mag. 5 (2005) 19–31.

[5] A. Bertrand, M. Moonen, Robust distributed noise reduction in hearing aids with external acoustic sensor nodes, EURASIP Journal on Advances in Signal Processing 2009 (2009) 14.

[6] T. C. Lawin-Ore, S. Doclo, Analysis of rate constraints for MWF-based noise reduction in acoustic sensor networks, in: Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP'11), Prague, Czech Republic.

[7] S. Markovich, Golan, S. Gannot, I. Cohen, A reduced bandwidth binaural MVDR beamformer, in: Proc. Int. Workshop on Acoust. Echo and Noise Contr. (IWAENC'10), Tel-Aviv, Israel.

[8] A. Bertrand, M. Moonen, Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating, IEEE Trans. Signal Process. 58 (2010) 5277–5291.

[9] A. Bertrand, M. Moonen, Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology, IEEE Trans. on Signal Process. 59 (2011) 2196–2210.

[10] H. Rowaihy, S. Eswaran, M. Johnson, D. Verma, A. Bar-noy, T. Brown, A survey of sensor selection schemes in wireless sensor networks, in: SPIE Defense and Security Symp. Conf. on Unattended Ground, Sea, and Air Sensor Technologies and Applications IX '07, Baltimore, MD., USA.

[11] S. Joshi, S. Boyd, Sensor selection via convex optimization, IEEE Trans. on Signal Process. 57 (2009) 451–462.

[12] M. Shamaiah, S. Banerjee, H. Vikalo, Greedy sensor selection: Leveraging submodularity, in: 49th IEEE Conf. on Decision and Control (CDC '10), Atlanta, GA., USA, pp. 2572–2577.

[13] D. Golovin, M. Faulkner, A. Krause, Online distributed sensor selection, in: Proc. of the 9th Int. Symp. on Information Processing in Sensor Networks (IPSN '10), Stockholm, Sweden, pp. 220–231.

[14] G. Thatte, U. Mitra, Sensor selection and power allocation for distributed estimation in sensor networks: Beyond the star topology, IEEE Trans. Signal Process. 56 (2008) 2649–2661.

[15] A. Bertrand, M. Moonen, Efficient sensor subset selection and link failure response for linear MMSE signal estimation in wireless sensor networks, in: Proc. of the European Signal Process. Conf. (EUSIPCO '10), Aalborg, Denmark, pp. 1092–1096.

[16] A. Bertrand, J. Szurley, P. Ruckebusch, I. Moerman, M. Moonen, Efficient calculation of sensor utility and sensor removal in wireless sensor networks for adaptive signal estimation and beamforming, IEEE Trans. Signal Process. 60 (2012) 5857–5869.

[17] Z. Quan, W. Kaiser, A. H. Sayed, Innovations diffusion: A spatial sampling scheme for distributed estimation and detection, IEEE Trans. Signal Process. 57 (2009) 738–751.

[18] Y.-H. Chang, X. Yu, Reduced-rank antenna selection for MIMO DS-CDMA channels, in: IEEE Int. Symp. on Circuits and Systems (ISCAS '05), volume 2, Kobe, Japan, pp. 1730–1733.

[19] H. Godrich, A. Petropulu, H. Poor, Sensor selection in distributed multiple-radar architectures for localization: A knapsack problem formulation, IEEE Trans. Signal Process. 60 (2012) 247–260.

[20] H. Godrich, A. Petropulu, H. V. Poor, A combinatorial optimization framework for subset selection in distributed multiple-radar architectures, in: Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP '11), Prague, Czech Republic, pp. 2796–2799.

[21] F. Bian, D. Kempe, R. Govindan, Utility-based sensor selection, in: The 5th Int. Conf. on Information Process. in Sensor Networks (IPSN '06), Nashville, TN., USA, pp. 11–18.

[22] A. Das, D. Kempe, Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection, in: Proc. of the 28th Int. Conf. on Machine Learning (ICML '11), Bellevue, WA., USA, pp. 1057–1064.

[23] S. S. Haykin, Adaptive filter theory, Prentice-Hall information and system sciences series, Prentice Hall, 2002.

[24] X. Li, Blind channel estimation and equalization in wireless sensor networks based on correlations among sensors, IEEE Trans. Signal Process. 53 (2005) 1511–1519.

[25] B. Cornelis, M. Moonen, J. Wouters, A VAD-robust multichannel Wiener filter algorithm for noise reduction in hearing aids, in: Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP '11), pp. 281–284.

[26] J. Benesty, J. Chen, Y. Huang, I. Cohen, Noise Reduction in Speech Processing, volume 2 of *Springer Topics in Signal Processing*, Springer-Verlag, 2009.

[27] G. H. Golub, C. F. Van Loan, Matrix Computations, The Johns Hopkins University Press, 3rd edition, 1996.

[28] D. H. Bailey, H. R. P. Gerguson, A Strassen-Newton algorithm for high-speed parallelizable matrix inversion, in: Proc. of the 1988 ACM/IEEE Conf. on Supercomputing, Orlando, FL., USA, pp. 419–424.

[29] S. Doclo, A. Spriet, J. Wouters, M. Moonen, Frequency-domain criterion for the speech distortion weighted multichannel Wiener filter for robust noise reduction, Speech Communication 49 (2007) 636–656.

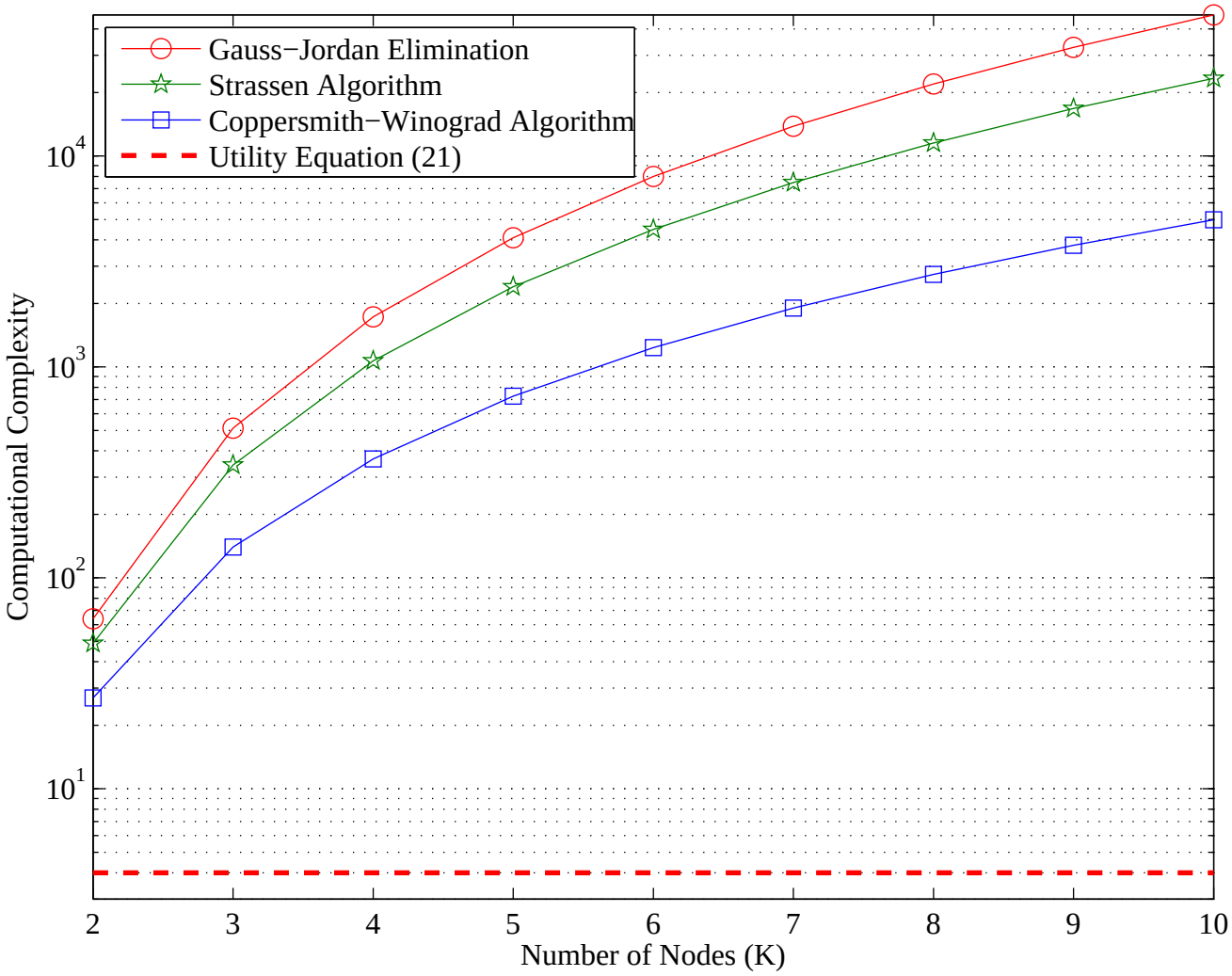

Figure 1: Computational complexity for finding U_{k-q} using the naive approach compared to (21).

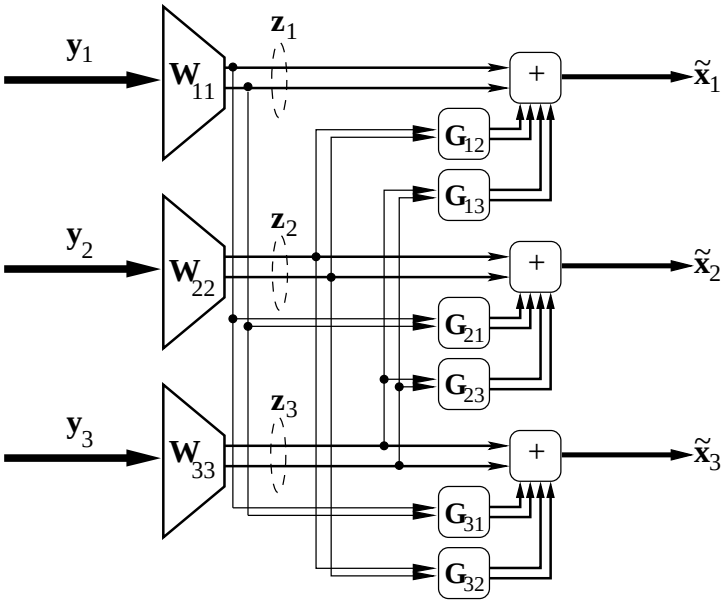

Figure 2: Depiction of the DANSE algorithm with three nodes each with two broadcast signals ($Q = 2$).

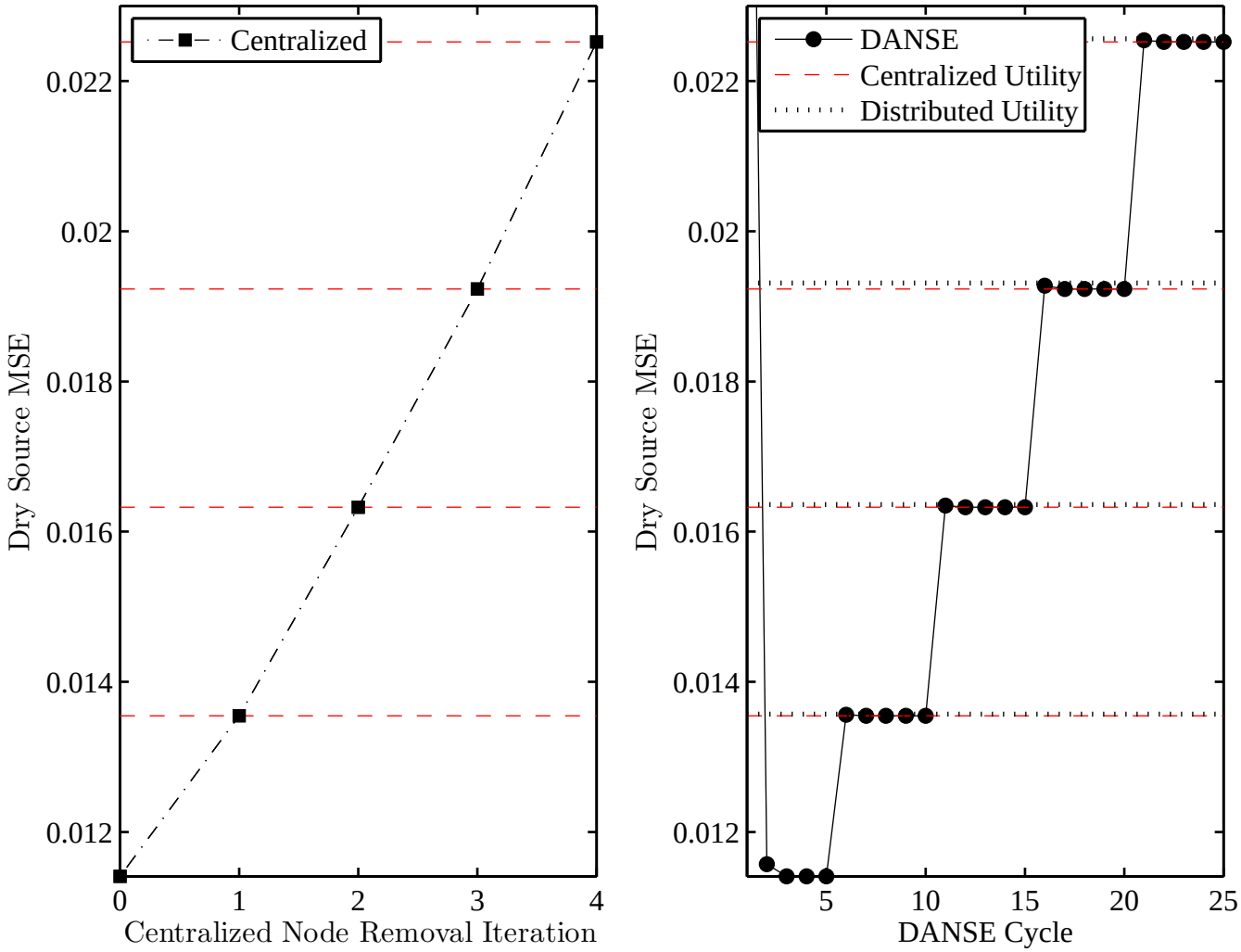

Figure 3: Comparison of centralized and distributed node removal process.

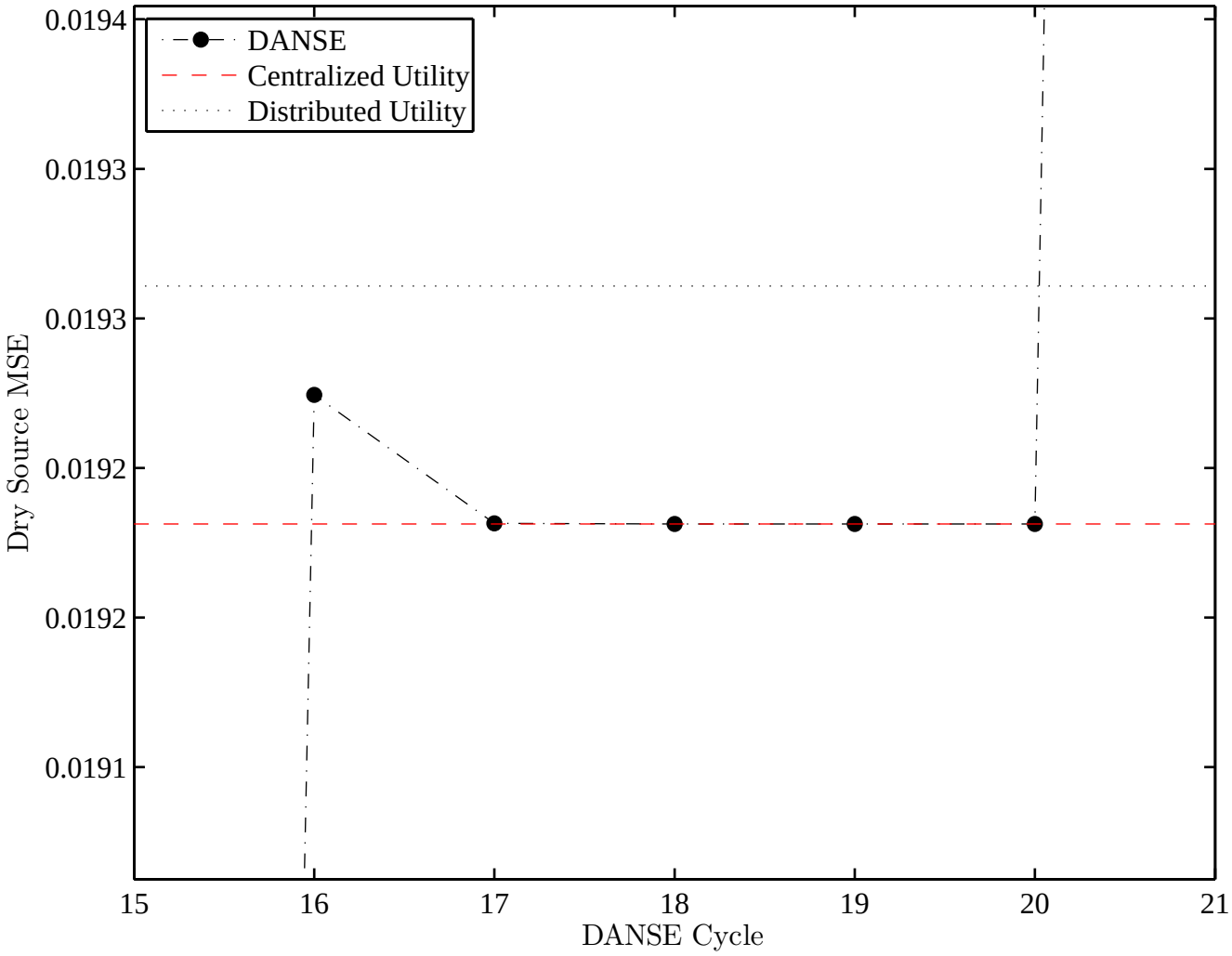

Figure 4: Utility upper bound in a distributed scenario compared to centralized (optimal) utility.

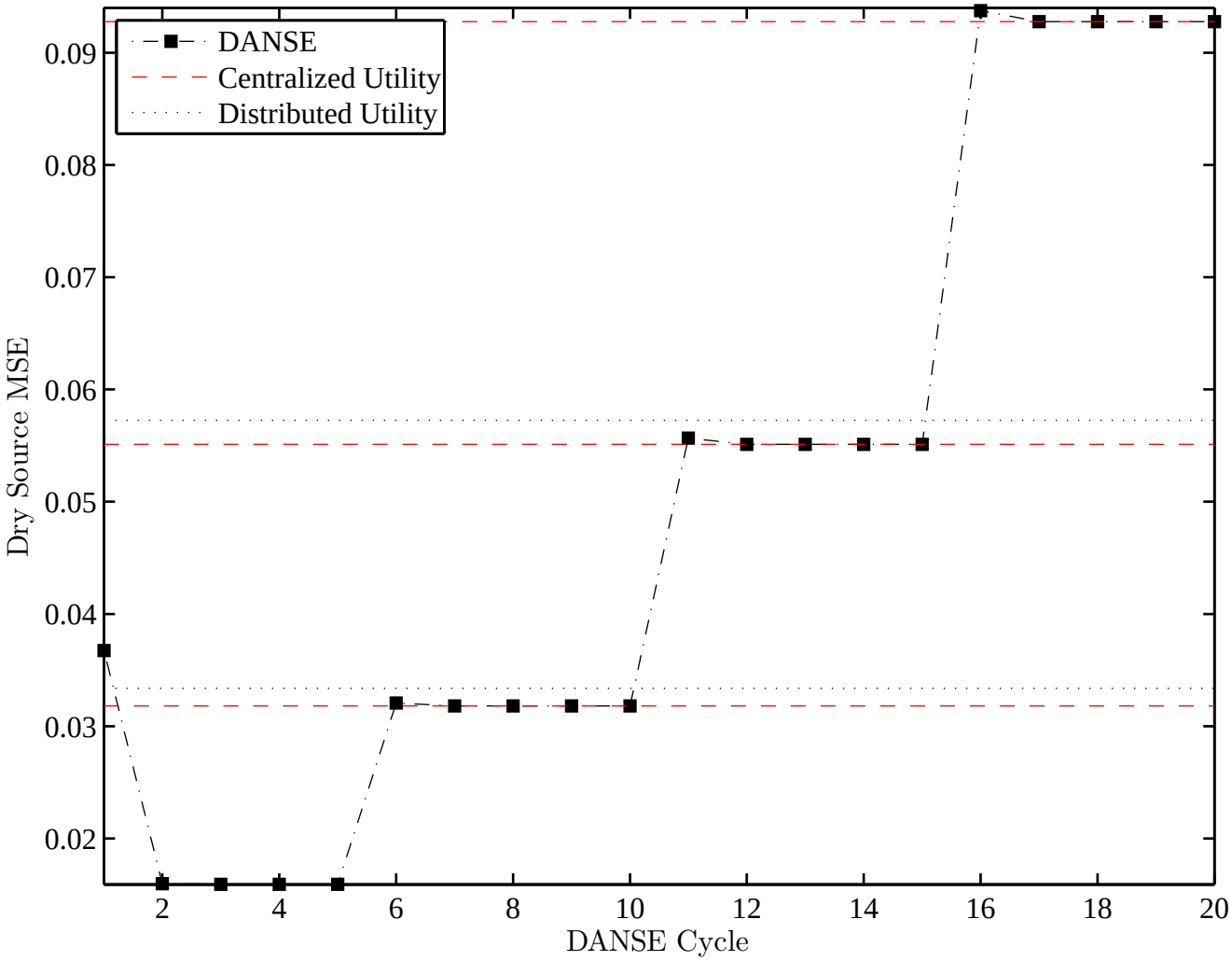

Figure 5: Utility upper bound in a distributed scenario compared to centralized (optimal) utility with $M_k \geq K(Q-1)$.

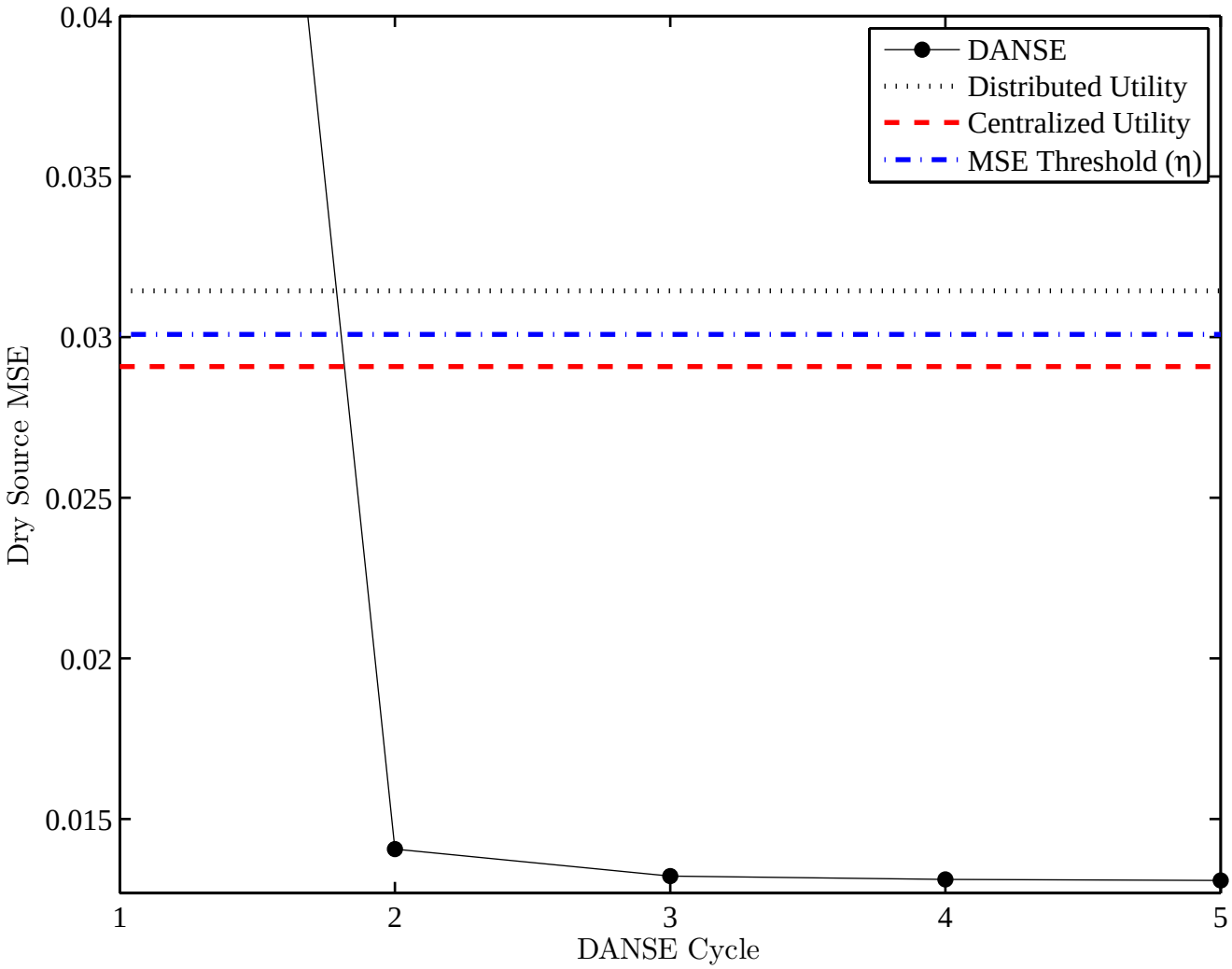

Figure 6: Distributed scenario where a node is kept in the network that would have been removed if the optimal estimator was used.

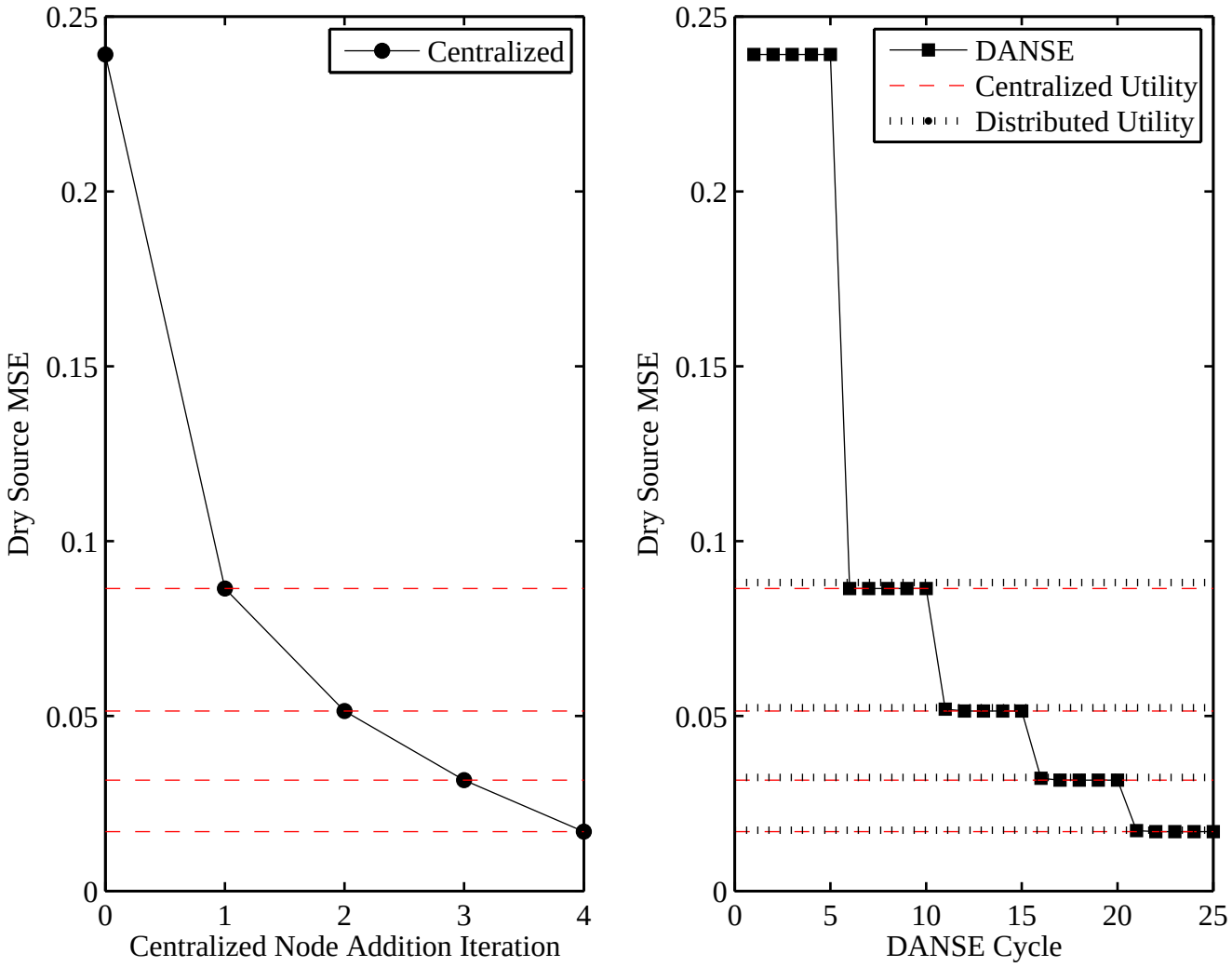

Figure 7: Comparison of centralized and distributed node addition process.

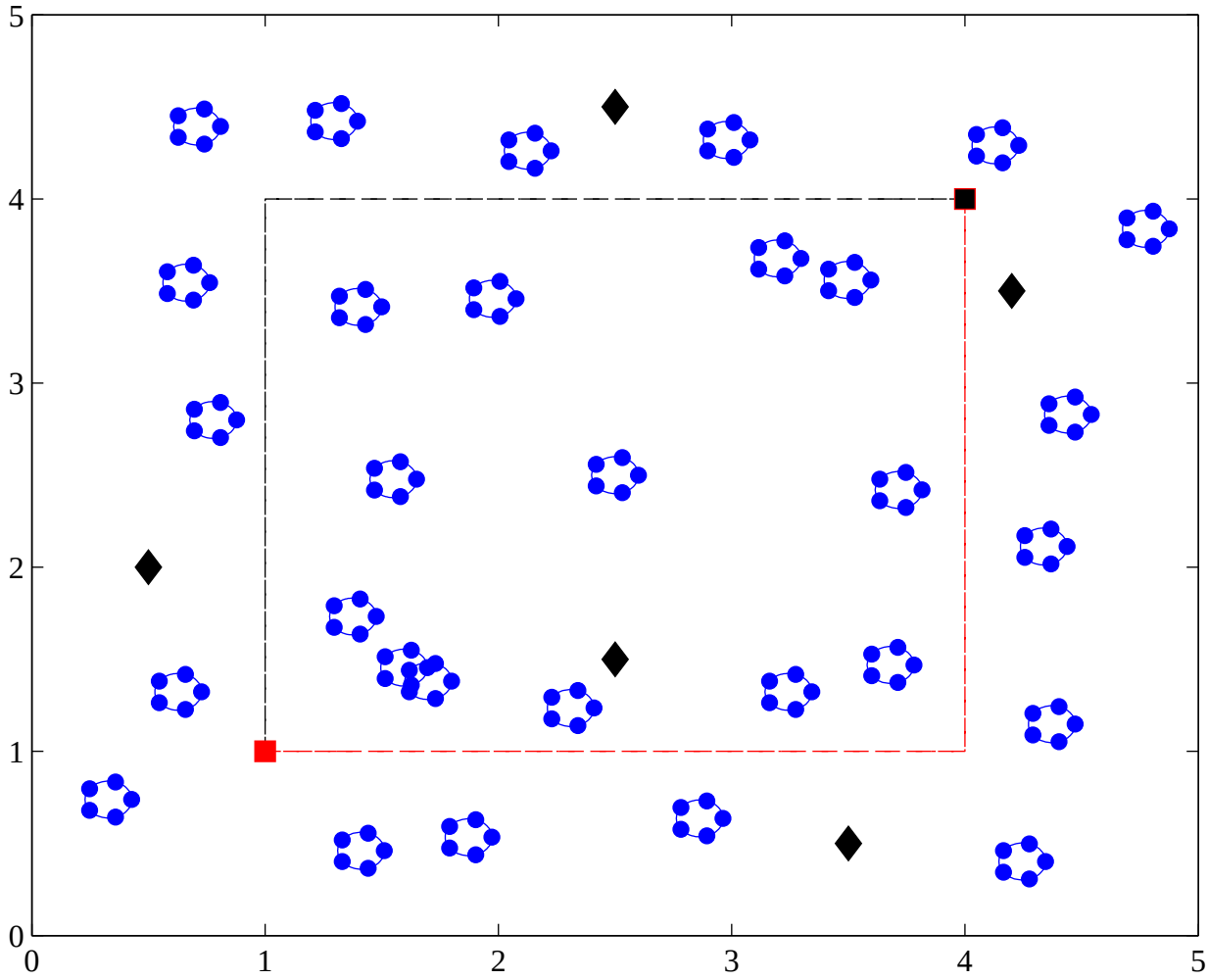

Figure 8: Depiction of simulated scenario. The network contains 30 nodes each with 5 sensors (○). There are two desired sources (□) that follow paths indicated by the dashed colored lines respective of the color of the source. There are also five white Gaussian noise sources present ♦.

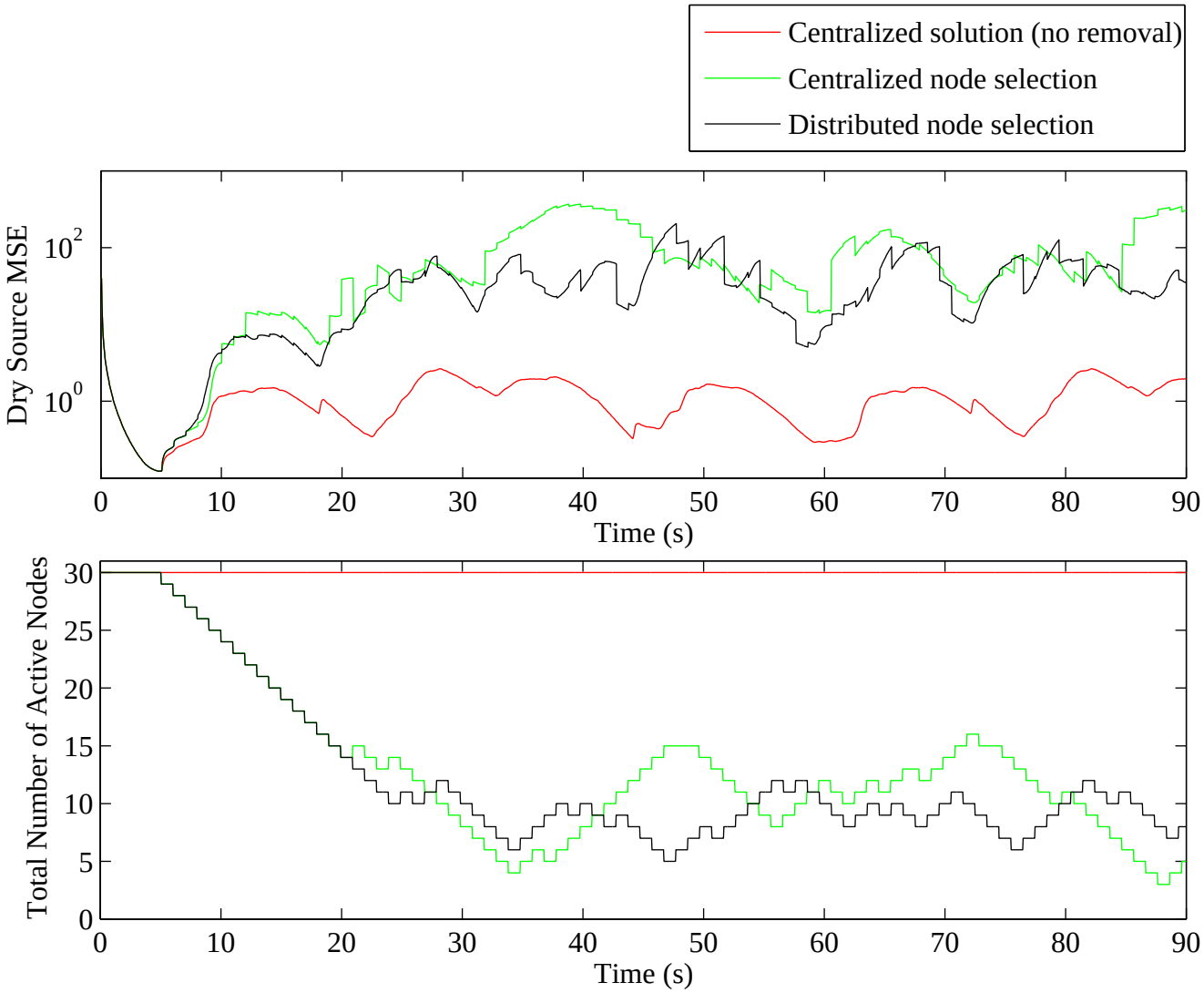

Figure 9: Comparison of MSE during centralized scenario without node removal as well as centralized and distributed node selection with total number of active nodes.

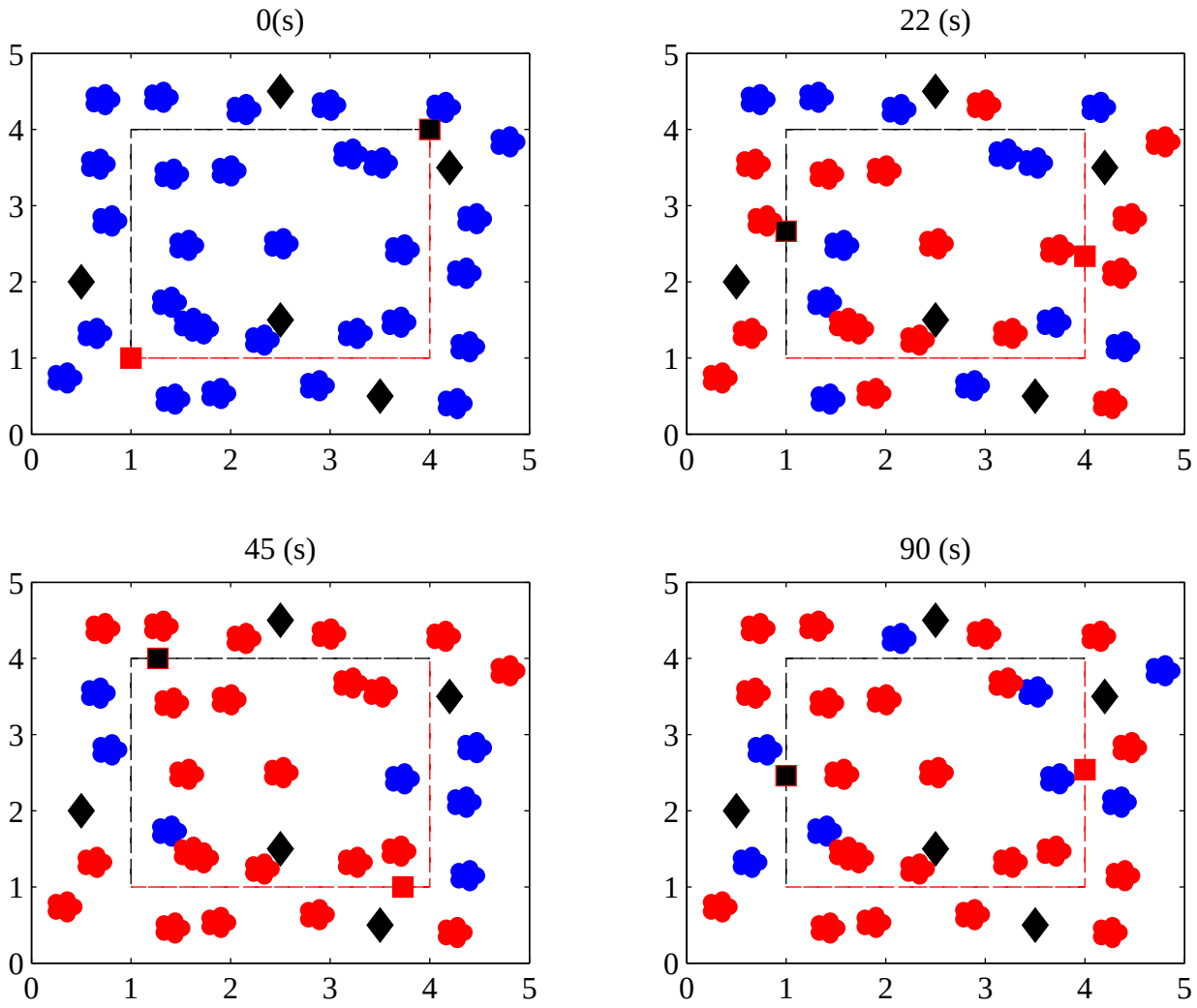

Figure 10: Node selection during various times in the estimation process using distributed utility bounds. The active nodes are shown in blue and the nodes that are only receiving signals from the other nodes and not transmitting their $\mathbf{z}_k$ signals are shown in red.

Table 1: Greedy Centralized Node Selection

1. Each node $\forall k \in K$ collects observations of the signals $\mathbf{y}_q,\ \forall q \in K$.
2. Each node calculates its utility $U_{s_{-k}}$ based on (35)
   - **If** $U_{s_{-k}} \leq \eta$ **and** node is connected to network
     - Remove node from network
     - Update $\mathbf{R}_{\mathbf{y}_{-k}\mathbf{y}_{-k}}$ and $\hat{\mathbf{W}}_{-k}$, $\forall q \in K \setminus \{k\}$
       Note : if there are multiple nodes for which the utility is below the threshold the node with the smallest utility is removed.
   - **ElseIf** $U_{s_{-k}} > \eta$ **and** node is not connected to network
     - Add node to network
     - Update $\mathbf{R}_{\mathbf{yy}}$ and $\hat{\mathbf{W}}_k$, $\forall q \in K \setminus \{k\}$
       Note : if there are multiple nodes for which the utility exceeds the threshold the node with the largest utility is added.
3. return to 1.

Table 2: The DANSE Algorithm

1. Initialize $0 \rightarrow i$, $1 \rightarrow u$
   Initialize $\mathbf{W}_{kk}^0$ and $\mathbf{G}_{-k}^0$ randomly, $\forall k \in K$
2. Each node $\forall k \in K$ performs the following update
   - Collect (new observations) of sensor signals $\mathbf{y}_k[iB+n]$, $n = 0 \dots B-1$.
   - Compress the sensor signals using (37),
     $\mathbf{z}_k^i = \mathbf{W}_{kk}^{iH} \mathbf{y}_k[iB+n]$, $n = 0 \dots B-1$
   - Broadcast signal $\mathbf{z}_k^i[iB+n]$, $n = 0 \dots B-1$
   - Collect broadcast signal $\mathbf{z}_{-k}^i[iB+n]$, $n = 0 \dots B-1$
   - Update estimates of $\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^i$ and $\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{x}}_k}^i$ with new observations
   - Update node specific parameters
     $$\begin{bmatrix} \cdot & \mathbf{W}_{kk}^{i+1} & \cdot \\ & \mathbf{G}_{-k}^{i+1} & \end{bmatrix} = \begin{cases} (\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^i)^{-1} \mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{x}}_k}^i & (\text{if } k = u) \\ \begin{bmatrix} \cdot & \mathbf{W}_{kk}^i & \cdot \\ & \mathbf{G}_{-k}^i & \end{bmatrix} & (\text{if } k \neq q) \end{cases}$$
3. Compute $\tilde{\mathbf{x}}_k[iB+n]$, $n = 0 \dots B-1$, as
   $\tilde{\mathbf{x}}_k[iB+n] = \mathbf{W}_{kk}^{i+1} \mathbf{y}_k[iB+n] + \mathbf{G}_{-k}^{i+1} \mathbf{z}_{-k}^i[iB+n]$
4. $i{+}1 \rightarrow i$
5. (u mod J) + $1 \rightarrow u$
6. return to 2.

Table 3: Greedy Distributed Node Selection

1. Each node $\forall k \in K$ collects its own $\mathbf{y}_k$ signals and broadcast signals, $\mathbf{z}_{-k}$ from other nodes connected in the network.
2. Each node calculates its utility $\overline{U}_{s_{-k}}$ based on (52) and scaled to the dry source as in (35)
   - **If** $\overline{U}_{s_{-k}} \leq \eta$ **and** node is connected to network
     - Remove node from network
     - Update $\mathbf{R}^{i+1}_{\tilde{\mathbf{y}}_{k_{-k}}\tilde{\mathbf{y}}_{k_{-k}}}$ and $\overline{\mathbf{W}}^{i+1}_{k_{-k}}$, $\forall q \in K \setminus \{k\}$
       Note : if there are multiple nodes for which the utility is below the threshold the node with the smallest utility is removed.
   - **ElseIf** $\overline{U}_{s_{-k}} > \eta$ **and** node is not connected to network
     - Add node to network
     - Update $\mathbf{R}^{i+1}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$ and $\overline{\mathbf{W}}^{i+1}_{k}$, $\forall q \in K \setminus \{k\}$
       Note : if there are multiple nodes for which the utility exceeds the threshold the node with the largest utility is added.
3. return to 1.