Distributed Relaxation on Augmented Lagrangian for Consensus based Estimation in Sensor Networks

Guang Xu, Henning Paul, and Armin Dekorsy
Department of Communications Engineering, University of Bremen, Bremen, Germany
Email: {xu,paul,dekorsy}@ant.uni-bremen.de

***Abstract*—This paper presents several new distributed algorithms to solve consensus based estimation problems in a decentralized way by adopting the well-known relaxation methods Jacobi, Gauss-Seidel and successive over-relaxation within a sensor network. In distributed estimation, all nodes collaborate to estimate the signals emitted from some common sources, employing iterative processing with one-hop data exchange. Consequently, the Jacobi-based consensus estimation algorithm produces a considerable communication effort due to its parallel processing. On the contrary, significant overhead can be saved by the Gauss-Seidel based consensus estimation algorithm with sequential update and exchange of local information. Additionally, both distributed algorithms can be accelerated by successive over-relaxation, resulting in further reduction of the communication effort for the distributed estimation. The evaluation of all the algorithms has been carried out in presence of both ideal and erroneous inter-node links in a randomly generated network. Moreover, the influence of the network topology on the distributed estimation has also been investigated, and the simulative results indicate that a network with low connectivity is preferred by the proposed algorithms.**

I. INTRODUCTION

The recent technological advances in wireless communication and distributed signal processing are promoting the applications of distributed monitoring, tracking and control in wireless sensor networks [1]. One common scenario of interest is the cooperative estimation, in which a group of sensor nodes is randomly deployed in a network and connected through inter-node links. All nodes aim to estimate signals emitted from some common source points. To this end, either centralized or distributed estimation can be applied within such a network [2]. For the centralized approach, all the received data and channel information are forwarded by local nodes via multiple hops to a central node where a joint estimation is performed. While for the distributed approach, each node only needs to perform the local estimation iteratively with some information exchanged between neighboring nodes. Such distributed estimation can be realized by several types of algorithms, e.g, the diffusion based algorithms [3], [4], [5], which can only achieve a consensus on the average of observations over the network. The subgradient method combined with primal decomposition algorithms [6], [7] can then be applied to the distributed consensus regression by solving a convex optimization problem [8]. Moreover, another class of algorithms based on the primal and dual decomposition method can also be applied to the distributed estimation, e.g., the least squares (LS) criterion based algorithms [9], [10] or maximum likelihood (ML) based algorithms [11], [12]. For the distributed estimation, considerable communication overhead is produced by the distributed algorithms, which is analyzed in [13]. In order to reduce the overhead, one alternative is to accelerate the convergence of the distributed algorithm [14], [15] or reduce the information being exchanged during the distributed processing [16].

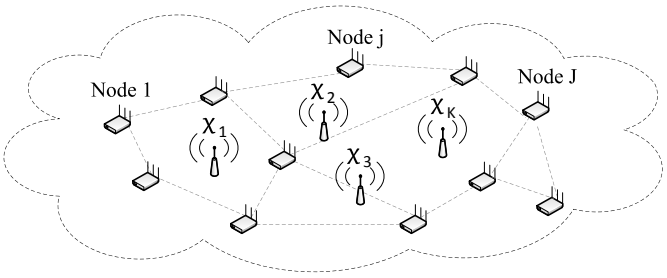

Fig. 1. A sensor network with J nodes receiving and detecting the common messages from K sources in a cooperative way.

Here, we are focusing on a different way to realize the distributed estimation which is based on relaxation methods [17], instead of using an average or a decoupling approach utilized in above algorithms. This leads to our new algorithms like the Jacobi based consensus estimation (Jacobi-CE), the Gauss-Seidel based consensus estimation (GS-CE) as well as their accelerated versions, the successive over-relaxation based consensus estimation (SOR-CE) algorithms. The derivation of these algorithms is detailed in the following section, and the corresponding performance in terms of convergence and communication overhead is evaluated in presence of both ideal and erroneous inter-node links considering different network topologies.

The remainder of this paper is structured as follows: The system model is described in Section II. A detailed derivation and discussion on the proposed distributed algorithms are given in Section III. In the subsequent Section IV, the proposed algorithms are simulated numerically, and the corresponding results are evaluated. Finally, the paper is concluded in Section V.

II. SYSTEM DESCRIPTION

The system scenario is shown in Fig. 1, where a sensor network composed of a set of nodes $\mathcal{J} = \{1, ..., J\}$ monitors some data sources. Here, each node is assumed to be equipped

with N_{R} receive antennas, and neighboring nodes are connected through inter-node links resulting in a set of edges $\mathcal{E}$ in between. In our investigation, both ideal (no disturbance of data exchange) and non-ideal inter-node links are considered, and each node $j \in \mathcal{J}$ is assumed to share information only with its neighboring nodes in the set $\mathcal{N}_j \subseteq \mathcal{J}$ through symmetric (bidirectional) transmission. For each source point k, it emits a data vector $\boldsymbol{x}_k \in \mathbb{C}^{N_{\mathrm{T}}\times 1}$ with N_{T} antennas. The transmitted signals per time instance from all K sources can be collected into a stacked vector $\boldsymbol{x} = [\boldsymbol{x}_1^{\mathrm{T}}, .., \boldsymbol{x}_K^{\mathrm{T}}]^{\mathrm{T}}$ containing in total $N_{\mathrm{I}} = N_{\mathrm{T}} \cdot K$ ($N_{\mathrm{I}} > N_{\mathrm{R}}$ is assumed) system input components. Those messages are observed by each node j leading to a local observation $\mathbf{y}_j \in \mathbb{C}^{N_{\mathrm{R}}\times 1}$ which is described by the linear model

$$\mathbf{y}_j = \mathbf{H}_j \boldsymbol{x} + \mathbf{n}_j, \tag{1}$$

disturbed by a channel $\mathbf{H}_j \in \mathbb{C}^{N_{\mathrm{R}}\times N_{\mathrm{I}}}$ which is known locally, and additive white Gaussian noise (AWGN) $\mathbf{n}_j \in \mathbb{C}^{N_{\mathrm{R}}\times 1}$ with a variance σ_n^2. Each node can detect the source messages $\boldsymbol{x}$ based on the local observation individually. However, due to the under-determined system (1) per node, the sources signals $\boldsymbol{x}$ cannot be estimated properly using only the local information. In order to recover the source information over all nodes, one alternative is to calculate a joint estimate $\mathbf{x}_{\text{cen}}$ in a central node, where all information over the whole network is aggregated, performing e.g., the LS estimation:

$$\mathbf{x}_{\text{cen}} = \arg\min_{\mathbf{x}'\in\mathbb{C}^{N_{\mathrm{I}}}} \frac{1}{2} \left\|\mathbf{y} - \mathbf{H}_{\text{cen}}\mathbf{x}'\right\|^2 \tag{2}$$

with a stacked vector $\mathbf{y} = [\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_J^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{C}^{JN_{\mathrm{R}}\times 1}$ and a stacked channel matrix $\mathbf{H}_{\text{cen}} = [\mathbf{H}_1^{\mathrm{T}}, \ldots, \mathbf{H}_J^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{C}^{JN_{\mathrm{R}}\times N_{\mathrm{I}}}$. Such a LS problem can be solved, e.g., by a zero forcing (ZF) approach leading to the intuitive solution[1]:

$$\mathbf{x}_{\mathrm{ZF}} = (\mathbf{H}_{\text{cen}}^{\mathrm{H}}\mathbf{H}_{\text{cen}})^{-1}\mathbf{H}_{\text{cen}}^{\mathrm{H}}\mathbf{y}. \tag{3}$$

Although the joint estimation can be performed in a central node, the information aggregation normally requires a complex routing protocol and produces a high communication cost particularly for a large scale network. To this end, instead of performing the joint estimation in a central node, all nodes in the network can collaborate to achieve the central solution in a distributed way with simple one-hop communication to share the information. For the distributed estimation, the central LS problem (2) has to be reformulated into a sum of separate local LS problems with additional constraints to keep the consistency over the whole network on the local estimates $\mathbf{x}_j = \mathbf{x}_i,\ i,j = 1,..,J$ which should be identical to the centralized estimate. In this respect, our target becomes a constrained LS problem given by

$$\begin{aligned} \mathbf{x}_j = \arg\min_{\mathbf{x}'_j\in\mathbb{C}^{N_{\mathrm{I}}}} & \sum_{j=1}^{J} \left\|\mathbf{y}_j - \mathbf{H}_j\mathbf{x}'_j\right\|^2 \\ \text{s.t.} \quad & \mathbf{x}_j = \mathbf{x}_i, \quad \forall i \in \mathcal{N}_j, \forall j \in \mathcal{J}. \end{aligned} \tag{4}$$

[1]The solution of (2) can also be extended to a minimum mean square error (MMSE) solution considering the effect of the noise, but the centralized solution is only used as a benchmark for the distributed approaches. Thus, without loss of generality, we only consider the ZF method as an example in this paper.

Such a constrained convex problem can be solved in a distributed fashion by the primal and dual decomposition based algorithms, e.g., the priority based augmented Lagrangian consensus estimation (PALCE) algorithm [16], which adopts an approximation method for decoupling the constraints of (4) in order to achieve a distributed implementation. In our new approach, we adopt several relaxation methods, namely the Jacobi, Gauss-Seidel (GS) and successive over-relaxation (SOR) [17] to solve the consensus constrained problem (4) in a distributed way rather using a decoupling method. Here, we first reconstruct the problem (4) into a central form using a stacked estimates vector $\mathbf{x} = [\mathbf{x}_1^{\mathrm{T}}, \cdots, \mathbf{x}_J^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{C}^{JN_{\mathrm{I}}\times 1}$, a block diagonal channel matrix $\mathbf{H} = \text{blkdiag}(\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_J) \in \mathbb{C}^{JN_{\mathrm{R}}\times JN_{\mathrm{I}}}$, and a matrix $\mathbf{A} \in \mathbb{R}^{2|\mathcal{E}|N_{\mathrm{I}}\times JN_{\mathrm{I}}}$ containing block elements $\mathbf{A}_{ij} \in \{\mathbf{I}, -\mathbf{I}, \mathbf{0}\}$ to represent all the constraints in (4). The reformulated target problem of (4) is thus given by

$$\begin{aligned} \mathbf{x} = \arg\min_{\mathbf{x}'\in\mathbb{C}^{JN_{\mathrm{I}}}} & \left\|\mathbf{y} - \mathbf{H}\mathbf{x}'\right\|^2 \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{0}. \end{aligned} \tag{5}$$

To solve (5), we use the augmented Lagrangian (AL) method, since the regularization term in the AL function facilitates higher robustness compared to the standard Lagrangian [18]. The corresponding AL cost function on (5) is given by

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{2}\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|^2 - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \frac{1}{2\mu}\|\mathbf{A}\mathbf{x}\|^2 \tag{6}$$

with a penalty parameter μ and a stacked multipliers vector $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^{\mathrm{T}}, .., \boldsymbol{\lambda}_J^{\mathrm{T}}]^{\mathrm{T}}$ where the vector $\boldsymbol{\lambda}_j \in \mathbb{C}^{|\mathcal{N}_j|N_{\mathrm{I}}\times 1},\ j = 1,..,J$ is composed of multipliers $\boldsymbol{\lambda}_{ji}$ for the constraint between node j and node $i,\ i \in \mathcal{N}_j$. Then, we can solve the vector $\mathbf{x}$ by setting $\partial\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})/\partial\mathbf{x} = 0$ and obtaining a linear equation:

$$\begin{aligned} (\mathbf{H}^{\mathrm{H}}\mathbf{H} + \frac{1}{\mu}\mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x} &= \mathbf{H}^{\mathrm{H}}\mathbf{y} + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda} \\ \mathbf{M}\mathbf{x} &= \mathbf{b}, \end{aligned} \tag{7}$$

defining the matrix $\mathbf{M} = \mathbf{H}^{\mathrm{H}}\mathbf{H} + \frac{1}{\mu}\mathbf{A}^{\mathrm{T}}\mathbf{A}$, which is a symmetric positive definite (SPD) matrix consisting of $J \times J$ block matrices $\mathbf{M}_{ji} \in \mathbb{C}^{N_{\mathrm{T}}\times N_{\mathrm{T}}},\ i,j = 1,2,..,J$. Note that $\mathbf{M}_{ji} \neq \mathbf{0}$ only if $i \in \mathcal{N}_j$. In addition, both vector $\mathbf{b} = \mathbf{H}^{\mathrm{H}}\mathbf{y} + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}$ and vector $\mathbf{x}$ can be decomposed into J sub-vectors $\mathbf{b}_j \in \mathbb{C}^{N_{\mathrm{I}}\times 1}$ and $\mathbf{x}_j \in \mathbb{C}^{N_{\mathrm{I}}\times 1},\ j = 1,2,..,J$, respectively.

III. Distributed Relaxation based algorithms

According to [17], the use of relaxation methods facilitates the distributed processing. Thus, we can solve the linear equation (7) iteratively in a distributed fashion by using the block-wise Jacobi, Gauss-Seidel and successive over-relaxation methods.

A. Distributed Jacobi-CE algorithm

For the Jacobi method, the matrix $\mathbf{M}$ in (7) is decomposed into three parts $\mathbf{M} = \mathbf{D}+\mathbf{L}+\mathbf{U}$, where $\mathbf{D}$ is the block diagonal matrix consisting of block elements $\mathbf{M}_{jj},\ j = 1,..,J$, $\mathbf{L}$ is the lower part of $\mathbf{M}$, which consists of matrices $\mathbf{M}_{ji},\ j > i$, and matrix $\mathbf{U}$ contains the upper part of $\mathbf{M}$ with matrices

$\mathbf{M}_{ji},\ j<i$. Subsequently, we can solve the linear equation (7) with following update equation for the estimate $\mathbf{x}^{k+1}$ in iteration $k+1$:

$$\mathbf{D}\mathbf{x}^{k+1}=\mathbf{b}-\mathbf{L}\mathbf{x}^{k}-\mathbf{U}\mathbf{x}^{k}, \tag{8}$$

where the estimate $\mathbf{x}^{k+1}$ is updated with the information $\mathbf{x}^{k}$ from the previous iteration k while the vector $\mathbf{b}$ remains constant. Note that $\mathbf{x}^{k+1}$ converges to the optimal value $\mathbf{x}^{*}=\mathbf{M}^{-1}\mathbf{b}$ if the spectral radius ρ_{Jacobi} of the iteration matrix $-\mathbf{D}^{-1}(\mathbf{L}+\mathbf{U})$ is smaller than 1 [19], which is satisfied by (7) in our system. Moreover, for an intuitive illustration, a detailed expression of (8) is rewritten as

$$\begin{bmatrix}\mathbf{M}_{11}\mathbf{x}_1^{k+1}\\ \mathbf{M}_{22}\mathbf{x}_2^{k+1}\\ \vdots\\ \mathbf{M}_{JJ}\mathbf{x}_J^{k+1}\end{bmatrix}=\begin{bmatrix}\mathbf{b}_1-\sum_{i\neq 1}^{J}\mathbf{M}_{1i}\mathbf{x}_i^{k}\\ \mathbf{b}_2-\sum_{i\neq 2}^{J}\mathbf{M}_{2i}\mathbf{x}_i^{k}\\ \vdots\\ \mathbf{b}_J-\sum_{i=1}^{J-1}\mathbf{M}_{Ji}\mathbf{x}_i^{k}\end{bmatrix}. \tag{9}$$

As (9) indicates, we can solve the local estimates $\mathbf{x}_j, j\in\mathcal{J}$ for each node separately. Once the estimates are updated, we can update the Lagrangian multipliers $\boldsymbol{\lambda}_{ji}$ in (6) within the same iteration by solving the dual problem with the steepest ascent method [20]. To this end, the targeted problem (4) can be solved in a distributed way leading to the Jacobi-based consensus estimation algorithm with the following update equations on local node j:

$$\begin{aligned}\mathbf{x}_j^{k+1}=&\mathbf{M}_{jj}^{-1}\left(\mathbf{b}_j-\sum_{i\neq j}^{J}\mathbf{M}_{ji}\mathbf{x}_i^{k}\right)\\ =&\left(\mathbf{H}_j^{\mathrm{H}}\mathbf{H}_j+\frac{2|\mathcal{N}_j|}{\mu}\mathbf{I}_{N_{\mathrm{T}}}\right)^{-1}\left(\mathbf{H}_j^{\mathrm{H}}\mathbf{y}_j\right.\\ &\left.+\sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ji}^{k}-\sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ij}^{k}+\sum_{i\in\mathcal{N}_j}\frac{2\mathbf{x}_i^{k}}{\mu}\right),\end{aligned} \tag{10a}$$

$$\boldsymbol{\lambda}_{ji}^{k+1}=\boldsymbol{\lambda}_{ji}^{k}-\frac{1}{\mu}(\mathbf{x}_j^{k}-\mathbf{x}_i^{k}),\qquad j=1,2,..,J. \tag{10b}$$

For the update of estimate $\mathbf{x}_j^{k+1}$ and multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ at iteration $k+1$, each node j only requires the last estimates $\mathbf{x}_i^{k}$ and multipliers $\boldsymbol{\lambda}_{ij}^{k}$ from its neighboring nodes $i\in\mathcal{N}_j$. Note that all nodes in the Jacobi-CE algorithm can perform the estimation in parallel, and the network is assumed to be perfectly synchronized in this paper. Once the local estimates $\mathbf{x}_j^{k+1}$ and multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ are updated, every node exchanges the newest information with its neighboring nodes. For the transmission of local estimates, since all neighboring nodes $i,\ i\in\mathcal{N}_j$ require the same estimate from node j, the local estimates $\mathbf{x}_j$ can thus be broadcasted to the neighboring nodes. The multipliers $\boldsymbol{\lambda}_{ji}$, which are related to the constraint between nodes j and i, are then unicasted from node j to node i. Hence, in each iteration, all J nodes need to broadcast their estimates and $2|\mathcal{E}|$ inter-node links are used to deliver the multipliers, which results in a total overhead $\mathcal{O}_{\text{Jacobi-CE}}=JN_{\mathrm{I}}+2|\mathcal{E}|N_{\mathrm{I}}$ per iteration over the whole network for the Jacobi-CE algorithm.

B. Distributed GS-CE algorithm

Due to the information exchange between nodes, a high communication effort over inter-node links is required because of the parallel processing per iteration in the Jacobi-CE algorithm. In order to reduce the overhead, another relaxation method, the Gauss-Seidel method [17] with a sequential update per iteration can be applied for the distributed estimation. Similar to the Jacobi method, we can solve the problem (7) in an iterative way by the Gauss-Seidel method with the following update equation:

$$\begin{aligned}(\mathbf{D}+\mathbf{L})\,\mathbf{x}^{k+1}&=\mathbf{b}-\mathbf{U}\mathbf{x}^{k}\\ \mathbf{D}\mathbf{x}^{k+1}&=\mathbf{b}-\mathbf{L}\mathbf{x}^{k+1}-\mathbf{U}\mathbf{x}^{k},\end{aligned} \tag{11}$$

where the matrices $\mathbf{D},\mathbf{L},\mathbf{U}$ are defined as above. Here, the estimate $\mathbf{x}^{k+1}$ converges to the optimal value $\mathbf{x}^{*}$ if the spectral radius ρ_{GS} of the iteration matrix $-(\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}$ is smaller than 1 [19], which is also fulfilled by (7) in our system.

Recalling the intuitive form (9), the implementation of the linear system (11) can also be split among the distributed nodes. Nevertheless, different to the Jacobi method, the update of estimate $\mathbf{x}_j^{k+1}$ requires the newest update $\mathbf{x}_i^{k+1}$ for $i<j$, which indicates that the estimates $\mathbf{x}_j,\ j=1,2,..,J$ cannot be updated simultaneously. Thus, for the distributed implementation of the GS method, all the nodes have to calculate their local estimates in a sequential way. In the same iteration, the corresponding multipliers $\boldsymbol{\lambda}_{ji}$ can also be updated in a distributed way among the nodes using the steepest ascent. To this end, we can solve the target problem (4) by the GS based consensus estimation algorithm with the following update equations:

$$\begin{aligned}\mathbf{x}_j^{k+1}=&\mathbf{M}_{jj}^{-1}\left(\mathbf{b}_j-\sum_{i=1}^{j-1}\mathbf{M}_{ji}\mathbf{x}_i^{k+1}-\sum_{i=j+1}^{J}\mathbf{M}_{ji}\mathbf{x}_i^{k}\right)\\ =&\left(\mathbf{H}_j^{\mathrm{H}}\mathbf{H}_j+\frac{2|\mathcal{N}_j|}{\mu}\mathbf{I}_{N_{\mathrm{T}}}\right)^{-1}\left(\mathbf{H}_j^{\mathrm{H}}\mathbf{y}_j+\sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ji}^{k}\right.\\ &\left.-\sum_{i\in\mathcal{N}_j^{-}}\boldsymbol{\lambda}_{ij}^{k}-\sum_{i\in\mathcal{N}_j^{+}}\boldsymbol{\lambda}_{ij}^{k}+\sum_{i\in\mathcal{N}_j^{-}}\frac{2\mathbf{x}_i^{k+1}}{\mu}+\sum_{i\in\mathcal{N}_j^{+}}\frac{2\mathbf{x}_i^{k}}{\mu}\right),\end{aligned} \tag{12a}$$

$$\boldsymbol{\lambda}_{ji}^{k+1}=\boldsymbol{\lambda}_{ji}^{k}-\frac{1}{\mu}(\mathbf{x}_j^{k+1}-\mathbf{x}_i^{k+1}),\qquad i\in\mathcal{N}_j^{-}, \tag{12b}$$

$$\boldsymbol{\lambda}_{ji}^{k+1}=\boldsymbol{\lambda}_{ji}^{k}-\frac{1}{\mu}(\mathbf{x}_j^{k+1}-\mathbf{x}_i^{k}),\qquad i\in\mathcal{N}_j^{+}, \tag{12c}$$

where $\mathcal{N}_j^{-}$ and $\mathcal{N}_j^{+}$ denote the sets of neighboring nodes of node j with indices $i<j$ and $i>j$, respectively. The local estimate $\mathbf{x}_j^{k+1}$ of node j in iteration $k+1$ is updated with the newest estimates $\mathbf{x}_i^{k+1}$ received from the neighboring nodes $i\in\mathcal{N}_j^{-}$, besides, the estimates $\mathbf{x}_i^{k},\ i\in\mathcal{N}_j^{+}$ as well as the multipliers $\boldsymbol{\lambda}_{ij}^{k}$ received in the previous iteration are also required. Due to the sequential update of the distributed nodes, the update of multipliers $\boldsymbol{\lambda}_{ji}$ is divided into two types. For the case $i<j$, the multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ are updated with the local estimate $\mathbf{x}_j^{k+1}$ and the newly received estimates $\mathbf{x}_i^{k+1}$; while

for $i > j$, node j can only utilize the neighboring estimates $\mathbf{x}_i^k$ from the last iteration to update the corresponding multipliers. Once the local information is updated, node j broadcasts the estimate $\mathbf{x}_j^{k+1}$ and transmits the multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ to its $|\mathcal{N}_j|$ neighboring nodes. In the following step, the node $j+1$ starts to update. Here, note that the local information is also sequentially exchanged within one iteration, i.e., only one node transmits data over inter-node links per step leading to the overhead $\mathcal{O}_{\text{GS-CE}} = (1 + |\mathcal{N}_j|)N_{\text{I}}$.

Moreover, in order to make a fair comparison between the Jacobi-CE and the GS-CE algorithms regarding the communication overhead, we introduce a synchronized time instance for each exchange of local estimates and multipliers, which is counted by step n. Thus, for Jacobi-CE, all nodes update and exchange the local information per step, so that one iteration in Jacobi-CE is identical to one step as defined here. For GS - CE, only one node updates and exchanges its local information per step, such that one iteration in GS-CE terminates when J steps update of all nodes is performed. Therefore, only a small quantity of overhead is produced per step by the sequential update in GS-CE compared to the parallel update in Jacobi-CE.

C. Distributed SOR-CE algorithms

In order to further reduce the communication effort, we can use another relaxation approach, the successive over-relaxation [17], which accelerates the convergence of both Jacobi and GS methods, since the spectral radius ρ_{SOR} of the iteration matrix is smaller compared to those of the Jacobi and GS methods (a detailed convergence analysis can be found in [19]). Therefore, we apply this acceleration approach to the above algorithms leading to the Jacobi-based SOR on consensus estimation (JSOR-CE) and GS-based SOR on consensus estimation (GSSOR-CE) algorithms, respectively.

Based on the Jacobi method, the JSOR method solves for the estimate $\mathbf{x}$ in (7) with an additional relaxation parameter ω which is set to be $\omega > 1$. Correspondingly, the solution of JSOR for the problem (7) in the centralized form reads [17]:

$$\mathbf{D}\mathbf{x}^{k+1} = \omega\left(\mathbf{b} - \mathbf{L}\mathbf{x}^{k} - \mathbf{U}\mathbf{x}^{k}\right) + (1-\omega)\mathbf{D}\mathbf{x}^{k} \tag{13}$$

with the spectral radius ρ_{JSOR} of iteration matrix $\mathbf{D}^{-1}((1-\omega)\mathbf{D} - \omega(\mathbf{L}+\mathbf{U}))$ smaller than 1. The update of the estimate $\mathbf{x}^{k+1}$ can be viewed as an relaxation between the Jacobi solution and previous estimate $\mathbf{x}^k$ weighted by the relaxation parameter. Following the same principle of distributed Jacobi implementation, the update of the estimate $\mathbf{x}$ in (13) can also be distributed among the nodes. Besides, for solving the consensus constrained problem (4), the multipliers also need to be updated per iteration, which is identical to the Jacobi-CE algorithm. Thus, the corresponding update equations of the JSOR-CE algorithm for the local estimate $\mathbf{x}_j$ as well as the multipliers $\boldsymbol{\lambda}_{ji}$ of node j are given by

$$\begin{aligned}
\mathbf{x}_j^{k+1} &= \omega\mathbf{M}_{jj}^{-1}\left(\mathbf{b}_j - \sum_{i\neq j}^{J}\mathbf{M}_{ji}\mathbf{x}_i^k\right) + (1-\omega)\mathbf{x}_j^k \\
&= \left(\mathbf{H}_j^{\text{H}}\mathbf{H}_j + \frac{2|\mathcal{N}_j|}{\mu}\mathbf{I}_{N_{\text{T}}}\right)^{-1}\omega\left(\mathbf{H}_j^{\text{H}}\mathbf{y}_j + \sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ji}^k \right. \\
&\quad \left. - \sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ij}^k + \sum_{i\in\mathcal{N}_j}\frac{2\mathbf{x}_i^k}{\mu}\right) + (1-\omega)\mathbf{x}_j^k, \qquad (14a)\\
\boldsymbol{\lambda}_{ji}^{k+1} &= \boldsymbol{\lambda}_{ji}^k - \frac{1}{\mu}(\mathbf{x}_j^k - \mathbf{x}_i^k), \quad j = 1,2,..,J. \qquad (14b)
\end{aligned}$$

A proper relaxation parameter ω needs to be chosen for the update of the local estimate in order to achieve a faster convergence compared to the Jacobi-CE algorithm. Apart from that, the communication effort produced by the iterative processing in JSOR-CE is identical to the Jacobi-CE algorithm, as the estimate $\mathbf{x}_j$ is transmitted by node j in a broadcast way, while the multipliers $\boldsymbol{\lambda}_{ji}$ are delivered in a unicast way. Thus, a total overhead $\mathcal{O}_{\text{JSOR-CE}} = JN_{\text{I}} + 2|\mathcal{E}|N_{\text{I}}$ is produced per step in JSOR-CE algorithm.

Moreover, as mentioned, the SOR method can also be applied to the GS approach. Thus, the GS update (11) is also extended with the relaxation factor ω resulting in the centralized GSSOR update equation [17]:

$$\mathbf{D}\mathbf{x}^{k+1} = \omega\left(\mathbf{b} - \mathbf{L}\mathbf{x}^{k+1} - \mathbf{U}\mathbf{x}^{k}\right) + (1-\omega)\mathbf{D}\mathbf{x}^{k} \tag{15}$$

with the spectral radius ρ_{GSSOR} of iteration matrix $(\omega\mathbf{L} + \mathbf{D})^{-1}((1-\omega)\mathbf{D} - \omega\mathbf{U})$ smaller than 1. Like JSOR, (15) also exhibits the property of decomposablity among nodes for distributed implementation. Utilizing the update of multipliers $\boldsymbol{\lambda}_{ji}$ in the GS-CE algorithm (16b) and (16c), we can solve the target problem (4) in a distributed way leading to the GSSOR-CE algorithm with following update equations:

$$\begin{aligned}
\mathbf{x}_j^{k+1} &= \omega\mathbf{M}_{jj}^{-1}\left(\mathbf{b}_j - \sum_{i=1}^{j-1}\mathbf{M}_{ji}\mathbf{x}_i^{k+1} - \sum_{i=j+1}^{J}\mathbf{M}_{ji}\mathbf{x}_i^k\right) + (1-\omega)\mathbf{x}_j^k \\
&= \omega\left(\mathbf{H}_j^{\text{H}}\mathbf{H}_j + \frac{2|\mathcal{N}_j|}{\mu}\mathbf{I}_{N_{\text{T}}}\right)^{-1}\left(\mathbf{H}_j^{\text{H}}\mathbf{y}_j + \sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ji}^k \right. \\
&\quad \left. - \sum_{i\in\mathcal{N}_j}\boldsymbol{\lambda}_{ij}^k + \sum_{i\in\mathcal{N}_j^-}\frac{2\mathbf{x}_i^{k+1}}{\mu} + \sum_{i\in\mathcal{N}_j^+}\frac{2\mathbf{x}_i^k}{\mu}\right) + (1-\omega)\mathbf{x}_j^k, \qquad (16a)\\
\boldsymbol{\lambda}_{ji}^{k+1} &= \boldsymbol{\lambda}_{ji}^k - \frac{1}{\mu}(\mathbf{x}_j^{k+1} - \mathbf{x}_i^{k+1}), \quad i \in \mathcal{N}_j^-, \qquad (16b)\\
\boldsymbol{\lambda}_{ji}^{k+1} &= \boldsymbol{\lambda}_{ji}^k - \frac{1}{\mu}(\mathbf{x}_j^{k+1} - \mathbf{x}_i^{k}), \quad i \in \mathcal{N}_j^+, \qquad (16c)
\end{aligned}$$

where the local estimates $\mathbf{x}_j^{k+1}$ and multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ are updated in a sequential way with information sharing among the neighboring nodes. Here, like the GS-CE algorithm, only one node is required to broadcast the local estimate and to transmit the multipliers oriented to the neighboring nodes per step. Hence, in each step a total overhead $\mathcal{O}_{\text{GSSOR-CE}} = (1 + |\mathcal{N}_j|)N_{\text{I}}$ is generated. To this end, the communication overhead produced by the Jacobi type and GS type algorithms together with the PALCE algorithm from the state of the art

TABLE I
TOTAL OVERHEAD GENERATED OVER INTER-NODE LINKS PER STEP FOR VARIOUS TYPES OF DISTRIBUTED ALGORITHMS

	Line	Full Mesh	Random
Jacobi type	$(3J-2)N_{\mathrm{I}}$	JJN_{I}	$(J+2\lvert\mathcal{E}\rvert)N_{\mathrm{I}}$
GS type	$3N_{\mathrm{I}}$	JN_{I}	$(1+\lvert\mathcal{N}_j\rvert)N_{\mathrm{I}}$
PALCE	$1.5JN_{\mathrm{I}}$	$J(J+3)/4N_{\mathrm{I}}$	$(J+0.5\lvert\mathcal{E}\rvert)N_{\mathrm{I}}$

is summarized in the table I[2] for different network topologies. It can be seen that the overhead of GS type algorithms per step is the lowest among all, which increases linearly with the number of neighboring nodes $|\mathcal{N}_j|$ per node j, while for the other algorithms, the overhead grows with the number of edges $|\mathcal{E}|$ in the network.

IV. PERFORMANCE EVALUATION

In this section, the numerical performance of the distributed algorithms is investigated. For the evaluation, all algorithms are simulated by means of the Monte Carlo method for a network consisting of $J = 10$ nodes, which are connected with a connectivity ratio r that is defined as the ratio between actual number of edges and the maximum number of edges when all nodes are connected:

$$r = \frac{|\mathcal{E}|}{J(J-1)/2}. \tag{17}$$

Note that the ratio should always be kept $r \geq 2/J$, since the whole network is required to be connected. The connectivity ratio is also an indicator for the average number of neighboring nodes over the network. Here, each node is set to be equipped with two receive antennas $N_{\mathrm{R}} = 2$. Besides, we assume $K = 5$ source points modeled with a single transmit antenna $N_{\mathrm{T}} = 1$ each that are deployed within the network to broadcast a source message $\boldsymbol{x}$ which contains Gaussian random values $\mathcal{N}(0, \sigma_{\boldsymbol{x}}^2)$ with mean 0 and variance $\sigma_{\boldsymbol{x}}^2 = 1$. In the sequel, the distributed algorithms are applied to such a network with different connectivity ratios. In addition to that, both ideal and non-ideal inter-node links are considered for the investigation.

A. aMSE performance for ideal inter-node links

For the evaluation of the distributed estimation, we are using the metric of the averaged mean squared error (aMSE), which is defined as the mean squared error on the estimates $\mathbf{x}_j$ that is averaged over all nodes:

$$\text{aMSE} = \frac{1}{J}\sum_{j=1}^{J} \mathrm{E}\left\{\|\boldsymbol{x} - \mathbf{x}_j\|^2\right\}, \tag{18}$$

while for the centralized estimation, we use the MSE, i.e, $\mathrm{E}\{\|\boldsymbol{x} - \mathbf{x}_{\text{cen}}\|^2\}$ as a reference.

In Fig. 2, the aMSE performance of the distributed algorithms is illustrated considering a signal to noise ratio (SNR[3]) of 10 dB and ideal inter-node links. It can be observed that the aMSE of all algorithms decreases over the steps and converges to the centralized ZF solution, but their convergence speed is not all the same. Due to the sequential update, the GS type algorithms show slower convergence with respect to the update steps compared to the Jacobi type algorithms. In addition, the convergence of both Jacobi-CE and GS-CE algorithms is accelerated by the SOR approach resulting in better performance of the JSOR-CE and GSSOR-CE algorithm, respectively. Compared to the algorithms from the state of the art, only JSOR-CE shows a similar convergence to the PALCE algorithm in this scenario. However, when we consider the communication overhead produced during the distributed estimation as shown in Fig. 3, a significant effort can be saved by GSSOR-CE and GS-CE compared to the Jacobi type

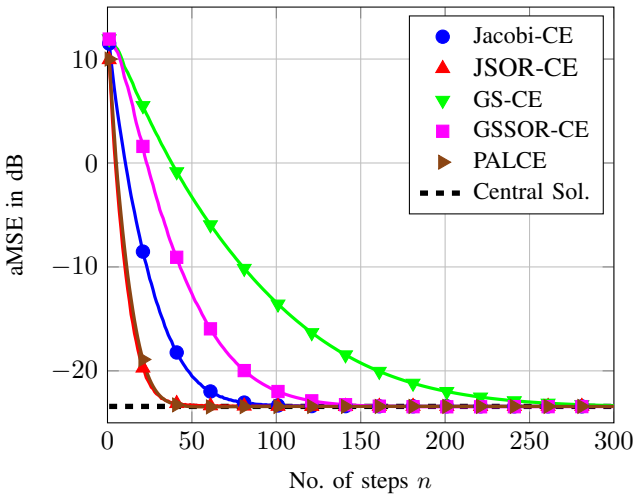

Fig. 2. aMSE performance evaluated over 1000 trials vs. No. of steps. $J = 10, K = 5, N_{\mathrm{R}} = 2, N_{\mathrm{T}} = 1$, random topology $r = 0.6$, SNR $= 10$dB. Each marker represents the performance of every 20 steps.

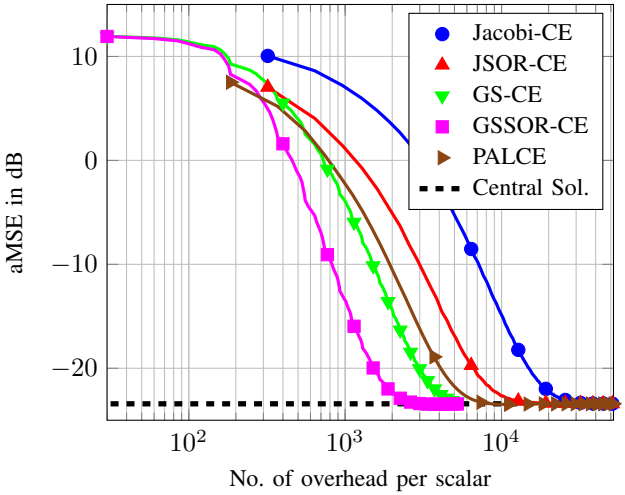

Fig. 3. aMSE performance evaluated over 1000 trials vs. total no. of overhead. $J = 10, K = 5, N_{\mathrm{R}} = 2, N_{\mathrm{T}} = 1$, random topology $r = 0.6$, SNR $= 10$dB. Each marker represents the performance of every 20 steps.

[2]Note that for the line topology, the overhead produced by the nodes at the front and the end of the line in GS type algorithm is $2N_{\mathrm{I}}$, since they are connected with only one neighboring node.

[3]The SNR here is related to the transmitted signal from source to the nodes, it should be distinguished with the SNR_{in} for the inter-node links.

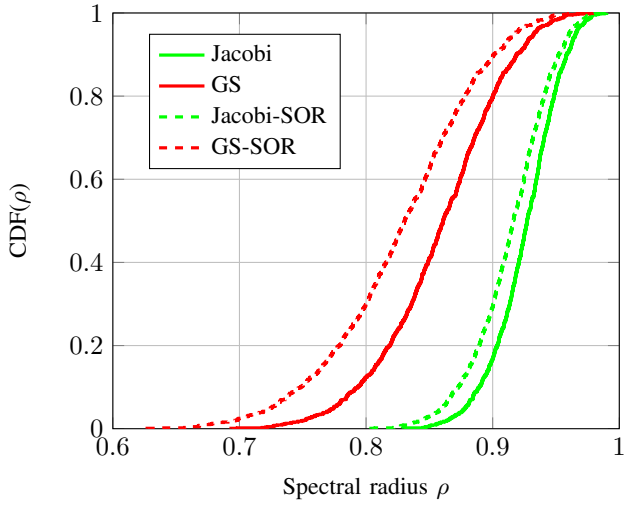

Fig. 4. Cumulative distribution function of spectral radius ρ of iteration matrix in Jacobi, Gauss-Seidel, Jacobi-SOR and GS-SOR methods over 1000 trials.

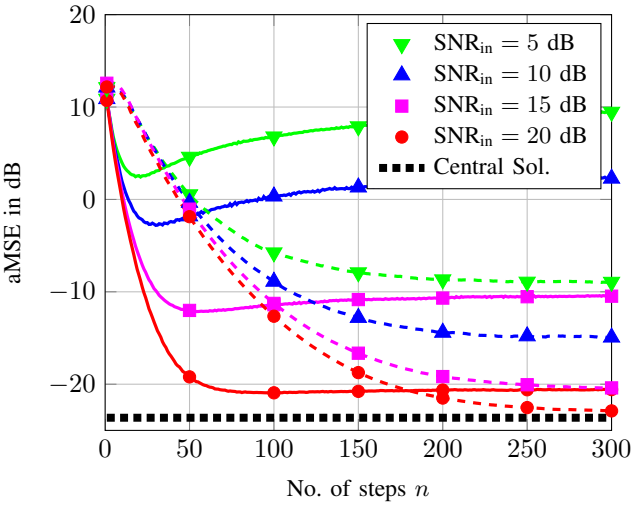

Fig. 5. aMSE performance of the Jacobi-CE (—) and the GS-CE (····) algorithms regarding noisy inter-node links with various SNR_{in}. $J = 10, K = 5, N_{\text{R}} = N_{\text{T}} = 1$, random topology $r = 0.6$, SNR $= 10$dB.

algorithms and the PALCE algorithm, since in each step only one node needs to transmit the information leading to the lowest overhead per step, which is labeled by the markers in Fig. 3. On the other hand, Fig. 3 also indicates that the convergence of GS type algorithms with respect to the iteration is faster than Jacobi type algorithms, since $\rho_{\text{GS}} < \rho_{\text{Jacobi}}$ according to [19], which is also verified by stimulative results regarding the cumulative distribution function (CDF) of ρ in both type algorithms in Fig. 4.

B. aMSE performance for erroneous inter-node links

In the sequel, we consider a more realistic scenario, where the inter-node links are not ideal any more but are disturbed by some common erroneous factors, e.g., noise and quantization error are taken into account. Regarding the noise, we assume

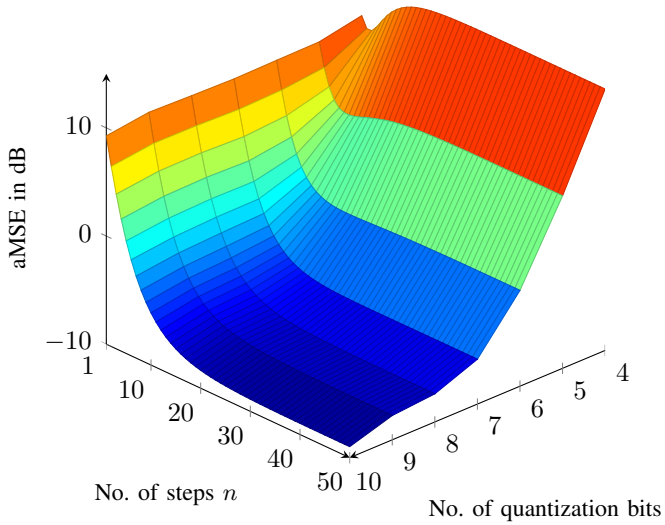

Fig. 6. aMSE performance of the Jacobi-CE algorithm regarding different levels of quantization. $J = 10, K = 5, N_{\text{R}} = N_{\text{T}} = 1$, random topology $r = 0.6$, SNR $= 10$dB.

that AWGN, which is determined by the SNR_{in} for all the inter-node links, is added to the information vector $\mathbf{x}_i$ and $\boldsymbol{\lambda}_{ij}$ received by each node j from its neighboring nodes i. Fig. 5 illustrates the effect of noisy inter-node links on the estimation performance for both Jacobi-CE and GS-CE algorithms. As can be seen, the error floor of both algorithms decreases, as the SNR_{in} increases from 5 dB to 20 dB. It can also be noticed that the Jacobi-CE algorithm is quite sensitive to the inter-node noise, since the performance is significantly deteriorated when the SNR_{in} is low. On the other hand, the GS-CE algorithm achieves a higher performance against the inter-node noise compared to Jacobi-CE, as the error floor of GS-CE is much lower than Jacobi-CE for the same SNR_{in}. In addition to the effect of inter-node noise, when modulated symbols are adopted for exchanging the information among nodes, then the estimates $\mathbf{x}_j$ and multipliers $\boldsymbol{\lambda}_{ji}$ need to be quantized according to a quantization level before being transmitted. Due to this procedure, an additional quantization error[4] is encountered, as Fig. 6 depicts the aMSE performance of the Jacobi-CE algorithm regarding various quantization levels from 4 bits to 10 bits. An acceptable estimation performance can be achieved by a high quantization level with a sufficient number of steps. It can also be observed that nearly 20 dB improvement is achieved by 10 quantization bits compared to the quantization with only 4 bits. For a low number of quantization bits, similar to inter-node noise, the performance of the distributed algorithm is also seriously deteriorated by the additional error. Thus, in order to ensure an acceptable performance, a proper quantization resolution and range needs

[4]Note that the dynamic range of quantization is also a factor that affects the quantization error. Here, the performance is deteriorated due to the clipping effect of a limited dynamic range.

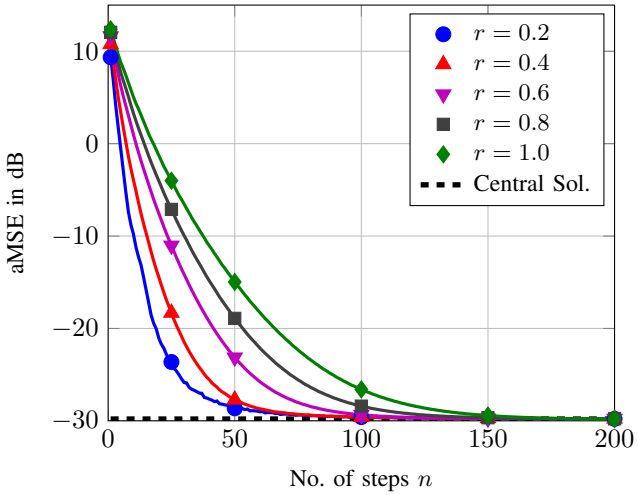

Fig. 7. aMSE performance of Jacobi-CE algorithm averaged over 1000 trials on random topology with connectivity ratio $r = 0.2, 0.3, ..., 1$, SNR $= 10$dB $J = 10, K = 5, N_{\mathrm{R}} = 2, N_{\mathrm{T}} = 1$, SNR=10 dB.

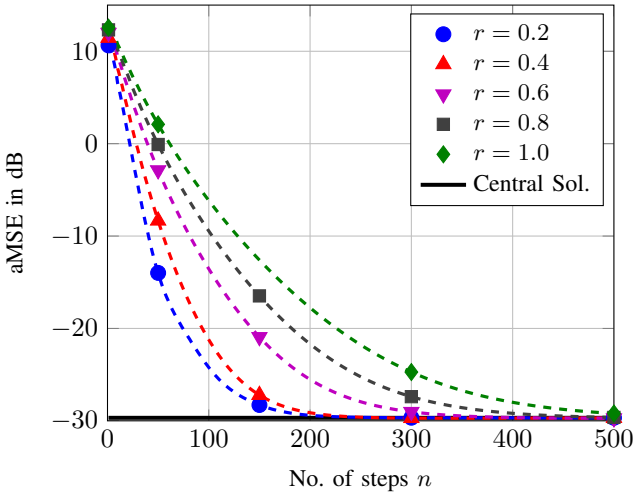

Fig. 8. aMSE performance of GS-CE algorithm averaged over 1000 trials on random topology with connectivity ratio $r = 0.2, 0.3, ..., 1$, SNR $= 10$dB $J = 10, K = 5, N_{\mathrm{R}} = 2, N_{\mathrm{T}} = 1$, SNR=10 dB.

to be chosen, considering the corresponding communication overhead.

C. aMSE performance for various network topologies

Here, we are focusing on the influence of the network topology on the distributed estimation. Still, the number of nodes and sources is fixed at $J = 10$ and $K = 5$. The nodes are randomly deployed and connected according to different connectivity ratios r in a range of 0.2 to 1. Note that when $r = 1$, then the network is fully meshed, and for the same connectivity ratio, the network topology could be different (e.g., both the line topology and the star topology have same connectivity ratio). Here, the performance of each connectivity ratio is an averaged performance simulated over 1000 randomly generated networks with the same r. Fig. 7 and Fig. 8 illustrate the aMSE performance w.r.t. different connectivity ratios for the Jacobi-CE and the GS-CE algorithm, respectively. In both figures, all the distributed estimation algorithms converge to the central solution, but interestingly it can be found that the convergence of distributed estimation in a low connectivity network is faster than that with a high connectivity ratio. Note that the convergence rate strongly relies on the graph's Laplacian matrix [21] which is determined by the topology, but detailed analysis of this is beyond the scope of this paper. Consequently, we can further reduce the communication effort by decreasing connectivity ratio of the network. However, for a link failure case, the whole network might get disconnected when r is quite low. Thus, a trade off between the connectivity ratio and the risk of link failure should be concerned.

V. CONCLUSION

In this paper, we presented Jacobi, Gauss-Seidel and successive over-relaxation based algorithms for distributed consensus estimation in sensor networks. The Jacobi-CE algorithm adopts parallel processing among nodes, resulting in a faster convergence, but it produces higher overhead compared to the GS-CE algorithm where the nodes update sequentially leading to a relatively low overhead per step. Both algorithms are successfully accelerated by the SOR method leading to a further reduction on the communication overhead. Furthermore, all the proposed algorithms have been evaluated in a scenario with ideal inter-node links, and a centralized estimation performance can be obtained by the distributed processing. For the implementation in presence of erroneous inter-node links, the performance of the distributed estimation degrades due to the additional errors in the update, but the GS-CE algorithm can achieve a higher performance compared to the Jacobi-CE algorithm. Moreover, a fast convergence can be achieved by the proposed distributed algorithms when applied within a network with a low connectivity ratio. However, an analytical investigation regarding the effect of connectivity on the convergence of the distributed algorithms needs to be made in the future.

ACKNOWLEDGMENT

The work leading to this publication was partially funded by the German Research Foundation (DFG) under grant Pa2507/1.

REFERENCES

[1] C. Chen, S. Zhu, X. Guan, and X. Shen, *Wireless Sensor Networks: Distributed Consensus Estimation*, 1st ed. Springer International Publishing, Dec. 2014.

[2] A. Swami, Q. Zhao, Y. Hong, and L. Tong, *Wireless Sensor Networks: Signal Processing and Communications*. Wiley, Nov. 2007.

[3] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[4] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *42nd IEEE Conference on Decision and Control*, vol. 5, Dec. 2003, pp. 4997–5002.

[5] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*, Apr. 2005, pp. 63–70.

[6] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[7] B. Johansson, T. Keviczky, M. Johansson, and K. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *47th IEEE Conference on Decision and Control (CDC)*, Dec. 2008, pp. 4185–4190.

[8] D. P. Bertsekas, A. E. Ozdaglar, and A. Nedi, *Convex analysis and optimization*, ser. Athena scientific optimization and computation series. Belmont (Mass.): Athena Scientific, 2003.

[9] H. Zhu, A. Cano, and G. Giannakis, "Distributed Consensus-based Demodulation: Algorithms and Error Analysis," *IEEE Transactions on Wireless Communications*, vol. 9, no. 6, pp. 2044–2054, 2010.

[10] H. Paul, J. Fliege, and A. Dekorsy, "In-Network-Processing: Distributed Consensus-Based Linear Estimation," *IEEE Communications Letters*, vol. 17, no. 1, pp. 59–62, 2013.

[11] I. Schizas, A. Ribeiro, and G. Giannakis, "Consensus in Ad Hoc WSNs With Noisy Links part I: Distributed Estimation of Deterministic Signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, Jan. 2008.

[12] H. Zhu, G. Giannakis, and A. Cano, "Distributed in-network channel decoding," *IEEE Transactions on Signal Processing*, vol. 57, no. 10, Oct. 2009.

[13] D. Wübben, H. Paul, B.-S. Shin, G. Xu, and A. Dekorsy, " Distributed Consensus-based Estimation for Small Cell Cooperative Networks," in *10th International Workshop on Broadband Wireless Access (BWA 2014), co-located with IEEE Globecom 2014*, Austin, TX, USA, 2014.

[14] G. Xu, H. Paul, D. Wübben, and A. Dekorsy, "Fast Distributed Consensus-based Estimation for Cooperative Wireless Sensor Networks," in *18th International ITG Workshop on Smart Antennas (WSA 2014)*, Erlangen, Germany, Mar. 2014.

[15] I. Necoara, I. Dumitrache, and J. Suykens, "Fast primal-dual projected linear iterations for distributed consensus in constrained convex optimization," in *49th IEEE Conference on Decision and Control (CDC)*, Dec. 2010, pp. 1366–1371.

[16] G. Xu, H. Paul, D. Wübben, and A. Dekorsy, "Distributed Augmented Lagrangian Method for Cooperative Estimation in Small Cell Networks," in *10th International ITG Conference on Systems, Communications and Coding (SCC 2015)*, Hamburg, Germany, Feb. 2015.

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

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[19] D. M. Young and W. Rheinboldt, *Iterative solution of large linear systems*. New York, USA: Academic Press, 1971.

[20] D. P. Bertsekas, "Multiplier methods: A survey," *Automatica*, vol. 12, no. 2, pp. 133–145, 1976.

[21] R. Merris, "Laplacian matrices of graphs: a survey," *Linear Algebra and its Applications*, vol. 197-198, pp. 143–176, 1994.