Distributed Consensus-Based Linear Estimation with Erroneous Links

Henning Paul, Ban-Sok Shin, and Armin Dekorsy
Department of Communications Engineering
University of Bremen
Bremen, Germany
Email: paul@ant.uni-bremen.de, shin@ant.uni-bremen.de, dekorsy@ant.uni-bremen.de

***Abstract*—In a cooperative broadcast scenario, a group of nodes in a network aims to reconstruct a common message. We present a consensus-based linear estimation algorithm that shows improved convergence speed compared to state of the art techniques and whose signaling effort can be reduced without performance loss under certain conditions. We investigate its performance in the presence of erroneous inter-node links using different models for these error influences. Additionally, we propose a technique to mitigate the error influences and restore satisfactory overall system performance. All results are corroborated by computer simulations considering different system parameters and network setups.**

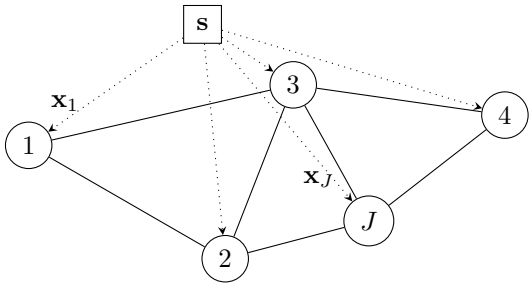

Fig. 1. A network of J nodes receiving different observations $\mathbf{x}_j$ of the same quantity $\mathbf{s}$.

I. Introduction

Cooperative communication in wireless networks has received much attention in the research community through the past years. A common scenario is the cooperative broadcast case, in which a group of nodes, connected via inter-node links, intends to recover a common message broadcasted by a detached station. The amount of information on the message locally available at the sensing nodes often does not suffice, so that cooperation among the nodes is necessary for successful reconstruction. While it is possible to perform centralized reconstruction in a dedicated Data Fusion Center, this would be a single point of failure. Distributed algorithms promise higher robustness against node and link failures. By performing some amount of processing at nodes and employing inter-node communication, we aim for obtaining the centralized solution through in-network processing. Recently, we presented an distributed, iterative algorithm [1] that achieves this goal and establishes a consensus, i.e., that eventually, the centralized solution is available at all nodes. Compared to other algorithms from literature for consensus-based distributed estimation (e.g., [2], [3], [4]), our approach exhibits the advantage that every communication of nodes with its neighbors can be realized in a broadcast fashion only, resulting in relatively low communication effort, while still showing fast convergence. In this paper, we will investigate the algorithm's behavior in the presence of different classes of erroneous inter-node links; with additive errors as well as link failures. Furthermore, we will propose countermeasures against these effects, in particular, a digital filtering technique for denoising. Finally, we will introduce an approximation in order to reduce the required communication effort.

The remainder of this paper is structured as follows: In section II, we introduce the system model and the estimation problem formulation. Also, we will describe our distributed consensus-based linear estimation algorithm that will be used in the following. In section III, we will introduce the error models that are used to describe the error influences on the inter-node links. In the following section IV, simulation results will be presented. Here the presented error models are applied to networks in which the distributed consensus algorithm is implemented. Section V proposes a filtering technique on the variables exchanged during the execution of the distributed algorithm in order to mitigate the effects of erroneous variable exchange. Finally, section VI concludes the paper and provides an outlook to future work.

II. Problem formulation

Fig. 1 illustrates the cooperative broadcast scenario. J nodes are connected with inter-node links, forming a sensor network with adjacency matrix $\mathbf{A}$. In the following, we assume that the graph describing the network is connected, so every node is able to reach every other node, albeit using several hops. At every node j, knowledge of the original message $\mathbf{s}$ is desired, but only a disturbed observation $\mathbf{x}_j$ of it is available. The relationship between these variables is modeled as linear, using a disturbance matrix[1] $\mathbf{H}_j$ and an additional noise term $\mathbf{n}_j$. This results in the linear equation

$$\mathbf{x}_j = \mathbf{H}_j\mathbf{s} + \mathbf{n}_j \tag{1}$$

with $\mathbf{s} \in \mathbb{R}^{N\times 1}$, $\mathbf{x}_j \in \mathbb{R}^{M\times 1}$, $\mathbf{H}_j \in \mathbb{R}^{M\times N}$ and $\mathbf{n}_j \in \mathbb{R}^{M\times 1}$, where N and M denote the dimensions of message and observation vectors respectively.

[1] In communications applications, e.g., this corresponds to the source-sensor channel matrix.

To reconstruct $\mathbf{s}$ in (1), individual Least Squares (LS) estimation per node can be employed, resulting in J local estimates $\hat{\mathbf{s}}_j$:

$$\hat{\mathbf{s}}_j = \arg\min_{\mathbf{s}'} ||\mathbf{x}_j - \mathbf{H}_j\mathbf{s}'||^2. \tag{2}$$

In general, these estimates will differ over j and therefore cannot represent a consensus solution: The presence of noise or a rank deficiency of $\mathbf{H}_j$ leads to poor estimation, in particular if $M < N$ and (1) therefore is underdetermined. In order to exploit the entire information on $\mathbf{s}$ available in the network and find a common, unique solution, the centralized problem

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}'} ||\mathbf{x}^{\mathrm{s}} - \mathbf{H}^{\mathrm{s}}\mathbf{s}'||^2 \tag{3}$$

using the stacked observation vector $\mathbf{x}^{\mathrm{s}} = \left[\mathbf{x}_1^{\mathrm{T}}, \ldots, \mathbf{x}_J^{\mathrm{T}}\right]^{\mathrm{T}}$ and stacked disturbance matrix $\mathbf{H}^{\mathrm{s}} = \left[\mathbf{H}_1^{\mathrm{T}}, \ldots, \mathbf{H}_J^{\mathrm{T}}\right]^{\mathrm{T}}$ needs to be solved, e.g. through centralized processing in a data fusion center. Obviously, to facilitate this, the sensors' observations and disturbance matrices need to be forwarded to this central processing node, necessitating routing protocols and communication effort. Furthermore, an outage of the fusion center will cause the whole network to fail, making it a single point of failure.

Therefore, we are aiming to solve the centralized problem (3) in a distributed fashion within the network. Recently, we proposed an iterative algorithm that is able to achieve this aim [1]. It is based on the solution of the optimization problem (3) through the Augmented Lagrangian method [5] and the Alternating Direction Method of Multipliers [6], [7]. In order to facilitate this solution, the stacked problem is reformulated as a set of local optimization problems which are coupled through equality constraints:

$$\begin{aligned}\{\hat{\mathbf{s}}_j | j \in \mathcal{J}\} = \arg \min_{\{\mathbf{s}_j | j \in \mathcal{J}\}} \sum_{j=1}^{J} ||\mathbf{x}_j - \mathbf{H}_j\mathbf{s}_j||^2 \\ \text{s.t.} \quad \mathbf{s}_j = \mathbf{s}_i \quad \forall i \in \mathcal{N}_j.\end{aligned}$$

Upon final solution of the optimization problem, the local estimate $\mathbf{s}_j$ of every node j in the set $\mathcal{J}$ of all nodes is supposed to be identical with the estimates $\mathbf{s}_i$ of all nodes i in its (graph theoretic) neighborhood $\mathcal{N}_j$. $\mathcal{N}_j$ can be determined by the positions of the nonzero elements of the j-th row of the network's adjacency matrix $\mathbf{A}$. Obviously, the constraints cause a direct coupling of the variables $\mathbf{s}_j$, so it is not possible to parallelize this optimization problem without modification. Our approach introduces auxiliary variables $\mathbf{z}_j$ at each node, which allows for a distribution of the local optimization problems among the nodes: The single constraint $\mathbf{s}_j = \mathbf{s}_i$ is replaced with 2 constraints $\mathbf{s}_j = \mathbf{z}_j$, $\mathbf{z}_j = \mathbf{s}_i$, so the auxiliary variable $\mathbf{z}_j$ assumes the role of an "intermediate" estimate and thus allows for a decoupling of the local optimization problems.

After straightforward calculations which can be found in [1], the following update equations for the local estimates $\mathbf{s}_j$, the auxiliary variables $\mathbf{z}_j$ and a set of Lagrange multipliers $\boldsymbol{\lambda}_{i,j}$ introduced in the course of the optimization are obtained:

$$\begin{aligned}\mathbf{s}_j(k+1) = &\left(\mathbf{H}_j^{\mathrm{T}}\mathbf{H}_j + \frac{|\mathcal{N}_j|+1}{\mu}\mathbf{I}\right)^{-1} \\ &\cdot \left[\mathbf{H}_j^{\mathrm{T}}\mathbf{x}_j + \sum_{i \in \mathcal{N}_j \cup j} \left(\boldsymbol{\lambda}_{j,i}(k) + \frac{1}{\mu}\mathbf{z}_i(k)\right)\right],\end{aligned} \tag{4}$$

$$\mathbf{z}_j(k+1) = \frac{\mu}{|\mathcal{N}_j|+1} \sum_{i \in \mathcal{N}_j \cup j} \left[-\boldsymbol{\lambda}_{i,j}(k) + \frac{1}{\mu}\mathbf{s}_i(k+1)\right], \tag{5}$$

$$\boldsymbol{\lambda}_{i,j}(k+1) = \boldsymbol{\lambda}_{i,j}(k) - \frac{1}{\mu}(\mathbf{s}_i(k+1) - \mathbf{z}_j(k+1)). \tag{6}$$

μ is a parameter that allows for control of the step size and therefore the convergence speed.

These equations illustrate that for the update of the local estimate $\mathbf{s}_j(k+1)$ in the $k+1$st iteration, the auxiliary variables $\mathbf{z}_i(k)$ of the previous iteration from all the nodes in the neighborhood $\mathcal{N}_j$ (and its own) have to be known, so do the Lagrange multipliers $\boldsymbol{\lambda}_{i,j}(k)$. Similarly, the subsequent update of $\mathbf{z}_j(k+1)$ requires knowledge of the previously updated local estimates $\mathbf{s}_i(k+1)$ in the node's neighborhood. Finally, the update of the Lagrange multipliers $\boldsymbol{\lambda}_{i,j}(k+1)$ relies on the previously calculated local estimates and auxiliary variables. This implies that all variables need to be exchanged with neighboring nodes immediately after they have been updated. Since $\mathbf{s}_j$ and $\mathbf{z}_j$ are only specific to their originating node and not to the node at which they are processed, it is sufficient to share these in a broadcast fashion. This does not hold for the Lagrange variables $\boldsymbol{\lambda}_{i,j}$, which are specific to only one edge of the network graph, and in particular only to one direction of this edge. But since the update of $\boldsymbol{\lambda}_{i,j}(k+1)$ only depends on its previous value and the variables $\mathbf{s}_i(k+1)$ and $\mathbf{z}_j(k+1)$ which were already received previously within the $k+1$st iteration, it is possible to perform the update of all required $\boldsymbol{\lambda}_{i,j}$ and $\boldsymbol{\lambda}_{j,i}$ locally. Later we will show that this simplification is only allowed in the case of error-free exchange of variables.

III. Erroneous Networks

The basic performance of the algorithm, in particular compared with [2], has been presented in [1]. These investigations restricted on the case of error-free exchange of information among the nodes. Nevertheless, in practical wireless networks, communication among nodes in general is imperfect: Depending on the amount of effort spent on error correction, messages, i.e., variables in our case, might get corrupted. This corruption can either appear as an additional error on the exchanged variable or as a loss of a variable on a certain link among two nodes j, i. These error models are widely accepted and have been employed for the analysis of wireless sensor networks numerous times, e.g. in [3], also in [2] and [8], a general model employing graph theoretic analysis was presented in [9]. Therefore, we will use these model also for our subsequent considerations. In particular, the three following cases will be analyzed:

a) Additional errors on the inter-node links: The exchanged variables $\mathbf{s}_j(k)$, $\mathbf{z}_j(k)$ and (if actually exchanged) $\boldsymbol{\lambda}_{i,j}(k)$ are superimposed with an additional error. This error is

modeled to be gaussian distributed and dependent on the inter-sensor link's SNR. Since the error is superimposed on the link between the nodes, each neighboring node of j will receive a replica of $\mathbf{s}_j(k)$ with an individual, mutually uncorrelated noise term. Consequently, the received noisy variable shall be denoted as $\tilde{\mathbf{s}}_{j,i}(k) = \mathbf{s}_j(k) + \mathbf{n}^{\mathrm{s}}_{j,i}(k)$. For the rest of this paper, we will model the elements of the additional error $\mathbf{n}^{\mathrm{s}}_{j,i}(k)$ as gaussian distributed, zero-mean and uncorrelated.

b) Inter-node link failures: If an error correction mechanism is implemented on the inter-node links, it is possible to discard a received message if it has been corrupted during the transmission, e.g. through noise or interference. In this case, occuring with a certain probability p_{f}, this message (in our algorithm $\mathbf{s}_j(k)$, $\mathbf{z}_j(k)$ or $\boldsymbol{\lambda}_{i,j}(k)$) is not available for use in the update equations. This effect can be modeled by a time-dependent adjacency matrix $\mathbf{A}(k)$ and thus, a time-dependent neighborhood $\mathcal{N}_j(k)$ of the node j. Of course, the case that within one iteration k the transmission of $\mathbf{s}_i(k)$ to node j succeeds, while the transmission of $\mathbf{z}_i(k)$ fails cannot be covered without further modification. We expect that such an irregular update will result in a different network performance compared to the case where $\mathcal{N}_j(k)$ is constant over one iteration, but this case is not considered in this paper.

c) Combined effects: Above error models can be combined, representing a network, in which the communication between nodes is never error-free and might fail with a certain probability p_{f}.

IV. Simulation Results

We investigate the convergence behavior of the algorithm (4) – (6) by means of numerical simulations. As figures of merit, the required number of iterations and the resulting overall square error

$$\mathrm{OSE}(k) = \sum_{j=1}^{J} ||\mathbf{s} - \mathbf{s}_j(k)||^2 \tag{7}$$

between the nodes' estimates and the true value at iteration k are employed. The stopping criterion for the iterative processing is based on the gradient of the Lagrangian cost function. The sum of the squared norms of the cost functions gradients w.r.t. the variables is compared to a threshold τ:

$$\sum_{j=1}^{J} ||\nabla_{\mathbf{s}_j} \mathcal{L}(\mathbf{s}, \mathbf{z}, \boldsymbol{\lambda})||^2 < \tau^2. \tag{8}$$

In the following, the value $\tau = 0.1$ shall be employed. It has been determined experimentally and shows good results.

As a reference for performance comparison, we use the algorithm presented in [2], whose parameters have been chosen such that a fair comparison is assured, the stopping criterion was also chosen identical. The algorithm parameter μ was chosen to the value 1. The networks in which the consensus algorithms are running have been generated randomly: J nodes are placed onto a unit square randomly, following a 2-dimensional uniform distribution. If the euclidian distance between two nodes i, j falls below a threshold $r_{\max}$, the two nodes are modeled to share an inter-node link and the corresponding element of the adjacency matrix $\mathbf{A}$ is set to 1 and to 0 otherwise. In order to ensure that the resulting graph is connected, the so-called Fiedler eigenvalue [10] is investigated: This second smallest eigenvalue of the graph's Laplacian matrix is larger than 0 if the graph is connected[2].

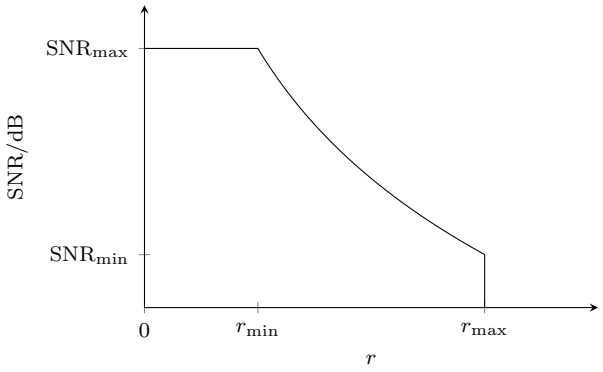

Fig. 2. SNR model based on path loss approximation with path loss exponent $\alpha = 2$

For the properties of the inter-node links, different models, based on the error models above are used: For the additional error introduced above, the SNR is either equal on all inter-node links or different depending on the distance between the nodes using a path loss approximation. This approximation is depicted in Fig. 2. If the distance r between two nodes lies in a range $r_{\min}$ and $r_{\max}$, the (linear) SNR is proportional to $r^{-\alpha}$ with α being the path loss exponent. If r is smaller than $r_{\min}$, the SNR saturates (due to the receiver's impairments) at $\mathrm{SNR}_{\max}$. If r exceeds $r_{\max}$, as stated above, the two nodes are assumed not to share a communication link. If additional link failures occur with a probability of p_{f}, no communication is possible on the link, regardless of the distance of the nodes.

A. Link failures

Fig. 3 shows the cumulative density function (CDF) of the required number of iterations n_{req} for a random network consisting of $J = 6$ nodes with $r_{\max} = 0.3$ for 1000 random realizations of $\mathbf{s}$, $\mathbf{H}_j$ and $\mathbf{n}_j$ with $M = 5$ and $N = 2$. As elements for $\mathbf{s}$ and $\mathbf{H}_j$, real valued, zero mean gaussian distributed values of unit variance were chosen. The elements of the noise vectors $\mathbf{n}_j$ were also taken from a real valued, zero mean gaussian distribution, but with an exemplary variance of 0.1, resulting in an SNR of $10\,\mathrm{dB}$ on the source-sensor links. The failure probability of the inter-node links was chosen to $p_{\mathrm{f}} = 0.4$, with the state of the link randomly determined for every iteration. On non-failing links, the information exchange was assumed to be error-free. It can be seen that the convergence of the algorithm in [2] is slowed down significantly by such link failures. In contrast to our algorithm, the average number of required iterations is increased only slightly, whereas the minimum number of iterations does not change significantly, indicating a high robustness.

[2]Its value is a measure for the connectedness of the graph and an indicator for the convergence speed of the algorithm. Investigations on this issue are subject of future work.

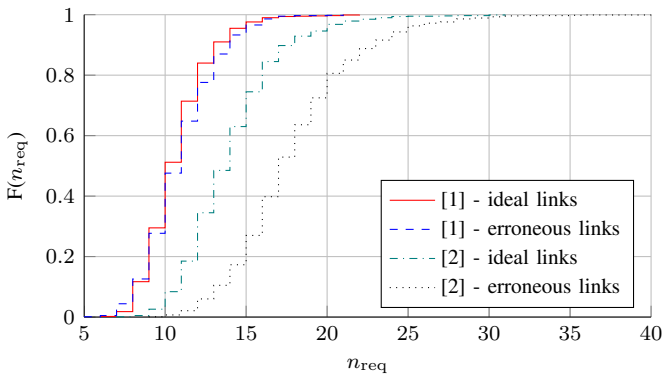

Fig. 3. Empirical cumulative density function (CDF) of required number of iterations until fulfillment of stopping criterion over 1000 random signal, channel and noise realizations in a randomly generated network with $J = 6$ nodes

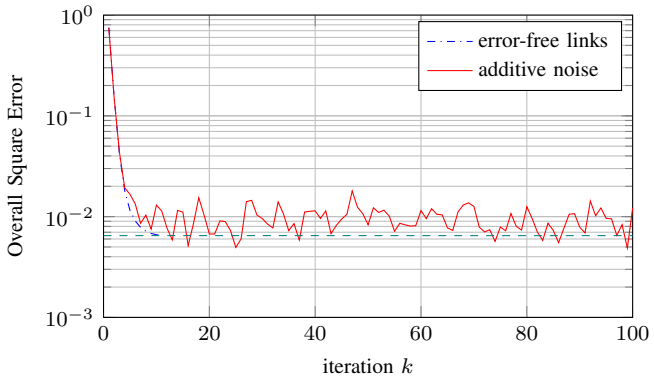

Fig. 4. Convergence behavior of the algorithm [1] in the presence of additive noise with centralized solution as reference (dashed line) in a randomly generated network with $J = 6$ nodes

B. Noisy links

While random link failures alone do not cause a significant deterioration of the convergence behavior, this is not the case for additive random errors on the exchanged messages. Figure 4 shows the behavior of our algorithm in a network with same parameters as above, but with perfect, error free information exchange among the nodes in the first case and the path loss-based SNR model in the second case, with $\mathrm{SNR}_{\max} = 20\,\mathrm{dB}$, $\alpha = 2$, $r_{\min} = 0.1$ and $r_{\max} = 0.3$ and no link failures. Depicted is the OSE over iteration k. While for the error free case the LS solution (depicted as dashed line) is achieved very fast, the erroneous exchange of $\mathbf{s}_j(k)$, $\mathbf{z}_j(k)$ and $\boldsymbol{\lambda}_{i,j}(k)$ leads to a noisy behavior, i.e. the OSE varying around the LS solution, due to which the stopping criterion is never fulfilled. Therefore, in such scenarios, the number of iterations must be limited to a sensible number. However, we can evaluate the resulting (average) OSE for this number of iterations and use it as a figure of merit for comparison of the different algorithms.

Please note that in such a scenario, the exchange of the Lagrange multipliers $\boldsymbol{\lambda}_{i,j}(k)$ is required. If these are all updated locally, based on erroneously received variables $\tilde{\mathbf{s}}_{j,i}(k)$ and $\tilde{\mathbf{z}}_{j,i}(k)$, our investigations indicated that no convergence is achieved.

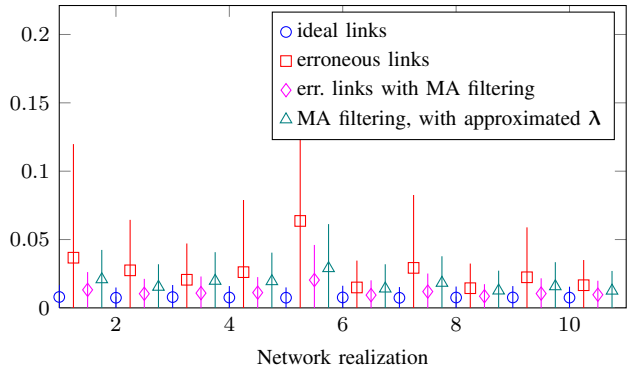

Fig. 5. Mean OSE ± standard deviation after 25 iterations for 10 different, randomly generated networks with $J = 6$ nodes, based on 1000 realizations of signal, channel and noise each

C. Combined link errors

In the following, the combined error model, comprising both additive errors and complete link failures shall be investigated. For this purpose, above path loss-based SNR model is extended with the time-variant neighborhood $\mathcal{N}_j(k)$ introduced in the previous section. Like above, the link failure probability was set to $p_{\mathrm{f}} = 0.4$.

Fig. 5 shows the mean and its standard deviation of the OSE after 25 iterations for 10 different, randomly generated networks with $J = 6$ nodes, based on 1000 realizations of signal, channel and noise each. While the error free, ideal link case results in a very small average OSE with little variation, the erroneous links cause quite large mean errors with also significantly increased standard deviation. Therefore, in order to mitigate this effect, we will investigate the possibility of denoising using digital filtering techniques in the following.

V. Filtering of Exchanged Variables

We propose a weighted moving average (MA) filtering in temporal direction for the denoising of the $\tilde{\mathbf{s}}_{j,i}(k)$ and $\tilde{\mathbf{z}}_{j,i}(k)$ variables exchanged among nodes. Of course, this approach reduces the convergence speed, since older, outdated values of the variable from the previous iterations are included in every new update, so a trade-off between noise reduction and reduction of convergence speed has to be found. Fig. 5 shows the resulting mean OSE after 25 iterations for a MA filter of length 10 with linearly decreasing weights, weighting $\tilde{\mathbf{s}}_{j,i}(k-1)$ most and $\tilde{\mathbf{s}}_{j,i}(k-10)$ least[3]. A normalization to a sum of 1 for the coefficients of the MA filter was applied to ensure convergence. This is motivated as follows: In the equilibrium state, the variables do not change over iteration k anymore, therefore the weighted sum of 10 past values equals the equilibrium value of this variable multiplied by the sum of the filter coefficients. In order not to introduce a bias, the sum must be normalized as stated above. Further attention must be paid to the fact that due to link failures, certain past values $\tilde{\mathbf{s}}_{j,i}$ and $\tilde{\mathbf{z}}_{j,i}$ might not be available. To this end, our algorithm uses a FIFO (First-In-First-Out) buffer that stores the 10 most recent values, the MA filtering is always applied on these 10 most recent values, except for the very first iterations of the

[3] $\tilde{\mathbf{z}}_{j,i}(k)$ correspondingly.

algorithm where the buffer is filled. Here, the MA filtering is only applied on available values (with respectively corrected filter coefficients). The alternative approach of using at most 10 values for filtering, and less if some have been lost due to link outages, showed slightly worse performance and therefore is not discussed here.

Evaluating the resulting mean OSE and especially its standard deviation as depicted in Fig. 5 and comparing it to the noisy case without filtering, it can be seen that these figures of merit can be reduced significantly using this weighted MA approach, improving the achievable estimation accuracy.

A. Approximation for reduction of communication overhead

Additionally, in order to reduce the transmission overhead among the nodes, we propose the use of the local Lagrange multipliers $\boldsymbol{\lambda}_{j,j}(k)$ at each node j only instead of the multipliers $\boldsymbol{\lambda}_{i,j}(k)$. In this way, only variables $\mathbf{s}_j(k)$ and $\mathbf{z}_j(k)$ need to be exchanged among the nodes. The resulting mean OSE is also depicted in Fig. 5. It can be seen that there is only a slight degradation in the error performance compared to the case of exchanged multipliers[4]. This reduces the communication effort significantly, since the exchange of $\boldsymbol{\lambda}_{i,j}(k)$ is particularly costly: In constrast to the node-specific variables $\mathbf{s}_j(k)$ and $\mathbf{z}_j(k)$, which can be broadcasted, the $\boldsymbol{\lambda}_{i,j}(k)$ variables are edge-specific and therefore have to be exchanged between nodes in a unicast fashion.

VI. Conclusion and Future Work

In this paper, we have investigated the behavior of a novel algorithm for the distributed consensus-based linear estimation in the presence of erroneous inter-node links, like those commonplace in wireless sensor networks. We have presented models for these kind of errors and proposed a filtering technique for their mitigation. In future work, we will approach the problem of erroneous inter-node links through a more robust approach w.r.t. the optimization criterion which is the basis of our algorithm.

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 317941. The authors would like to acknowledge the contributions of their colleagues in iJOIN, although the views expressed are those of the authors and do not necessarily represent the project.

References

[1] H. Paul, J. Fliege, and A. Dekorsy, "In-network-processing: Distributed consensus-based linear estimation," *IEEE Commun. Lett.*, vol. 17, no. 1, pp. 59–62, Jan. 2013.

[2] H. Zhu, A. Cano, and G. Giannakis, "Distributed consensus-based demodulation: algorithms and error analysis," *IEEE Trans. Wireless Commun.*, vol. 9, no. 6, pp. 2044–2054, Jun. 2010.

[3] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[4]The thorough analysis of the consequences of this approximation on the resulting estimate will be covered in future work.

[4] F. Cattivelli and A. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[5] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, USA: Springer Science+Business Media, 2006.

[6] 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.

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

[8] H. Zhu, A. Cano, and G. B. Giannakis, "Distributed in-network channel decoding," *IEEE Trans. Signal Process.*, vol. 57, no. 10, Oct. 2009.

[9] R. Rajagopal and M. Wainwright, "Network-based consensus averaging with general noisy channels," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 373 –385, Jan. 2011.

[10] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 98, pp. 298–305, 1973.