Consensus Distributed Conjugate Gradient Algorithms for Parameter Estimation over Sensor Networks

Tamara Guerra Miller[1], Songcen Xu[2] and Rodrigo C. de Lamare[1,2]

[1]CETUC, PUC-Rio, Brazil

[2]Department of Electronics, University of York, United Kingdom

***Abstract*—This paper proposes distributed adaptive algorithms based on the conjugate gradient (CG) method and the consensus strategy for parameter estimation over sensor networks. In particular, we present a conventional distributed CG algorithm and a distributed CG algorithm that exploits sparsity in the set of parameters using l_1 and *log-sum* penalty functions. The proposed consensus distributed CG (Consensus-CG) algorithm has an improved performance in terms of mean square deviation (MSD) and convergence as compared with the consensus least-mean square (Consensus-LMS) algorithm and a close performance to the consensus distributed recursive least-squares (Consensus-RLS) algorithm. Similar results are obtained with the proposed sparsity-aware consensus distributed CG algorithm. Numerical results show that the proposed algorithms are reliable and can be applied in several scenarios.**

Keywords*— *Distributed Processing, Conjugate Gradient, Sparsity Aware.

I. INTRODUCTION

For several years, sensor networks have been applied in medicine, industry, agriculture, etc. Distributed processing has become a very common and useful approach to extract information in a network by performing estimation of the desired parameters. The efficiency of the network depends on the communication protocol used to exchange information between the nodes, as well as the algorithm to obtain the parameters. Another important aspect is to prevent a failure in any agent that may affect the operation and the performance of the network. Similar to a single node adaptive processing, the performance of the network may vary in time. Distributed schemes can offer better estimation performance of the parameters as compared with the centralized approach, based on the principle that each node communicates with the other nodes and exploits the spatial diversity in the network [1].

The main strategies for communication in distributed processing are incremental, consensus and diffusion. In the incremental protocol, the communication flows cyclically and the information is exchanged from one node to the adjacent nodes. In this strategy the flow of information must be preset at the initialization [2]. In the diffusion mechanism, each node communicates with the rest of the nodes [3]. The consensus strategy is an elegant procedure to enforce agreement among cooperating nodes [4].

In many scenarios, the impulse responses of unknown systems can be assumed to be sparse, containing only a few large coefficients interspersed among many negligible ones [5]. Many studies have shown that exploiting the sparsity of a system is beneficial to enhancing the estimation performance [6]. Most of the studies developed for distributed processing exploiting sparsity are focused on the least-mean square (LMS) and recursive least-squares (RLS) algorithms using different penalty functions [7]-[9]. These penalty functions perform a regularization that attracts to zero the coefficients of the parameter vector that are not associated with the weights of interest. The most well-known and exploited penalty functions are the l_0-norm, the l_1-norm and the log-sum [10]. With these techniques a better network performance is achieved, in the presence of sparsity in the set of parameters.

The Conjugate Gradient (CG) algorithm [11] has been studied and developed for distributed processing, using the diffusion strategy [11], which often results in algorithms that are more computationally complex than consensus techniques. The faster convergence performance of CG algorithms over the LMS algorithm and its lower computational complexity and better numerical stability than the RLS algorithm makes it suitable for this task. However, prior work on distributed CG techniques is rather limited as a consensus-type algorithm and techniques that exploit possible sparsity of the signals have not been developed so far.

In this paper we propose distributed CG algorithms based on the consensus strategy for parameter estimation over sensor networks. Specifically, we develop a distributed CG algorithm using the consensus protocol and a sparsity-aware CG algorithm with l_1 and log-sum penalty functions. The proposed algorithms are compared with recently reported algorithms in the literature. The particular application presented in this paper is parameter estimation over sensor networks, which can be found in many scenarios of practical interest.

This paper is organized as follows. Section II describes the system model and the problem statement. Section III presents the proposed distributed consensus CG algorithm. Section IV details the proposed sparsity-aware distributed consensus CG algorithm. Section V presents and discusses the simulation results. Finally, Section VI gives the conclusions and discusses possible future directions.

Notation: In the following parts of this paper, matrices and vectors are denoted by boldface upper case letters and boldface lower case letters, respectively. The superscript $(\cdot)^H$ denotes the Hermitian operator. The $\|\cdot\|_1$ denotes the l_1 norm and the $E[\cdot]$ denotes the expectation operator.

II. SYSTEM MODEL AND PROBLEM STATEMENT

In this section, we describe the system model of the distributed estimation scheme and introduce the problem statement.

A. System Model

The network consists of N nodes that exchange information between them, where each node represents an adaptive parameter vector with neighborhood described by the set N_k. The main task of the parameter estimation problem is to adjust the unknown $M \times 1$ weight vector $\boldsymbol{\omega}_k$ of each node, where M is the length of the filter [1]. The desired signal $d_{k,i}$ at each time i is a scalar random process given by

$$d_{k,i} = \boldsymbol{\omega}_0{}^H \mathbf{x}_{k,i} + n_{k,i}, \quad (1)$$

where $\boldsymbol{\omega}_0$ is the M x 1 system weight vector, $\mathbf{x}_{k,i}$ is the $M \times 1$ input signal vector and $n_{k,i}$ is the measurement noise. The output estimate is given by

$$y_{k,i} = \boldsymbol{\omega}_{k,i}{}^H \mathbf{x}_{k,i}, \quad (2)$$

The main goal of the network is to minimize the following cost function:

$$C(\boldsymbol{\omega}_{k,i}) = \sum_{k=1}^{N} \mathrm{E}\left[\left|d_{k,i} - \boldsymbol{\omega}_{k,i}{}^H \mathbf{x}_{k,i}\right|^2\right]. \quad (3)$$

By solving this minimization problem it is possible to obtain the optimum solution of the weight vector at each node. The optimum solution for the cost function is given by

$$\boldsymbol{\omega}_{k,i} = \mathbf{R}_{k,i}{}^{-1} \mathbf{b}_{k,i}, \quad (4)$$

where $\mathbf{R}_{k,i}$ =E[$\mathbf{x}_{k,i}\, \mathbf{x}_{k,i}^H$] is the M x M correlation matrix of the input data vector $\mathbf{x}_{k,i}$, and $\mathbf{b}_{k,i}$ =E[$d_{k,i}\, \mathbf{x}_{k,i}$] is the M x 1 cross-correlation vector between the input data and the desired response $d_{k,i}$.

B. Problem Statement

We consider a consensus algorithm for a network where each agent k has access at each time instant to the realization $\{d_{k,i}, \mathbf{x}_{k,i}\}$ of zero-mean spatial data $\{d_k, \mathbf{x}_k\}$[12][15].

For a network with possibly sparse parameter vectors, the cost function also involves a penalty function which exploits sparsity. In this case the network needs to solve the following optimization problem:

$$\min C(\boldsymbol{\omega}_{k,i}) = \sum_{k=1}^{N} \mathrm{E}\left[\left|d_{k,i} - \mathbf{x}_{k,i}\boldsymbol{\omega}_{k,i}\right|^2\right] + f(\boldsymbol{\omega}_{k,i}), \quad (5)$$
$$\text{subject to } \boldsymbol{\omega}_k = \boldsymbol{\omega}_m, k = 1,2,..., K, m \in N_k$$

where $f(\boldsymbol{\omega}_{k,i})$ is a penalty function that exploits the sparsity in the parameter vector $\boldsymbol{\omega}_{k,i}$. The consensus cooperation strategy is a constrained optimization problem that enforces the equality of the parameter vectors $\boldsymbol{\omega}_{k,i}$ [17], which means all network agents converge to the same weight value, i.e., $\boldsymbol{\omega}_k = \boldsymbol{\omega}_m, k = 1,2,\ldots,K, m \in N_k$.

In the following sections we focus on distributed CG versions of the consensus protocol algorithm to solve (5).

III. PROPOSED DISTRIBUTED CONSENSUS CG

In this section, we present the proposed distributed CG algorithm using the consensus strategy with a penalty function that is equal to zero. This corresponds to the consensus strategy without the exploitation of sparsity. We first derive the CG algorithm and then consider the consensus protocol.

A. Derivation of the CG algorithm

The CG method can be applied to adaptive filtering problems [11] [16]. The main objective in this task is to solve equation (4). The cost function of the CG algorithm for one agent is given by

$$C_{CG}(\boldsymbol{\omega}) = \frac{1}{2}\boldsymbol{\omega}^H \mathbf{R}\boldsymbol{\omega} - \mathbf{b}^H \boldsymbol{\omega}. \quad (6)$$

For distributed processing over sensor networks, we present the following derivation. The CG algorithm does not need to solve the matrix inversion of $\mathbf{R}$, which is an advantage as compared with RLS algorithms. It computes the weights $\boldsymbol{\omega}_{k,i}$ for each iteration j until convergence, i.e., $\boldsymbol{\omega}_{k,i}$ (j). The gradient of the method in the negative direction is obtained as follows [11]:

$$\mathbf{g}_{k,i}(j) = \mathbf{b}_{k,i}(j) - \mathbf{R}_{k,i}(j)\boldsymbol{\omega}_{k,i}(j) \quad (7)$$

Calculating the Krylov subspace [13] through different operations, the recursion is given by

$$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\omega}_{k,i}(j-1) - \alpha(j)\mathbf{p}_{k,i}(j), \quad (8)$$

where $\mathbf{p}$ is the conjugate direction vector of $\mathbf{g}$ and α is the step size that minimizes the cost function in (6) by replacing (7) in (4). Both parameters are calculated as follows

$$\alpha(j) = \frac{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)}{\mathbf{p}_{k,i}{}^H(j)\mathbf{R}_{k,i}(j)\mathbf{p}_{k,i}(j)}, \quad (9)$$

$$\mathbf{p}_{k,i}(j+1) = \mathbf{g}_{k,i}(i) + \beta(j)\mathbf{p}_{k,i}(j), \quad (10)$$

The parameter β is calculated using the Gram-Schmidt orthogonalization procedure [14].

$$\beta(j) = \frac{\mathbf{g}_{k,i}(j)^H \mathbf{g}_{k,i}(j)}{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)} \quad (11)$$

Applying the CG method to a distributed network the cost function is expressed based on the information exchanged between all nodes $k=1, 2 \ldots N$. Each of the equations presented so far takes place at each agent during the iterations of the CG algorithm. Therefore, we have the cost function:

$$C_{DCG}(\boldsymbol{\omega}_{k,i}) = \frac{1}{2}\sum_{k=1}^{N} \boldsymbol{\omega}_{k,i}^H \mathbf{R}_{k,i} - \sum_{k=1}^{N} \mathbf{b}_{k,i}{}^H \boldsymbol{\omega}_{k,i} \quad (12)$$

Using the data window with an exponential decay, the resulting autocorrelation and cross-correlation matrices are defined using the λ parameter, which is the same as the forgetting factor of the RLS algorithm. The correlation and cross-correlation functions are given by

$$\mathbf{R}_{k,i} = \lambda \mathbf{R}_{k,i-1} + \mathbf{x}_{k,i}\mathbf{x}_{k,i}^H \quad (13)$$

$$\mathbf{b}_{k,i} = \lambda \mathbf{b}_{k,i} + d_k^* \mathbf{x}_{k,1} \quad (14)$$

B. Consensus Distributed CG algorithm

In the consensus strategy, all nodes interact with their neighbors sharing and reaching agreement about the system parameter vector. Each node k is able to run its update simultaneously with the other agents [1] [3]. Fig.1 illustrates the consensus strategy.

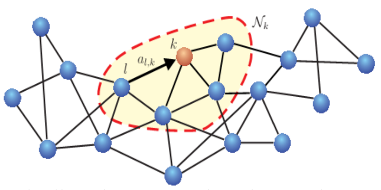

Fig. 1 Distributed consensus-based network processing

In consensus techniques there is a cooperation factor that is a convex combination of the iterations available at the neighborhood of agent k. This combination is then updated with the previous value of the node. This mechanism performs adaptation and learning at the same time [3].

For consensus distributed algorithms the combination step is based on the connectivity among nodes, where the local estimation is given by

$$\boldsymbol{\varphi}_{l,i-1} = \sum_{l \in N_k} a_{lk} \boldsymbol{\omega}_{l,i-1}, \quad (15)$$

where a_{lk} represents the combining coefficients of the data fusion which should comply with

$$\sum_{l} a_{lk} = 1, l \in N_{k,i-1} \forall k \quad (16)$$

The consensus cooperation strategy imposes a mathematical constraint so that all connected agents converge to the same parameter vector $\boldsymbol{\omega}_{k,i}$:

$$\boldsymbol{\omega}_k = \boldsymbol{\omega}_m, k = 1,2,\ldots,K, m \in N_k, \quad (17)$$

In this work the strategy adopted for the a_{lk} combiner is the Metropolis rule [1], given by

$$\begin{cases} a_{lk} = \dfrac{1}{\max(n_k, n_l)}, k \neq l \\ a_{lk} = 1 - \sum_{l \in N_k / k} a_{lk}, k = l \\ a_{kk} = 0 \end{cases}$$

The proposed distributed Consensus CG algorithm based on the derivation steps obtains the updated weight substituting (15) in (8), giving as result:

$$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\varphi}_{k,i}(j) + \alpha_{k,i}(j) \mathbf{p}_{k,i}(j). \quad (18)$$

The pseudo-code of the proposed distributed consensus CG algorithm is presented in TABLE I

TABLE I. PSEUDO CODE OF THE CONSENSUS CG ALGORITHM

Parameter initialization
$\boldsymbol{\omega}_{k,0} = 0$; $\mathbf{R}(0) = \mathbf{I} * \delta$; …
for each time instant i≥0
for each agent k=1,2,…,N
$\boldsymbol{\varphi}_{l,i-1} = \sum_{l \in N_k} a_{lk} \boldsymbol{\omega}_{l,i-1}$
for each CG iteration j =1 until convergence
$\alpha(j) = \dfrac{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)}{\mathbf{p}_{k,i}^H(j) \mathbf{R}_{k,i}(j) \mathbf{p}_{k,i}(j)}$
$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\varphi}_{k,i}(j) + \alpha_{k,i}(j) \mathbf{p}_{k,i}(j)$
$\mathbf{g}_{k,i}(j) = \mathbf{g}_{k,i}(j-1) - \alpha_{k,i}(j) \mathbf{R}_{k,i} \mathbf{p}_{k,i}(j-1)$
$\beta(j) = \dfrac{\mathbf{g}_{k,i}(j)^H \mathbf{g}_{k,i}(j)}{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)}$
$\mathbf{p}_{k,i}(j+1) = \mathbf{g}_{k,i}(i) + \beta(j) \mathbf{p}_{k,i}(j)$
end for
$\boldsymbol{\omega}_{k,i} = \boldsymbol{\omega}_{k,i}(j_{last})$
end for
end for

C. Computational Complexity

The computational complexity of the proposed distributed Consensus CG algorithm has a cost of order $O(M^2)$ that depends on the connectivity of the network as well as the number of iterations J.

IV. PROPOSED SPARSITY-AWARE DISTRIBUTED CONSENSUS CG

Based on the previous development of a distributed CG algorithm, the following description presents the general strategy of distributed sparsity-aware consensus CG using l_1 (ZA) and log-sum (RZA) norm penalty functions.

A. ZA CG algorithm

The cost function in this case is given by

$$C_{CG}(\boldsymbol{\omega}_{k,i}) = \frac{1}{2} \sum_{k=1}^{N} \boldsymbol{\omega}_{k,i}^H \mathbf{R}_{k,i} \boldsymbol{\omega}_{k,i} - \sum_{k=1}^{N} \mathbf{b}_{k,i}^H \boldsymbol{\omega}_{k,i} + f_1 \quad (19)$$

where $f1$ denotes the l_1 penalty function (ZA) and is defined by

$$f_1 = \rho \left\| \boldsymbol{\omega}_{k,i}(i) \right\|_1 \quad (20)$$

Applying the partial derivation of the penalty function gives

$$\frac{\partial(f_1)}{\partial(\boldsymbol{\omega}_{k,i}{}^*)} = \operatorname{sgn}(\boldsymbol{\omega}_{k,i}) = \begin{cases} \dfrac{\boldsymbol{\omega}_{k,i}}{|\boldsymbol{\omega}_{k,i}|}, \boldsymbol{\omega}_{k,i} \neq 0 \\ 0, \boldsymbol{\omega}_{k,i} = 0 \end{cases} \quad (21)$$

The pseudo-code with the solution of the algorithm for this case is presented in TABLE II

TABLE II. PSEUDO CODE OF THE SPARSITY-AWARE CONSENSUS CG ALGORITHM

Parameter initialization
$\boldsymbol{\omega}_{k,0} = 0$; $\mathbf{R}(0) = \mathbf{I} * \delta$; …
for each time instant i≥0
for each agent k=1,2,…,N
$\boldsymbol{\varphi}_{l,i-1} = \sum_{l \in N_k} a_{lk} \boldsymbol{\omega}_{l,i-1}$
for each CG iteration j from 1 until convergence
$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\varphi}_{k,i}(j) + \alpha_{k,i}(j) \mathbf{p}_{k,i}(j)$

$$\alpha(j) = \frac{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)}{\mathbf{p}_{k,i}{}^H(j)\mathbf{R}_{k,i}(j)\mathbf{p}_{k,i}(j)}$$

$$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\varphi}_{k,i}(j) + \alpha_{k,i}(j)\mathbf{p}_{k,i}(j)$$

$$\mathbf{g}_{k,i}(j) = \mathbf{g}_{k,i}(j-1) - \alpha_{k,i}(j)\mathbf{R}_{k,i}\mathbf{p}_{k,i}(j-1)$$

$$\beta(j) = \frac{\mathbf{g}_{k,i}(j)^H \mathbf{g}_{k,i}(j)}{\mathbf{g}_{k,i}(j-1)^H \mathbf{g}_{k,i}(j-1)}$$

$$\mathbf{p}_{k,i}(j+1) = \mathbf{g}_{k,i}(i) + \beta(j)\mathbf{p}_{k,i}(j)$$

end for

$$\boldsymbol{\omega}_{k,i} = \boldsymbol{\omega}_{k,i}(j_{last}) - \rho\,\mathrm{sgn}(\omega_{k,i} - 1)$$

end for

end for

B. RZA CG algorithm

When the logarithmic penalty function f_2 is used in the cost function, we have

$$C_{CG}(\boldsymbol{\omega}_{k,i}) = \frac{1}{2}\sum_{k=1}^{N}\boldsymbol{\omega}_{k,i}{}^H\mathbf{R}_{k,i}\boldsymbol{\omega}_{k,i} - \mathbf{b}_{k,i}{}^H\boldsymbol{\omega}_{k,i} + \rho\sum_{i=1}^{M}\log(1+\frac{|\boldsymbol{\omega}_{k,i}|}{\varepsilon}) \quad (22)$$

The partial derivative of the penalty function applied with respect to $\boldsymbol{\omega}$ is shown below.

$$f_2 = \rho\sum_{i=1}^{M}\log(1+|\boldsymbol{\omega}_{k,i}|/\varepsilon) \quad (23)$$

$$\frac{\partial(f_2)}{\partial(\boldsymbol{\omega}_{k,i}{}^*)} = \frac{\mathrm{sgn}(\boldsymbol{\omega}_{k,i})}{1+\varepsilon\|\boldsymbol{\omega}_{k,i}\|_1} \quad (24)$$

The recursions for the RZA consensus CG are similar to the ZA consensus CG. The main difference lies in the weight update recursion described by

$$\boldsymbol{\omega}_{k,i}(j) = \boldsymbol{\varphi}_{k,i}(j) + \alpha_{k,i}(j)\mathbf{p}_{k,i}(j) - \rho\frac{\mathrm{sgn}(\boldsymbol{\omega}_{k,i})}{1+\varepsilon\|\boldsymbol{\omega}_{k,i}\|_1} \quad (25)$$

In both cases these sparsity-aware algorithms attract to zero the values of the parameter vector which are very small or are not useful. This results in an algorithm with a faster convergence and lower MSD values as can be seen in following sections.

C. Computational Complexity

Similarly to the standard version of the proposed distributed consensus CG the sparsity-aware version has a quadratic computational cost, which depends on the number of nodes connected and the CG iterations. TABLE III shows the operations in terms of additions and multiplications.

TABLE III. COMPUTATIONAL COMPLEXITY FOR CONSENSUS CG METHOD

Method	Additions	Multiplications
Consensus_CG	$3LM + LJ(M^2+4M-2)$	$L(M^2+2M)+LJ(3M^2+2M)$
ZA-Consensus-CG	$3LM + LJ(M^2+4M-2)$	$L(M^2+2M)+LJ(3M^2+3M)$
RZA-Consensus_CG	$3LM + LJ(M^2+3M-1)$	$L(M^2+2M)+LJ(3M^2+3M)$

V. SIMULATION RESULTS

In this work we evaluated the proposed standard distributed consensus CG algorithm as well as the sparsity-aware versions. The results are compared with the LMS [6] and RLS [8] algorithms based on the mean square deviation MSD of the network. We consider a network with 20 nodes and 1000 iterations per run. Each iteration corresponds to a time instant. The results are averaged over 100 experiments. The length of the filter is 20 and the variance of the input signal is equal to 1, which has been modeled as a complex Gaussian noise with a variance of 0.001.

A. Performance comparison between proposed standard and sparsity-aware distributed consensus CG algorithms.

The parameters of the simulations for each algorithm and the network were set to ensure an optimized performance. In the case of the sparsity-aware algorithm the system parameter vector was set with two values equal to one and the remaining parameters were set to zero. After all the iterations, the performance of each algorithm in terms of MSD is shown in Fig. 2. The results show, that the RZA-Consensus-CG outperforms both ZA Consensus-CG and standard Consensus-CG methods in terms of convergence speed and MSD values at steady state.

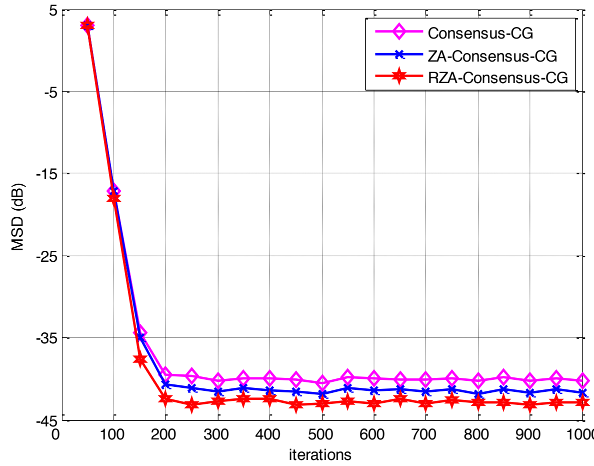

Fig. 2 MSD of the network against the iteration's number for distributed consensus standard and sparse-aware CG versions with λ=0.99, ρ_{ZA}=0.5*10^{-4}, ρ_{RZA}=1*10^{-3}, ε=0.1, δ=10^{-3}, S=2/20. Number of CG iterations J =5.

Different sparsity levels S were considered for the proposed algorithms. Fig.3 shows the MSD behavior of the RZA version for different levels of sparsity s.

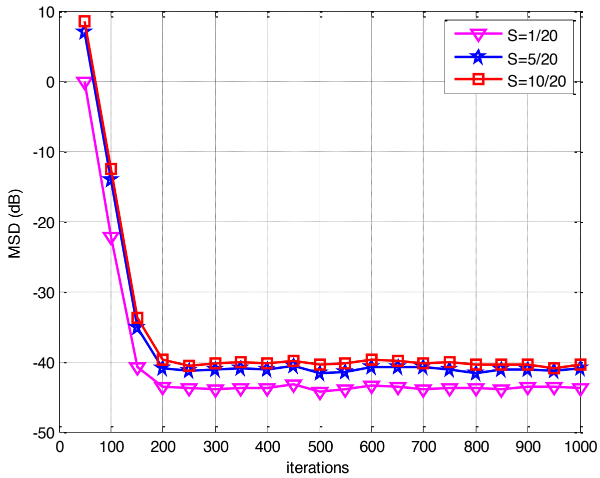

Fig. 3 Output MSD against the iteration's number for RZA distributed consensus CG with λ=0.95, ρ=0.5*10^{-3}, ε=0.1, δ=10^{-3}, S=1/20, 5/20, 10/20. Number of CG iterations J = 5.

It can be noticed in Fig.3 that if the number of nonzero values is increased the algorithm will take longer to converge and the deviation will be larger as compared to a sparse system with a lower number of nonzero values.

B. Comparison between LMS, RLS and the proposed distributed consensus CG algorithms.

The proposed algorithms were also compared with the distributed versions of the well-known LMS and RLS algorithms. Fig. 4 bellow shows the MSD of a network where the RZA version of the consensus LMS [6], RLS [8] and CG version were tested.

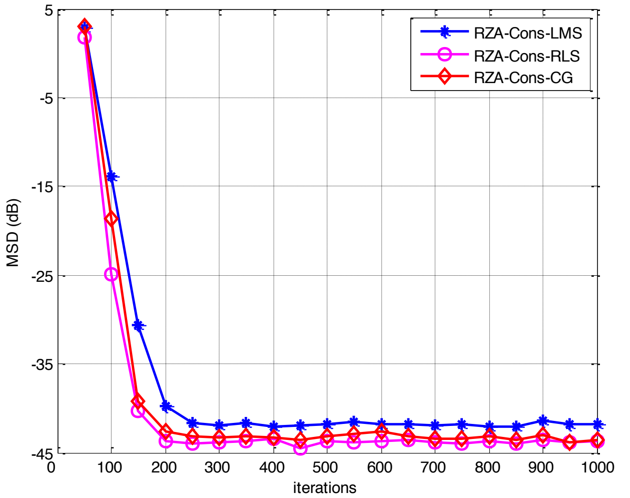

Fig. 4 MSD of the network against the iteration's number for RZA distributed consensus standard and sparse-aware CG versions with λ_{RLS}=0.99, λ_{CG}=0.99 $\rho_{RZA\_RLS}=10^{-3}$, $\rho_{RZA\_RLS}=5*10^{-4}$, $\rho_{RZA\_CG}=10^{-3}$, ε=0.1, $\delta=10^{-3}$, *S*=2/20. Number of CG iterations J = 5

C. Comparison between consensus CG algorithms and diffusion CG.

In Fig.5 it is presented the MSD simulation of the CTA-CG [12] and the consensus CG algorithms.

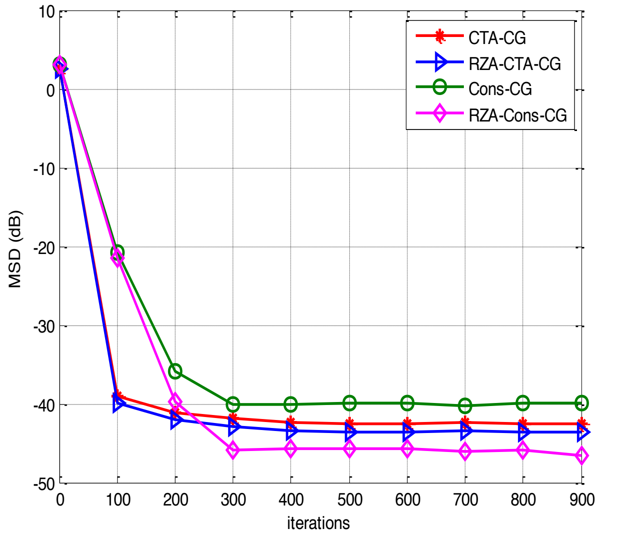

Fig. 5 MSD of the network against the iteration's number for distributed diffusion CTA and consensus CG with λ_{CTA}=0.99, $\lambda cons$=0.99, $\delta=10^{-2}$, $\rho_{RZA}=0{,}2*10^{-3}$, *S*=4/20, Number of CG iterations J = 5

It can be observed that the diffusion CG algorithm has a faster convergence as compared to the consensus CG, but in contrast the consensus protocol with sparsity reaches a lower MSD value at steady state.

VI. CONCLUSIONS

In this work we have proposed distributed consensus CG algorithms for parameter estimation over sensor networks. The proposed distributed CG algorithm using a consensus protocol has a faster convergence than the LMS and a very similar performance to the RLS. Simulation results have shown that the developed consensus CG and sparsity-aware consensus CG algorithms are suitable techniques for adaptive parameter estimation problems and can be employed in other applications. Due to the conditions of sparse parameter vectors, we will consider for future work the development of a strategy that allows one to transmit compressed data.

REFERENCES

[1] S. Yuan T, and A. H. Sayed, "Diffusion Strategies Outperform Consensus Strategies for Distributed Estimation over Adaptive Networks", IEEE Transactions on signal processing, vol. 60, no. 12, December 2012.

[2] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," IEEE Transactions Signal Process, vol. 48, no. 8, pp.223–229, Aug 2007.

[3] A. H. Sayed, "Adaptation, Learning, and Optimization over Networks". Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311–801, 2014.

[4] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, Mar. 1974.

[5] P. D. Lorenzo and A. H. Sayed, "Sparse Distributed Learning Based on Diffusion Adaptation", IEEE Transactions on signal processing, vol. 61, no. 6, March 15, 2013.

[6] Y. Liu, C. L. and Z. Zhang, "Diffusion Sparse Least-Mean Squares Over Networks", IEEE Transactions on Signal Processing, vol. 60, no. 8, August 2012

[7] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for System Identification", IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, April 2009.

[8] P. D. Lorenzo and S. Barbarossa, "Distributed Least Mean Squares Strategies for Sparsity-Aware Estimation over Gaussian Markov Random Fields", IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), May 2014.

[9] E. M. Eksioglu, "RLS Adaptive Filtering with Sparsity Regularization", 10th International Conference on Information Science, Signal Processing and their Applications ISSPA, may 2010.

[10] R. C. de Lamare and R. Sampaio-Neto, Sparsity-Aware Adaptive Algorithms Based on Alternating Optimization with shrinkage, IEEE Signal Processing Letters, vol. 21, no. 2, February 2014.

[11] P. S. Chang and A. N. Willson, Jr., "Analysis of Conjugate Gradient Algorithms for Adaptive Filtering", IEEE Transactions on Signal Processing, vol. 48, no. 2, February 2000

[12] S. Xu, and R.C de Lamare, , "Distributed conjugate gradient strategies for distributed estimation over sensor networks" Sensor Signal Processing for Defense SSPD, September 2012

[13] O. Axelsson, *Iterative Solution Methods*. New York: Cambridge Univ. Press, 1994

[14] G. H. Golub and C. F. Van Loan, Matrix Computations, 2nd Ed. Baltimore, MD: Johns Hopkins Univ. Press, 1989

[15] S. Xu, R. C. de Lamare and H. V. Poor, "Distributed Compressed Estimation Based on Compressive Sensing", IEEE Signal Processing letters, vol. 22, no. 9, September 2014.

[16] S. Wang, H. M. and B. Xi, D. Sun, Conjugate Gradient-based Parameters Identification, 8th IEEE International Conference on Control and Automation, China, June 2010.

[17] S. Theodoridis, Machine Learning a Bayesian and Optimization Perspective, Academic Press, March 2015.