A GREEDY SPARSITY–PROMOTING LMS FOR DISTRIBUTED ADAPTIVE LEARNING IN DIFFUSION NETWORKS

Symeon Chouvardas, Gerasimos Mileounis, Nicholas Kalouptsidis and Sergios Theodoridis

Dept. of Informatics and Telecommunications, University of Athens, Ilisia 15784, Greece

ABSTRACT

In this paper, a distributed adaptive algorithm for sparsity–aware learning in diffusion networks is developed. The algorithm follows the greedy roadmap for sparsity along with the adapt–combine cooperation strategy, based on the LMS rationale for adaptivity. A bound on the error norm between the obtained estimates and the target vector is computed, and the algorithm is shown to converge in the mean under some general assumptions. Finally, comparative experiments with a recently developed sparsity–promoting diffusion LMS demonstrate the enhanced performance of the proposed algorithm.

Index Terms— Adaptive distributed learning, Sparsity–aware learning, Greedy techniques.

1. INTRODUCTION

This paper considers the task of sparsity–aware learning in the context of a distributed environment in an online setting. The proposed novel algorithm bears its progeny, from its sparsity–promoting potential, to the family of greedy algorithms. Its adaptive nature is an offspring of the celebrated LMS family and its distributed learning potential complies with the so–called diffusion topology.

The diffusion topology is the main alternative to the *incremental topology* for decentralized distributed processing. In the incremental topology, each node communicates with only one node and the network has a cyclic topology, *e.g.*, [1]. In the diffusion topology each node communicates with a number of neighbouring nodes, *e.g.*, [2, 3]. The diffusion topology is easier to be adopted in real-time applications and it is more robust to node failures.

In this paper, we consider a diffusion network and the goal is to estimate, adaptively, an unknown target parameter vector, which is common to all nodes and it is also known to be sparse. Diffusion adaptive algorithms have been proposed in the context of the diffusion Least Mean Squares (LMS), [2], the diffusion Recursive Least Squares (RLS), [4], and the projection based Adaptive Projected Subgradient Method [5, 6]. The task of developing sparsity–promoting algorithms for distributed learning has only very recently been addressed, in [7] for the batch learning scenario and in [3, 8] for the online time adaptive case.

The majority of the sparsity promoting adaptive algorithms exploit sparsity by embedding into the optimization problem an ℓ_1–norm constraint, in its regularized form (see for example [9–12]) or via projections on the related ℓ_1–ball [13] or via generalized thresholding (non–convex) operators [14]. It has been established that, by exploiting the underlying sparsity of the target vector, leads to significantly improved convergence speeds compared to the unconstrained algorithms, see, *e.g.*, [15], for a related review. Another philosophy, which adaptively exploits sparsity of the unknown vector, employs greedy techniques, *e.g.*, [16]. In a nutshell, greedy techniques estimate the positions containing the non–zero coefficients of the unknown target vector to be estimated, and then perform the computations in this subset.

A new distributed greedy algorithm is developed in this paper. More specifically, a signal, known as proxy, is computed recursively at each node, based on the statistics of the measurements. In the sequel, an LMS iteration is performed, restricted on a properly selected subset, which is chosen to comprise the s (user–defined) dominant indices of the proxy. To the authors' knowledge, this is the first time that a greedy type sparsity–promoting algorithm is developed for the adaptive distributed learning scenario.

The theoretical properties of the algorithm are discussed and experimental results reveal that the proposed scheme outperforms the existing distributed adaptive LMS–based scheme.

Notation: Vectors will be denoted by boldface letters and matrices by uppercase boldface letters. Moreover, the symbols $(\cdot)^T$ and $\mathbb{E}[\cdot]$ will stand for the transpose and the expectation operators respectively. The set of all real numbers and the set of non–negative integers will be denoted by $\mathbb{R}$ and $\mathbb{Z}$ respectively. In addition, the term $[\boldsymbol{A}]_{ij}$ stands for the ij–th element of matrix $\boldsymbol{A}$. Finally, given a vector $\boldsymbol{w}$, the term $\mathrm{supp}_{\mathrm{s}}(\boldsymbol{w})$, called support set, stands for the subset containing the s non–zero coefficients of $\boldsymbol{w}$. The cardinality of a set $\mathcal{S}$ is denoted by $|\mathcal{S}|$.

2. PROBLEM STATEMENT

Our goal is to estimate a sparse unknown vector $\boldsymbol{w}_* \in \mathbb{R}^m$ exploiting measurements collected at the N nodes of a network in accordance to the diffusion topology, *e.g.*, [17]. We denote the node set by $\mathcal{N} = \{1, \ldots, N\}$, and we assume that each node is able to exchange information, with a subset of $\mathcal{N}$, $\mathcal{N}_k, k = 1, \ldots, N$, commonly referred to as the *neighbourhood* of k. Each node k, at each time instance, has access to the output measurement, $d_k(n) \in \mathbb{R}$, and the input vector, $\boldsymbol{x}_k(n) \in \mathbb{R}^m$, which are related via the linear model:

$$d_k(n) = \boldsymbol{x}_k^T(n)\boldsymbol{w}_* + v_k(n),\ \forall k \in \mathcal{N},\ \forall n \in \mathbb{Z}, \qquad (1)$$

where the term $v_k(n)$ stands for the additive noise process at each node. The unknown target vector is assumed to be s–sparse, *i.e.*, $\|\boldsymbol{w}_*\|_{\ell_0} \leq s \ll m$, where with $\|\cdot\|_{\ell_0} = |\mathrm{supp}_s(\cdot)|$ we denote the ℓ_0 "norm". In sparsity–aware learning, it is usually assumed that we have a prior knowledge of the approximate sparsity level (denoted by s), *i.e.*, the number of non–zero coefficients that actually contribute to the output, while the set of indices corresponding

Email: {schouv,gmil,kalou,stheodor}@di.uoa.gr.

This research has been co-financed by the European Union (European Social Fund ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: THALIS–UOA–SECURE WIRELESS NONLINEAR COMMUNICATIONS AT THE PHYSICAL LAYER.

Table 1. The Greedy Diffusion LMS (GreeDi–LMS) Algorithm

Algorithm description			Complexity
$\boldsymbol{w}_k(0) = 0, \boldsymbol{p}_k(0) = 0, \boldsymbol{R}_k(0) = 0.$		{Initialization}	
$0 < \lambda \leq 1$		{Forgetting factor}	
$k \in \mathcal{N}$		{Node Index}	
For $n := 1, 2, \ldots$ **do**			
1:	$\boldsymbol{p}_k(n) = \sum_{j=1}^{n} \lambda^{n-j} \frac{\boldsymbol{x}_k(j) d_k(j)}{n} = \frac{n}{n+1} \lambda \boldsymbol{p}_k(n-1) + \frac{\boldsymbol{x}_k(n) d_k(n)}{n+1}$	{Form cross-correlation}	$O(m)$
2:	$\boldsymbol{R}_k(n) = \sum_{j=1}^{n} \lambda^{n-j} \frac{\boldsymbol{x}_k(j) \boldsymbol{x}_k^T(j)}{n} = \frac{n}{n+1} \lambda \boldsymbol{R}_k(n-1) + \frac{\boldsymbol{x}_k(n) \boldsymbol{x}_k^T(n)}{n+1}$	{Form autocorrelation}	$O(m^2)$
3:	$\overline{\boldsymbol{p}}_k(n) = \sum\limits_{l \in \mathcal{N}_k} b_{l,k} \boldsymbol{p}_l(n)$	{Combine cross–correlation}	$O(\lvert\mathcal{N}_k\rvert m)$
4:	$\overline{\boldsymbol{R}}_k(n) = \sum\limits_{l \in \mathcal{N}_k} b_{l,k} \boldsymbol{R}_l(n)$	{Combine autocorrelation}	$O(\lvert\mathcal{N}_k\rvert m^2)$
5:	$\widehat{\mathcal{S}}_{k,n} = \mathrm{supp}_s \big(\boldsymbol{w}_k(n-1) + \tilde{\mu}_k(\overline{\boldsymbol{p}}_k(n) - \overline{\boldsymbol{R}}_k(n) \boldsymbol{w}_k(n-1))\big)$	{Identify large components}	$O(m)$
6:	$\boldsymbol{\psi}_k(n) = \boldsymbol{w}_{k\mid\widehat{\mathcal{S}}_{k,n}}(n-1) + \mu_k \boldsymbol{x}_{k\mid\widehat{\mathcal{S}}_{k,n}}(n)[d_k(n) - \boldsymbol{x}^T_{k\mid\widehat{\mathcal{S}}_{k,n}}(n) \boldsymbol{w}_{k\mid\widehat{\mathcal{S}}_{k,n}}(n-1)]$	{LMS iteration}	$O(s)$
7:	$\widetilde{\boldsymbol{w}}_k(n) = \sum\limits_{l \in \mathcal{N}_k} c_{l,k} \boldsymbol{\psi}_{l\mid\widehat{\mathcal{S}}_{l,n}}(n)$	{Combine local estimates}	$O(\lvert\mathcal{N}_k\rvert s)$
8:	$\widetilde{\mathcal{S}}_{k,n} = \max(\lvert\widetilde{\boldsymbol{w}}_k(n)\rvert, s)$	{Obtain the pruned support}	$O(m)$
9:	$\boldsymbol{w}_{k\mid\widetilde{\mathcal{S}}_{k,n}}(n) = \widetilde{\boldsymbol{w}}_{k\mid\widetilde{\mathcal{S}}_{k,n}}(n), \boldsymbol{w}_{k\mid\widetilde{\mathcal{S}}^c_{k,n}}(n) = \mathbf{0}$	{Prune the combined estimates}	$O(\lvert\mathcal{N}_k\rvert s)$
end For			

to non–zero elements remains unknown. Hence, when seeking for an s–sparse vector, our effort is twofold; first, one has to estimate where the s non–zero coefficients lie, and then obtain estimates of their corresponding values. Classical non–distributed adaptive algorithms estimate the unknown parameters based on the input/output information. In adaptive distributed learning, at each time instance the nodes seek for a common unknown vector and cooperate with each other, instead of acting as individual learners. More specifically, apart from the locally sensed measurements, information received by the neighbourhood is also exploited in order to produce estimates. This information comprises the currently available estimates of the unknown vector, as well as measurements which are sensed by the neighbouring nodes [2]. It is by now well established that the cooperation among the nodes leads to an enhanced performance, compared to the case where each node operates individually [2]. Usually, the cooperation among the nodes implies that each node receives estimates from the neighbourhood, and then takes a convex combination of them. Several modes of cooperation have been studied in the literature including the:

1. **Adapt Combine [2]:** In this strategy, each node computes an intermediate estimate by exploiting locally sensed measurements. After this step, each node receives these estimates from the neighbouring nodes and combines them, in order to produce the final estimate.

2. **Combine Adapt [6, 17]:** Here, the combination step precedes the adaptation one.

3. **Consensus Based [18]:** In this category, the computations are made in parallel and there is no clear distinction between the combine and the adapt steps.

In this paper, the Adapt Combine strategy is followed, since it has been verified that it converges faster to a lower steady state error floor compared to the other methodologies [19].

3. THE DISTRIBUTED GREEDY SPARSITY–PROMOTING LMS

Under a centralized operating mode, greedy algorithms seek to find an s–sparse approximation to $\boldsymbol{w}_*$ by iteratively applying the following two steps:

> **Greedy or subset selection:** Compute the support of the sparse vector with the aid of a signal which is referred to as proxy signal;
>
> **Greedy update:** Perform the estimation/adaptation step restricted to that set.

The advantages of such an approach lie in the speed of execution and high–performance. Application of the greedy principle to decentralized operation needs to address two issues: the combined filter updating and the support set consensus, which implies that the nodes exchange information in order to "agree" to the same support set.

The majority of greedy algorithms makes decisions for the support set based on a signal proxy of the form.

$$\boldsymbol{X}_k^T(n) \left(\boldsymbol{d}_k(n) - \boldsymbol{X}_k(n) \widehat{\boldsymbol{w}}_k\right) \approx \boldsymbol{X}_k^T(n) \boldsymbol{X}_k(n) \left(\boldsymbol{w}_* - \widehat{\boldsymbol{w}}_k\right),$$

where $\boldsymbol{X}_k(n) = [\boldsymbol{x}_k(1), \boldsymbol{x}_k(2), \cdots, \boldsymbol{x}_k(n)]^T$ and $\boldsymbol{d}_k(n) = [d_k(1), d_k(2), \cdots, d_k(n)]^T$. It is important to note that for such a proxy, the algorithm at every iteration, selects distinct indices. In other words, the error residual is orthogonal to the indices that have been previously chosen. An enhanced proxy, which allows multiple selection of indices, has the following form,

$$\mathrm{supp}_{s \text{ or } 2s} \left(\left|\boldsymbol{X}_k^T(n) \left(\boldsymbol{d}_k(n) - \boldsymbol{X}_k(n) \widehat{\boldsymbol{w}}_k\right)\right|\right) \cup \mathrm{supp}_s \left(\left|\widehat{\boldsymbol{w}}_k\right|\right),$$

and in such cases, the estimation step is restricted to a set of indices larger than s. However, both of these signals are not adequate for distributed learning, since they carry time–varying information from local estimates and hence pose problems to the support set consensus. In other words, distributed learning aims at finding a proxy, which is close to $\boldsymbol{w}_*$ and not to $(\boldsymbol{w}_* - \widehat{\boldsymbol{w}}_k)$.

Towards this end, we distinguished two different strategies to select the signal proxy, which serve the needs of adaptive distributed learning. The simplest one is based on a cross–correlation vector $\boldsymbol{p}_k(n) = \mathbb{E}[\boldsymbol{x}_k(n) d_k(n)]$, where each node $k \in \mathcal{N}$ enhances its estimate by cooperating with its neighbourhood $\mathcal{N}_k$ via the introduction of a diffusion step [2] (*e.g.*, see steps 1 & 2 of Table 1). The second one, which will be followed in this work, is shown experimentally to achieve faster support set recovery, since it takes into account the current local estimate, and it is given by [20]

$$\boldsymbol{w}_k(n-1) + \tilde{\mu}_k(\overline{\boldsymbol{p}}_k(n) - \overline{\boldsymbol{R}}_k(n)\boldsymbol{w}_k(n-1)) \approx \boldsymbol{w}_*, \qquad (2)$$

where $\boldsymbol{w}_k(n-1)$, $\tilde{\mu}_k$, $\overline{\boldsymbol{p}}_k(n)$ and $\overline{\boldsymbol{R}}_k(n)$ are defined in Table 1. The proposed proxy constitutes a distributed exponentially weighted extension of its centralized form (addressed in [20]).

Let us now briefly discuss the core steps of the proposed Greedy Diffusion LMS (GreeDi–LMS), of Table 1. Steps 1–5 constitute the *distributed greedy subset selection* process. Steps 1–4 create, in a cooperative manner, the basic elements needed for the proxy update and step 5 selects the s–largest components. The *distributed greedy update* is established via the Adapt–Combine LMS (steps 6 & 7) of [2]. The diffusion steps 2, 4 and 7 bring the local estimates closer to the global estimates based on the entire network [2]. Finally, because we are operating in a network of different nodes, some of them will achieve support set convergence faster than others. Therefore, at a node level, we pay more attention to the s–dominant positions via the introduction of a pruning step directly after the adaptation in the combined greedy update, steps 8 & 9.

Finally, the weights, $b_{l,k}$, $c_{l,k}$, used to fuse the information coming from the neighbourhood, can be chosen with respect to several combination rules. Two well known representatives are the Metropolis rule and the Uniform rule [17]. The Metropolis weights are given by

$$c_{k,l} = \begin{cases} \frac{1}{\max\{|\mathcal{N}_k|, |\mathcal{N}_l|\}}, & \text{if } l \in \mathcal{N}_k \text{ and } l \neq k, \\ 1 - \sum_{l \in \mathcal{N}_k \setminus k} c_{k,l}, & \text{if } l = k, \\ 0, & \text{otherwise,} \end{cases}$$

and the uniform ones equal to

$$c_{k,l} = \begin{cases} \frac{1}{|\mathcal{N}_k|}, & \text{if } l \in \mathcal{N}_k, \\ 0, & \text{otherwise.} \end{cases}$$

Remark 1. The computational complexity of GreeDi–LMS is linear except from the update of $\boldsymbol{R}_k(n)$, which requires $\mathcal{O}(m^2)$ operations per iteration. However, exploitation of the shift structure in the regressor vector allow us to drop the computational cost to $\mathcal{O}(m)$, *e.g.,* [21].

Remark 2. Another way to reduce complexity is by updating the gradient part of the proxy recursively, as it was done in [16] for the centralized case. However, such an update, although it is computationally lighter, it results in slower support convergence, since the resulting exponentially weighted average contains all $\boldsymbol{w}_k(t), t := 1, \ldots, n-1$ and not only the current estimate.

4. CONVERGENCE ANALYSIS

This section addresses the performance of the proposed algorithm in terms of steady–state behaviour. As it is well known, the LMS converges in the mean to the true parameter vector. This property is also satisfied in diffusion LMS schemes, *e.g.,* [2]. Nevertheless, the diffusion–based sparsity–promoting LMS [8] is not unbiased, due to the convex regularization term embedded in the cost function. In contrast GreeDi–LMS enjoys unbiasedness as demonstrated below.

We introduce the following assumptions:

(A1.) The input and output signals satisfy (1) and are ergodic processes, *i.e.,* $\lim_{n\to\infty} \frac{1}{n} d_k(n)\boldsymbol{x}_k(n) = \boldsymbol{p}_k := \mathbb{E}[d_k(n)\boldsymbol{x}_k(n)]$ and $\lim_{n\to\infty} \frac{1}{n}\boldsymbol{x}_k(n)\boldsymbol{x}_k^T(n) = \boldsymbol{R}_k := \mathbb{E}[\boldsymbol{x}_k(n)\boldsymbol{x}_k^T(n)]$.

(A2.) The input is a white noise process of variance σ_k^2, *i.e.,* $\boldsymbol{R}_k = \sigma_k^2 \boldsymbol{I}_m$.

Theorem 1. *Let $\tilde{\mu}_k(n) = \tilde{\nu}_k^{-1}(n)$, where*

$$\tilde{\nu}_k(n) = \sum_{l \in \mathcal{N}_k} b_{l,k} \frac{1}{m} \sum_{i=1}^{m} [\boldsymbol{R}_l(n)]_{ii}.$$

Then, $\exists n_0 \in \mathbb{Z}$ such that $\widehat{\mathcal{S}}_{k,n} = \mathcal{S}$, $\forall n \geq n_0$, $\forall k \in \mathcal{N}$.

Sketch of proof *. The ergodicity assumptions, in conjunction with standard bounded property of the LMS estimate $\boldsymbol{\psi}_k(n)$ imply that the various signal proxies converge to the average of the exponentially weighted cross–correlation term, *i.e.*, $\overline{\sigma}_k^{-2}\overline{\boldsymbol{p}}_k$, where $\overline{\sigma}_k^2 := \sum_{l \in \mathcal{N}_k} b_{l,k}\sigma_l^2$ and $\overline{\boldsymbol{p}}_k := \sum_{l \in \mathcal{N}_k} b_{l,k}\boldsymbol{p}_l$. Since $\boldsymbol{R}_k$ are diagonal and thus $\mathrm{supp}_s\left(\overline{\sigma}_k^{-2}\overline{\boldsymbol{p}}_k\right) = \mathcal{S}$, which occurs form the Wiener–Hopf equation, *e.g.,* [21], which states that $\overline{\sigma}_k^{-2}\overline{\boldsymbol{p}}_k = \overline{\sigma}_k^{-2}\sum_{l \in \mathcal{N}_k} b_{l,k}\boldsymbol{R}_l\boldsymbol{w}_*$. Therefore, the s dominant elements of the proxy eventually coincide with those of $\overline{\boldsymbol{p}}_k$.

The main conclusion drawn from Thm. 1 is that the true support turns out to be computed in a finite number of steps. This is a direct consequence of the fact that the proxy converges asymptotically to a vector, which has the same support as the unknown one. The physical interpretation of the above statement is that the distance between these two vectors will be sufficiently small, so that the s largest coefficients of the proxy will be in those positions as indicated by the true support. This has two direct implications, which can be explored further in order to reduce computational resources. First, once the signal proxy has converged then the algorithmic scheme of Table 1 drops to the Adapt–Combine LMS presented in [2] and hence the proof of convergence in the mean follows in a similar fashion. Second, if the signal proxy remains constant for a number of consequent iterations we may then stop performing the proxy selection process of the algorithm.

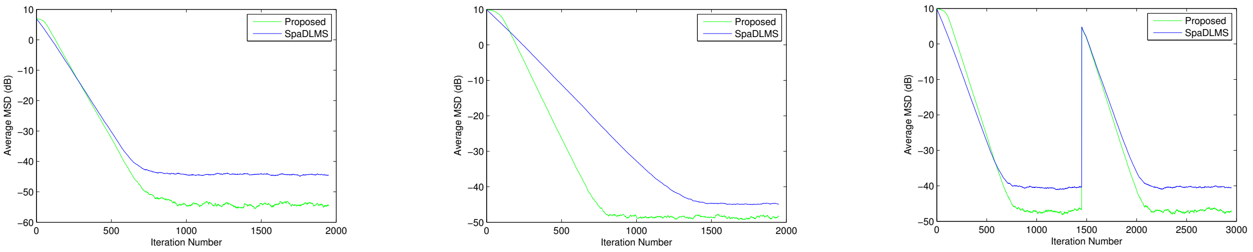
In addition, it follows from Thm. 1 that the proposed algorithm avoids the major obstacle of (non–weighted) ℓ_1–minimization methods, which cannot guarantee to recover the correct support and *at the same time* estimate the nonzero entries of $\boldsymbol{w}_*$, consistently. In sparse ℓ_1 regularized LMS–like algorithms, this is reflected by the introduction of a bias term in the mean converged vector.

The following result is also established in [22]. Given a distributed sparse adaptive filter, like the one described in Table 1, we are interested in studying the ℓ_2–norm of the error vector $\boldsymbol{w}_k(n) - \boldsymbol{w}_*$ at each node in the steady–state.

Theorem 2. *Each node $k \in \mathcal{N}$ produces an s–sparse approximation $\boldsymbol{w}_k(n)$ and the following asymptotic error bound for the whole network occurs:*

$$\underline{\epsilon}(n) \leq 2N \max_{k \in \mathcal{N}} \left\{3(1-\mu_k\lambda_k) + \sqrt{2}\delta_{3s,k}(\lambda, n)\,(1+\mu_k\lambda_k)\right\}$$
$$\sum_{k=1}^{N} \epsilon_k^{(2)}(n-1) + 2N \max_{k \in \mathcal{N}} \left\{\mu_k \|\boldsymbol{x}_{k|\widehat{\mathcal{S}}_{k,n}}^T(n)\|_{\ell_2}\right\} \sum_{k=1}^{N} |e_{o,k}(n)| \qquad (3)$$

*The full proof is omitted due to space limitations.



(a) 10 nodes, $m = 100$ & SR 10%

(b) 10 nodes, $m = 100$ & SR 5%

(c) Tracking performance

Fig. 1. Average MSD Performance of the proposed algorithm at various scenarios

where $\underline{\epsilon}(n) := \|\underline{\boldsymbol{w}}(n) - \underline{\boldsymbol{w}}_*\|_{\ell_2}$*,* $\underline{\boldsymbol{w}}(n) = [\boldsymbol{w}_1^T(n), \ldots, \boldsymbol{w}_N^T(n)]^T \in \mathbb{R}^{Nm}$*,* $\underline{\boldsymbol{w}}_* = [\boldsymbol{w}_*^T, \ldots, \boldsymbol{w}_*^T]^T \in \mathbb{R}^{Nm}$ *and* $\epsilon_k^{(2)}(n) := \|\boldsymbol{w}_k(n) - \boldsymbol{w}_*\|_{\ell_2}$*. Furthermore,* $e_{o,k}(n)$ *is the estimation error of the optimum Wiener filter,* $\delta_{3s,k}(\lambda, n)$ *is the* $3s$*th order exponentially–weighted restricted isometry constant of the measurement matrix [16],* λ_k *is the maximum eigenvalue of the input covariance matrix* $\boldsymbol{R}_k$ *and the step–size* μ_k*.*

Let us now discuss the intuitions gained from Thm. 2. The first and third term on the right hand side of Eq. (3) reminds us the error bound of the Adapt–Combine LMS [2]. The second term is analogous to the error bound of the Hard Thresholding Pursuit algorithm, [20], restricted on the complementary of the estimated support (corresponding to a batch of data of size n), mainly because we are using an exponentially weighted average proxy variant of [20]. In a way, the second term remedies the pay–off of not having found completely the true support set after n iterations.

5. COMPUTER SIMULATIONS

In this section, we present computer simulations, in order to compare the performance of the proposed algorithm, with the sparsity promoting diffusion LMS–based scheme of [8] (SpaDLMS). The performance metric is the average Mean Square Deviation, which equals to $\mathrm{MSD}(n) = 1/N \sum_{k=1}^{N} \|\boldsymbol{w}_k(n) - \boldsymbol{w}_*\|^2$ and the curves occur from 30 Monte Carlo (MC) runs, in order to reduce the realization dependency. At each MC run, a new sparsity pattern is generated and the non–zero elements of the parameter $\boldsymbol{w}_*$ for that run are draws from a multivariate Gaussian distribution $\boldsymbol{w}_{*|\mathcal{S}} \sim \mathcal{N}(0, \boldsymbol{I})$.

In the first experiment, we consider a diffusion network consisted of $N = 10$ nodes and the unknown vector has dimension equal to $m = 100$, with 10 non–zero coefficients (10% sparsity ratio (s/m)). The input is drawn from a Gaussian distribution, with mean value equal to zero and variance equal to 1, whereas the the variance of the noise equals to $\sigma_k^2 = 0.01\eta_k \ \forall k \in \mathcal{N}$, where $\eta_k \in [0.5, 1]$ is randomly chosen with respect to the uniform distribution. The combination coefficients are chosen with respect to the Metropolis rule [17]. In this experiment, it is assumed that both algorithms are optimized in the sense that the regularization parameter used in [8] is chosen according to the optimum rule presented in this study, which needs knowledge of the ℓ_1–norm of the unknown vector, whereas in the proposed algorithm we assume that we know the number of non–zero coefficients. Finally, the step–sizes are chosen so that the algorithms exhibit a similar convergence speed, and the forgetting factor λ equals to 1. From Fig. 1.a, it can be seen that the proposed algorithm outperforms SpaDLMS significantly, since it converges faster to a lower steady state error floor, at a similar convergence speed.

In the second experiment, the parameters remain the same as in the previous one, but we alter the sparsity level of the unknown vector. More specifically, we assume that the sparsity ratio drops to 5%. Fig. 1.b illustrates that the enhanced performance of the proposed scheme is still retained in this paradigm.

Finally, in the third experiment, we examine the tracking behaviour of the proposed scheme, *i.e.*, the performance of the proposed algorithm in time–varying environments, and the sensitivity in the case where our parameters are not optimized. In order to achieve these two goals, the following scenario is considered. We assume that at the first 1450 iterations the parameters are the same as in the first experiment, with the exception of the forgetting factor which now equals to $\lambda = 0.95$. At the next time instant the channel undergoes a sudden change. Specifically, two non–zero coefficients of the initial unknown vector, are nullified, but the parameters remain the same in both algorithms, hence they are no more "optimal". From Fig. 1.c, it can be readily seen that the proposed algorithm enjoys a good tracking speed, since after the sudden change in the unknown parameter, it reaches at steady state, faster than the SpaDLMS.

6. CONCLUSION

In this paper, a novel sparsity–promoting adaptive algorithm for distributed learning was proposed. The algorithm follows the principles of greedy algorithms. An analytical bound for the error is established and the algorithm is proved to converge in the mean sense. Numerical examples validate the enhanced performance of the developed scheme compared to an other recently proposed algorithm. Future research is focused on batch and blind distributed algorithms suitable for sparsity–aware learning.

7. REFERENCES

[1] C.G. Lopes and A.H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4064–4077, 2007.

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

[3] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity promoting adaptive algorithm for distributed learning," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5412 –5425, 2012.

[4] F.S. Cattivelli, C.G. Lopes, and A.H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive

networks," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1865–1877, 2008.

[5] R.L.G. Cavalcante, I. Yamada, and B. Mulgrew, "An adaptive projected subgradient approach to learning in diffusion networks," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2762–2774, 2009.

[6] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4692–4707, 2011.

[7] G. Mateos, J.A. Bazerque, and G.B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5262–5276, 2010.

[8] P. Di Lorenzo and A.H. Sayed, "Sparse distributed learning based on diffusion adaptation," *arXiv preprint arXiv:1206.3099*, 2012.

[9] B. Babadi, N. Kalouptsidis, and V. Tarokh, "SPARLS: The sparse RLS algorithm," *IEEE Trans. on Sig. Process.*, vol. 58, no. 8, pp. 4013–4025, 2010.

[10] N. Kalouptsidis, G. Mileounis, B. Babadi, and V. Tarokh, "Adaptive algorithms for sparse system identification," *Sig. Process. ELSEVIER*, vol. 91, no. 8, pp. 1910–1919, 2011.

[11] D. Angelosante, J.A. Bazerque, and G.B. Giannakis, "Online adaptive estimation of sparse signals: Where rls meets the ℓ_1 norm," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3436–3447, 2010.

[12] Y. Chen, Y. Gu, and A.O. Hero, "Sparse LMS for system identification," in *IEEE Proc. of ICASSP,*, 2009, pp. 3125–3128.

[13] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted ℓ_1 balls," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 936 –952, 2011.

[14] Y. Kopsinis, K. Slavakis, S. Theodoridis, and S. McLaughlin, "Generalized thresholding sparsity-aware algorithm for low complexity online learning," in *IEEE Proc. of ICASSP*. IEEE, 2012, pp. 3277–3280.

[15] S. Theodorids, Y. Kopsinis, and K. Slavakis, "Sparsity-aware learning and compressed sensing: An overview," *arXiv preprint arXiv:1211.5231*, 2012.

[16] G. Mileounis, B. Babadi, N. Kalouptsidis, and V. Tarokh, "An adaptive greedy algorithm with application to nonlinear communications," *IEEE Trans. Signal Process.*, vol. 58, no. 6, pp. 2998 –3007, 2010.

[17] C.G. Lopes and A.H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, 2008.

[18] I.D. Schizas, G. Mateos, and G.B. Giannakis, "Distributed LMS for consensus-based in-network adaptive processing," *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2365 –2382, 2009.

[19] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, dec. 2012.

[20] S. Foucart, "Hard thresholding pursuit: An algorithm for compressive sensing," *SIAM Journal on Numerical Analysis*, vol. 49, no. 6, pp. 2543–2563, 2011.

[21] A.H. Sayed, *Adaptive Filters*, John Wiley and Sons, 2008.

[22] S. Chouvardas, G. Mileounis, N. Kalouptsidis, and S. Theodoridis, "A greedy sparsity–promoting lms for distributed adaptive learning in diffusion networks," *to be submitted*, 2013.