A Sampling Algorithm for Diffusion Networks

Daniel G. Tiglea, Renato Candido, and Magno T. M. Silva
Escola Politécnica, University of São Paulo, Brazil
{dtiglea, renatocan, magno}@lps.usp.br

***Abstract*—In this paper, we propose a sampling mechanism for adaptive diffusion networks that adaptively changes the amount of sampled nodes based on mean-squared error in the neighborhood of each node. It presents fast convergence during transient and a significant reduction in the number of sampled nodes in steady state. Besides reducing the computational cost, the proposed mechanism can also be used as a censoring technique, thus saving energy by reducing the amount of communication between nodes. We also present a theoretical analysis to obtain lower and upper bounds for the number of network nodes sampled in steady state.**

***Index Terms*—Diffusion strategies, energy efficiency, adaptive networks, distributed estimation, convex combination.**

I. Introduction

Over the last decade, adaptive diffusion networks have attracted widespread attention since they can be used to efficiently estimate certain parameters of interest using information collected at spatially distributed nodes connected through a particular topology [1]–[12]. Many efforts have been devoted to obtain diffusion strategies that are able to learn and adapt from continuous streaming data and exhibit fast convergence, good tracking capability, and low computational cost. When these strategies are implemented on wireless sensor networks, energy consumption is the most critical constraint [10]–[12].

As a result, several selective transmission mechanisms have been proposed to reduce the energy consumption associated with the communication processes. Some of these approaches aim to reduce the amount of information sent in each transmission [13], [14], while others turn links off according to selective communication policies [7]–[10]. Finally, a certain set of solutions seeks to censor the nodes by avoiding the transmission of information to any of their neighbors [11], [12], [15], [16]. This allows the censored nodes to turn their transmitters off, thus saving more energy, and reduces the amount of information used in the processing [12], [15].

Recently, we proposed in [17] a sampling mechanism for the graph diffusion algorithm of [18]. This mechanism changes adaptively the amount of sampled nodes in the graph based on mean-squared error (MSE) in the neighborhood of each node. Thus, the number of sampled nodes decreases when the MSE is low, allowing for fast convergence in the transient and a significant reduction in the computational cost in steady state.

This paper extends our previous work [17] in different ways: (i) the algorithm of [17] is generalized to adaptive diffusion networks in order to reduce their computational cost, (ii) we obtain theoretical lower and upper bounds for the number of network sampled nodes in steady state, and (iii) we show that, with slight modifications, the proposed scheme can also be used as a censoring technique.

The paper is organized as follows. In Sec. II, we revisit the Adapt-Then-Combine diffusion Normalized Least-Mean-Squares (ATC dNLMS) algorithm [1]. In Sec. III, the adaptive sampling algorithm is derived. In Sec. IV, we present a theoretical analysis to predict bounds for the number of sampled nodes in steady state. Simulation results are shown in Sec. V and Sec. VI closes the paper with the conclusions.

Notation. We use normal fonts for scalars and boldface letters for vectors. Moreover, $(\cdot)^{\mathrm{T}}$ denotes transposition, $|\cdot|$ cardinality, $\mathrm{E}\{\cdot\}$ the mathematical expectation and $\|\cdot\|$ the Euclidean norm.

II. Distributed Adaptive Filtering

Let us consider a network of V nodes with a predefined topology. Two nodes are considered neighbors if they can exchange information, and we denote by $\mathcal{N}_k$ the neighborhood of node k including k itself. Each node k has access to an input signal $u_k(n)$ and to a reference signal $d_k(n) = \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}^{\mathrm{o}} + v_k(n)$, where $\mathbf{u}_k(n) = [u_k(n)\ \ u_k(n-1)\ \cdots\ u_k(n-M+1)]^{\mathrm{T}}$ is an M-length regressor vector, $\mathbf{w}^{\mathrm{o}}$ is an optimal system, and $v_k(n)$ is the measurement noise at node k, which is assumed to be independent of the other variables and zero-mean with variance $\sigma_{v_k}^2$. The objective of the network is to obtain an estimate of $\mathbf{w}^{\mathrm{o}}$ in a distributed manner by solving $\min_{\mathbf{w}} \sum_{k=1}^{V} \mathrm{E}\{|d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}|^2\}$ [1]–[3], [10], [11].

Several adaptive solutions have been proposed in the literature for this task, one of them being the ATC dNLMS algorithm [1]–[3]. It consists in two steps, given by

$$\begin{cases} \boldsymbol{\psi}_k(n+1) = \mathbf{w}_k(n) + \mu_k(n)\mathbf{u}_k(n)e_k(n) & \text{(1a)} \\ \mathbf{w}_k(n+1) = \sum_{j\in\mathcal{N}_k} c_{jk}\boldsymbol{\psi}_j(n+1), & \text{(1b)} \end{cases}$$

where $\boldsymbol{\psi}_k$ and $\mathbf{w}_k$ represent respectively the local and combined estimates of $\mathbf{w}^{\mathrm{o}}$ at node k, $\mu_k(n) = \widetilde{\mu}_k/[\delta + \|\mathbf{u}_k(n)\|^2]$ is a normalized step size with $0 < \widetilde{\mu}_k < 2$ and a regularization parameter $\delta > 0$, and

$$e_k(n) = d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_k(n) \tag{2}$$

is the estimation error [1]. Furthermore, $\{c_{jk}\}$ are combination weights satisfying $c_{jk} \geqslant 0$, $\sum_{j\in\mathcal{N}_k} c_{jk} = 1$, and $c_{jk} = 0$ for $j \notin \mathcal{N}_k$ [2], [3]. Possible choices for $\{c_{jk}\}$ include the Uniform, Laplacian, Metropolis, and Relative Degree rules [1], as well as adaptive schemes [4]–[6], such as the Adaptive Combination Weights (ACW) algorithm [19]. It incorporates information from the noise profile across the network, and

This work was supported by FAPESP under Grant 2017/20378-9, by CNPq under Grants 132586/2018-5 and 304715/2017-4, and by CAPES under Finance Code 001.

is obtained by solving an optimization problem in regards to $\{c_{jk}\}$. It can be summarized as [19]

$$c_{jk}(n) = \frac{\sigma_{jk}^{-2}(n)}{\sum_{\ell\in\mathcal{N}_k}\sigma_{\ell k}^{-2}(n)} \text{ if } j\in\mathcal{N}_k \text{ or } 0, \text{otherwise}, \quad (3)$$

where σ_{jk}^2 is updated as

$$\sigma_{jk}^2(n) = (1-\nu_k)\sigma_{jk}^2(n-1) + \nu_k\|\boldsymbol{\psi}_j(n+1) - \mathbf{w}_k(n)\|^2, \quad (4)$$

with $\nu_k > 0$ for $k = 1, \cdots, V$. Hence, larger weights are assigned to the nodes with smaller noise variances [19].

It is worth noting that one could also employ a Combine-Then-Adapt (CTA) strategy, in which the order of (1a) and (1b) is reversed [1]. For simplicity, in this paper we only consider the ATC strategy in our analysis, but the results can be straightforwardly extended to CTA versions as well.

III. The sampling algorithm

We propose an algorithm to decide if each node of the network should be sampled or not at each iteration. For this purpose, we introduce the variable $\bar{s}_k(n) \in \{0,1\}$ and recast (1a) as

$$\boxed{\boldsymbol{\psi}_k(n+1) = \mathbf{w}_k(n) + \bar{s}_k(n)\mu_k(n)\mathbf{u}_k(n)e_k(n).} \quad (5)$$

If $\bar{s}_k(n) = 1$, $d_k(n)$ is sampled, $e_k(n)$ is computed as in (2), the combination weights are updated according to (3) and (4), and (5) coincides with (1a). In contrast, if $\bar{s}_k(n) = 0$, $d_k(n)$ is not sampled, $\mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_k(n)$, $e_k(n)$ and $\mu_k(n)$ are not computed, $\{c_{jk}\}$ are not updated, and $\boldsymbol{\psi}_k(n+1) = \mathbf{w}_k(n)$.

To determine $\bar{s}_k(n)$, we define $s_k(n) \in [0,1]$ such that $\bar{s}_k(n) = 0$ for $s_k(n) < 0.5$ and $\bar{s}_k(n) = 1$ otherwise. We then minimize the following cost function with respect to $s_k(n)$:

$$J_{s,k}(n) = [s_k(n)]\beta\bar{s}_k(n) + \big[1 - s_k(n)\big]\sum_{j\in\mathcal{N}_k} c_{ik}(n)e_i^2(n), \quad (6)$$

where $\beta > 0$ is a parameter introduced to control how much the sampling of the nodes is penalized. Thus, when the error is high in magnitude or when node k is not being sampled ($\bar{s}_k = 0$), $J_{s,k}(n)$ is minimized by making $s_k(n)$ closer to one, leading to the sampling of node k. This ensures that the algorithm keeps sampling the nodes while the error is high and resumes the sampling of idle nodes at some point, enabling it to detect changes in the environment. In contrast, when node k is being sampled ($\bar{s}_k = 1$) and the error is small in magnitude in comparison to β, $J_{s,k}(n)$ is minimized by making $s_k(n)$ closer to zero, which leads the algorithm to stop sampling node k. This desirable behavior depends on a proper choice for β, which is addressed in Sec. IV.

Inspired by convex combination of adaptive filters (see [20], [21] and their references), rather than directly adjusting $s_k(n)$, we update an auxiliary variable $\alpha_k(n)$ related to it via [21]

$$s_k(n) = \phi[\alpha_k(n)] \triangleq \frac{\mathrm{sgm}[\alpha_k(n)] - \mathrm{sgm}[-\alpha^+]}{\mathrm{sgm}[\alpha^+] - \mathrm{sgm}[-\alpha^+]}, \quad (7)$$

where $\mathrm{sgm}[x] = (1+e^{-x})^{-1}$ is a sigmoidal function and α^+ is the maximum value α_k can assume. It is worth noting that $\phi[\alpha^+] = 1$ and $\phi[-\alpha^+] = 0$. In the literature, $\alpha^+ = 4$ is usually adopted [21].

By taking the derivative of (6) with respect to $\alpha_k(n)$, we obtain the following stochastic gradient descendent rule:

$$\alpha_k(n+1) = \alpha_k(n) + \mu_s\phi'[\alpha_k(n)]\Big[\sum_{i\in\mathcal{N}_k} c_{ik}(n)e_i^2(n) - \beta\bar{s}_k(n)\Big], \quad (8)$$

where $\mu_s > 0$ is a step size and

$$\phi'[\alpha_k(n)] \triangleq \frac{ds_k(n)}{d\alpha_k(n)} = \frac{\mathrm{sgm}[\alpha_k(n)]\{1 - \mathrm{sgm}[\alpha_k(n)]\}}{\mathrm{sgm}[\alpha^+] - \mathrm{sgm}[-\alpha^+]}. \quad (9)$$

It is worth noting that, because of the function $\phi[\cdot]$, (8) does not diverge even for large values of μ_s [20], [21].

Equation (8) cannot be used for sampling since it requires the errors to be computed to decide if the nodes should be sampled or not, which is contradictory. To address this issue, we replace $e_i(n)$ in (8) by its latest measurement we have access to, which is denoted by $\varepsilon_i(n)$. When the node is sampled, $\varepsilon_i(n) = e_i(n)$. We thus obtain

$$\boxed{\alpha_k(n+1) = \alpha_k(n) + \mu_s\phi'[\alpha_k(n)]\Big[\sum_{i\in\mathcal{N}_k} c_{ik}(n)\varepsilon_i^2(n) - \beta\bar{s}_k(n)\Big].} \quad (10)$$

This algorithm is named as adaptive sampling diffusion NLMS (AS-dNLMS). It reduces the number of sampled nodes in steady state, decreasing the computational cost at the expense of a slight increase during the transient. Table I shows the number of sums and multiplications executed per iteration in a single node of the network for both the dNLMS and AS-dNLMS algorithms with ACW weights. When the node is sampled, AS-dNLMS requires $\sum_{i\in\mathcal{N}_k}\bar{s}_i(n) + 2$ more multiplications and $|\mathcal{N}_k| + 1$ more additions than the original dNLMS algorithm. On the other hand, when the node is not sampled, AS-dNLMS requires $3M + 2 - \sum_{i\in\mathcal{N}_k}\bar{s}_i(n)$ less multiplications and $4M - |\mathcal{N}_k| + 1$ less sums. Thus, the higher the order of the filter M, the higher the computational cost reduction of AS-dNLMS in comparison with the original dNLMS algorithm. Considering the network as a whole, the computational cost of AS-dNLMS depends on the number of sampled nodes, which is addressed in Sec. IV.

Finally, we remark that a different version of AS-dNLMS can be obtained if, instead of using (5), $\boldsymbol{\psi}_k$ is not updated at all when node k is not sampled. Assuming that the nodes can store past information from their neighbors, this allows us to cut the number of communications between nodes, since in this case $\boldsymbol{\psi}_k$ and ε_k^2 remain static when $\bar{s}_k = 0$ and there is no need for node k to retransmit them. In other words, when node k is not sampled in this version of the algorithm, it only receives data and carries out (1b), and can thus turn its transmitter off. This results in a reduction in energy consumption as well as the computational cost. Lastly, when the node is sampled, $\varepsilon_i^2(n) = e_i^2(n)$ can be sent bundled with the local estimates $\boldsymbol{\psi}_i$ in both versions of AS-dNLMS so as to not increase the number of transmissions.

IV. Theoretical analysis

The good behavior of AS-dNLMS depends on a proper choice for β. Thus, we study how to choose this parameter such that we can ensure that every node will cease to be sampled at some point during steady state. To do so, we examine (10) while node k is being sampled. In this case, $\varepsilon_i^2(n)$ and $\beta\bar{s}_k(n)$ can be replaced by $e_i^2(n)$ and β, respectively. Then, subtracting $\alpha_k(n)$ from both sides of (10) and taking expectations, we get

$$\mathrm{E}\{\Delta\alpha_k(n)\} = \mu_s\mathrm{E}\Big\{\phi'[\alpha_k(n)]\Big[\sum_{i\in\mathcal{N}_k} c_{ik}(n)e_i^2(n) - \beta\Big]\Big\}, \quad (11)$$

TABLE I: Comparison between dNLMS and AS-dNLMS: number of operations per iteration for each node k.

Algorithm	Multiplications ($\otimes$)	Sums ($\oplus$)
dNLMS	$M(3+\lvert\mathcal{N}_k\rvert)+4$	$M(3+\lvert\mathcal{N}_k\rvert)+3$
AS-dNLMS	$\bar{s}_k(n)\cdot(3M+4)+M\lvert\mathcal{N}_k\rvert+\sum_{i\in\mathcal{N}_k}\bar{s}_i(n)+2$	$\bar{s}_k(n)\cdot(4M+2)+M\lvert\mathcal{N}_k\rvert-M+\lvert\mathcal{N}_k\rvert+2$

where $\Delta\alpha_k(n) \triangleq \alpha_k(n+1)-\alpha_k(n)$.

To make the analysis more tractable, $\phi'[\alpha_k(n)]$ and the term between brackets in (11) are assumed statistically independent. We also assume that the combination weights $\{c_{ik}\}$ are static. Simulation results show that these assumptions are reasonable and the analysis also holds for adaptive combination weights, as can be seen in Section V.

In order to stop sampling node k, $\alpha_k(n)$ should decrease along the iterations until it becomes negative. Since $\phi'[\alpha_k(n)]$ is always positive, to enforce $\mathrm{E}\{\Delta\alpha_k(n)\}$ to be negative while node k is sampled, β must satisfy

$$\beta > \textstyle\sum_{i\in\mathcal{N}_k} c_{ik}\mathrm{E}\{e_i^2(n)\}. \tag{12}$$

We then assume that, during steady state, $\mathrm{E}\{e_i^2(n)\} \approx \sigma_{v_i}^2$, which leads to

$$\textstyle\sum_{i\in\mathcal{N}_k} c_{ik}\mathrm{E}\{e_i^2(n)\} \leqslant \sigma_{\max}^2 \triangleq \max_i \sigma_{v_i}^2, \tag{13}$$

where $i = 1, 2, \cdots, V$. Thus, the condition

$$\beta > \sigma_{\max}^2 \tag{14}$$

is sufficient to ensure that, in the mean, the nodes will cease to be sampled during steady state.

Assuming that (14) is satisfied, we can estimate upper and lower bounds for the expected number of sampled nodes V_s in steady state. For this purpose, we consider each $\bar{s}_k(n)$ as a Bernoulli random variable that is equal to one with probability p_{s_k} or to zero with probability $1-p_{s_k}$ in steady state for $k = 1, \cdots, V$. Thus,

$$V p_{s_{\min}} \leqslant \mathrm{E}\{V_s\} \leqslant V p_{s_{\max}}, \tag{15}$$

where $p_{s_{\min}}$ and $p_{s_{\max}}$ are upper and lower bounds for p_{s_k}.

It is useful to note that the sampling mechanism exhibits a cyclic behavior in steady state. Hence, we could approximate p_{s_k} by the expected "duty cycle" of the mechanism, i.e.,

$$\widehat{p}_{s_k} = \theta_k/(\theta_k + \overline{\theta}_k), \tag{16}$$

where θ_k denotes the expected number of iterations per cycle in which node k is sampled and $\overline{\theta}_k$ is the expected number of iterations in which it is not. Since we are only interested in estimating $p_{s_{\min}}$ and $p_{s_{\max}}$, we do not have to evaluate (16) for every k. Instead, we only need to estimate upper and lower bounds for θ_k and $\overline{\theta}_k$. To do so, we must understand under which circumstances node k remains sampled for the greatest (or lowest) number of iterations in the mean. One way to do this is to estimate the maximum and minimum values $\mathrm{E}\{\alpha_k(n)\}$ and $\mathrm{E}\{\Delta\alpha_k(n)\}$ can assume during steady state.

Firstly, let us assume that at a certain iteration n, $\alpha_k(n)$ is negative but close to zero. Thus, setting $\alpha_k(n)$ to zero in (10) and taking expectations, we obtain

$$\mathrm{E}\{\alpha_k(n+1)|\alpha_k(n) \lessapprox 0\} = \mu_s\phi_0' \textstyle\sum_{i\in\mathcal{N}_k} c_{ik}\mathrm{E}\{\varepsilon_i^2(n)\}, \tag{17}$$

where $\phi_0'=\phi'[0]$. Thus, at n+1 the sampling of node k resumes and, recalling (14), $\mathrm{E}\{\Delta\alpha_k(n+1)|\alpha_k(n+1)>0\}<0$. Therefore, from iteration n+1 onward, α_k decreases until it becomes negative again, meaning that (17) yields the maximum value α_k can assume in the mean during steady state. Moreover, assuming $\sigma_{\min}^2 \leqslant \mathrm{E}\{\varepsilon_i^2(n)\} \leqslant \sigma_{\max}^2$ for all i, (17) yields a different value for each node k that lies in

$$\mu_s\phi_0'\sigma_{\min}^2 \leqslant \mathrm{E}\{\alpha_{k_{\max}}^{\text{s.s.}}\} \leqslant \mu_s\phi_0'\sigma_{\max}^2, \tag{18}$$

where $\mathrm{E}\{\alpha_{k_{\max}}^{\text{s.s.}}\}$ denotes the maximum value $\alpha_k(n)$ can assume in the mean in steady state and $\sigma_{\min}^2 \triangleq \min_i \sigma_{v_i}^2$, $i = 1, \cdots, V$. Analogously, we now assume that at a certain iteration n, $\alpha_k(n)$ is positive but approximately zero. Making this replacement in (10) and taking expectations, we obtain

$$\mathrm{E}\{\alpha_k(n+1)|\alpha_k(n) \gtrapprox 0\} = \mu_s\phi_0'\mathrm{E}\left\{\textstyle\sum_{i\in\mathcal{N}_k} c_{ik}\varepsilon_i^2(n) - \beta\right\}. \tag{19}$$

Thus, at $n+1$, node k ceases to be sampled. Therefore, we observe from (14) that $\mathrm{E}\{\Delta\alpha_k(n)|\alpha_k(n+1)<0\}>0$ and conclude that (19) provides the minimum value α_k can assume in the mean during steady state. For each node k, (19) yields a different value that lies in the interval

$$\mu_s\phi_0'(\sigma_{\min}^2 - \beta) \leqslant \mathrm{E}\{\alpha_{k_{\min}}^{\text{s.s.}}\} \leqslant \mu_s\phi_0'(\sigma_{\max}^2 - \beta), \tag{20}$$

where $\mathrm{E}\{\alpha_{k_{\min}}^{\text{s.s.}}\}$ denotes the minimum value $\alpha_k(n)$ can assume in the mean in steady state.

Next, we replace $\phi'[\alpha_k(n)]$ in (10) by its first-order Taylor expansion around $\alpha_k(n)=0$, which is simply equal to the constant ϕ_0'. When node k is being sampled ($\bar{s}_k=1$), subtracting $\alpha_k(n)$ from both sides of (10) and taking expectations yields

$$-\mu_s\phi_0'(\beta-\sigma_{\min}^2) \leqslant \mathrm{E}\{\Delta\alpha_k(n)\} \leqslant -\mu_s\phi_0'(\beta-\sigma_{\max}^2) < 0. \tag{21}$$

Analogously, when the node is not sampled ($\bar{s}_k(n)=0$),

$$\mu_s\phi_0'\sigma_{\min}^2 \leqslant \mathrm{E}\{\Delta\alpha_k(n)\} \leqslant \mu_s\phi_0'\sigma_{\max}^2. \tag{22}$$

Thus, in both cases there are upper and lower bounds for $\mathrm{E}\{\Delta\alpha_k(n)\}$ during steady state.

From a certain iteration n_0 onward, we consider the model

$$\mathrm{E}\{\alpha_k(n_0+\theta_k)\} = \mathrm{E}\{\alpha_k(n_0)\} + \theta_k\mathrm{E}\{\Delta\alpha_k(n)\}. \tag{23}$$

In order to estimate an upper bound $\theta_{\max}$ for θ_k, we assume that $\mathrm{E}\{\alpha_k(n_0)\} = \mathrm{E}\{\alpha_{k_{\max}}^{\text{s.s.}}\}$ and calculate the expected number of iterations required for $\mathrm{E}\{\alpha_k(n)\}$ to fall below zero in the scenario where the node is sampled for the maximum number of iterations. This occurs if $\mathrm{E}\{\alpha_k(n_0)\} = \mu_s\phi_0'\sigma_{\max}^2$, which is the upper bound for $\mathrm{E}\{\alpha_{k_{\max}}^{\text{s.s.}}\}$, and $\mathrm{E}\{\Delta\alpha_k(n)\} = -\mu_s\phi_0'(\beta-\sigma_{\max}^2)$, which is the least negative variation for $\mathrm{E}\{\Delta\alpha_k(n)\}$ according to (21). Making $\theta_k = \theta_{\max}$ and setting $\mathrm{E}\{\alpha_k(n_0+\theta_{\max})\} = 0$ in (23), after some algebra we get

$$\theta_{\max} = \max\{\sigma_{\max}^2/(\beta-\sigma_{\max}^2),\ 1\}, \tag{24}$$

where we are taking into account the fact that the node must be sampled at least once during each cycle. Analogously, using (23) for the lower bound $\theta_k = \theta_{\min}$, we obtain

$$\theta_{\min} = \max\{\sigma_{\min}^2/(\beta-\sigma_{\min}^2),\ 1\}. \tag{25}$$

For $\overline{\theta}_k$, we replace θ_k in (23) by $\overline{\theta}_k$ and consider that at the iteration n_0, $\mathrm{E}\{\alpha_k(n_0)\} = \mathrm{E}\{\alpha_{k_{\min}}^{\text{s.s.}}\}$. Thus, the upper bound $\overline{\theta}_{\max}$ for $\overline{\theta}_k$ can be obtained by setting $\mathrm{E}\{\alpha_k(n_0)\} = \mu_s\phi_0'\sigma_{\min}^2$, which is the lower bound for $\mathrm{E}\{\alpha_{k_{\min}}^{\text{s.s.}}\}$, and

$\mathrm{E}\{\Delta\alpha_k(n)\} = \mu_s\phi_0'\sigma^2_{\min}$, which is the minimum value for $\mathrm{E}\{\Delta\alpha_k(n)\}$ according to (22). We then get

$$\overline{\theta}_{\max} = \max\{(\beta - \sigma^2_{\min})/\sigma^2_{\min},\ \ 1\}. \tag{26}$$

Analogously, for the lower bound $\overline{\theta}_{\min}$ of $\overline{\theta}_k$, we get

$$\overline{\theta}_{\min} = \max\{(\beta - \sigma^2_{\max})/\sigma^2_{\max},\ \ 1\}. \tag{27}$$

Replacing (24) to (27) in (16) and (15), after some algebraic manipulations we finally obtain that, for $\beta \geqslant \sigma^2_{\max}$,

$$\boxed{V\frac{\sigma^2_{\min}}{\beta} \leqslant \mathrm{E}\{V_s\} \leqslant V\frac{\sigma^2_{\max}}{\beta}.} \tag{28}$$

This indicates that the higher the parameter β, the smaller the amount of sampled nodes in the mean during steady state, which is in accordance with our expectations. Since there is a trade-off between the tracking capability and the gains in terms of computational cost provided by the sampling mechanism, we should care not to choose excessively high values for β. Simulation results suggest that $\beta > 5\sigma^2_{\max}$ can deteriorate the performance in non-stationary environments. It is also worth noting that the upper and lower bounds for $\mathrm{E}\{V_s\}$ coincide when $\sigma^2_{\min} = \sigma^2_{\max}$. This makes sense, since in this case there is no reason for some nodes to be sampled more often than the others in steady state. Furthermore, although we initially assumed $\beta > \sigma^2_{\max}$, it is interesting to note that (28) also holds for $\beta = \sigma^2_{\max}$, since in this case the theoretical upper bound for $\mathrm{E}\{V_s\}$ is equal to the total number of nodes in the network.

V. Simulation Results

In this section, we test the proposed algorithm and the analysis of Sec. IV. The results presented were obtained over an average of 100 realizations. For the sake of better visualization, we filtered the curves by a moving-average filter with 64 coefficients. We consider the network shown in Fig. 1(a). The signals $u_k(n)$ and $v_k(n)$ are generated from i.i.d. zero-mean Gaussian random processes with variances $\sigma^2_{u_k} = 1$ and $\sigma^2_{v_k}$ as shown in Fig. 1(b) for $k = 1, \cdots, V$. For the optimal system $\mathbf{w}^{\mathrm{o}}$, we consider a random vector with $M = 50$ coefficients uniformly distributed in $[-1, 1]$.

To set the combination weights, we use the ACW algorithm with $\nu_k = 0.2$ for $k = 1, \cdots, V$ [19]. We also use $\delta = 10^{-5}$ and different values of $\tilde{\mu}_k$ for each node k, as shown in Fig. 1(c). As a performance indicator, we adopt the network mean-square deviation (MSD), given by $\frac{1}{V}\sum_{k=1}^{V}\mathrm{E}\{\|\mathbf{w}^{\mathrm{o}}(n) - \mathbf{w}_k(n)\|^2\}$. Furthermore, in the simulations of Figs. 2 and 4 we consider $\beta = 1.7\sigma^2_{\max} = 0.68$ and $\mu_s = 0.1571$ for the AS-dNLMS algorithm. These values were chosen due to the good performance they provided in terms of MSD, computational cost reduction and energy saving in these simulations.

Firstly, we compare the behavior of the AS-dNLMS algorithm with that of the original dNLMS with a random sampling technique in which V_s nodes are randomly sampled at each iteration. In order to simulate a change in the environment, in the middle of each realization we flip $\mathbf{w}^{\mathrm{o}}$. Figs. 2(a), 2(b) and 2(c) present respectively the MSD performance and the average number of sums and multiplications per iteration. We can observe from Fig. 2(a) that the more nodes are sampled, the faster the convergence rate. AS-dNLMS is able to detect

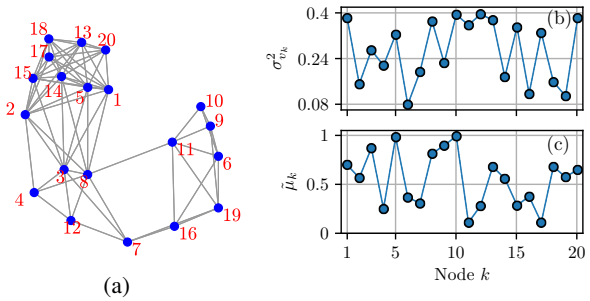

Fig. 1: (a) Network topology, (b) $\sigma^2_{v_k}$, and (c) $\tilde{\mu}_k$ used in the experiments..

the change in the optimal system and, since all nodes are sampled during the transients, it converges as fast as the dNLMS algorithm with all nodes sampled. From Figs. 2(b) and 2(c) we also observe that during the transients the computational cost of AS-dNLMS is slightly higher than that of the dNLMS algorithm with all nodes sampled, but decreases significantly after AS-dNLMS converges and ceases to sample every node at every iteration.

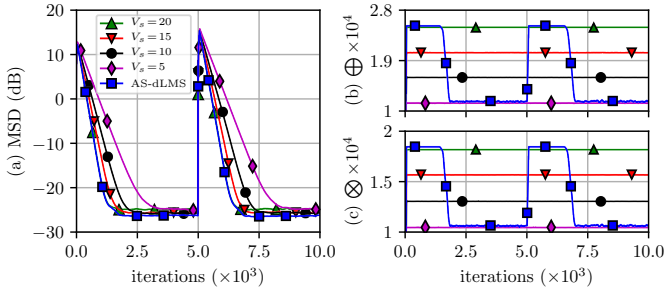

Fig. 2: Comparison between dNLMS with a random sampling technique with different amounts of sampled nodes and AS-dNLMS ($\beta = 0.68$, $\mu_s = 0.1571$). (a) MSD curves, (b) Sums, and (c) Multiplications per iteration.

In Fig. 3 we present simulation results showing the average number of sampled nodes during steady state in a stationary environment, as well as the theoretical bounds given by (28) for different values of $\beta/\sigma^2_{\max} \geqslant 1$. We can see that the higher β is, the less nodes are sampled, as expected. Furthermore, the experimental results lie between the theoretical curves for all values of $\beta/\sigma^2_{\max}$, validating the results of Sec. IV.

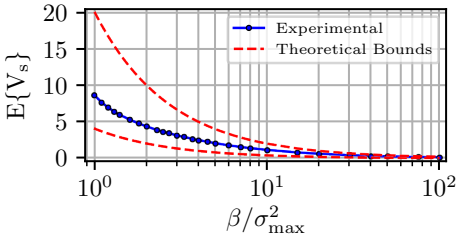

Fig. 3: Average number of sampled nodes during steady state and comparison with theoretical bounds of (28).

Finally, we consider the energy-saving version of AS-dNLMS in which node k does not communicate with its neighbors when it is not sampled. To assess its performance, we compare it with the ACW-Selective algorithm of [11] (ACW-S), the partial-update algorithm of [13] (PU-dNLMS), and the dNLMS algorithm with a probabilistic transmission strategy in which each link of the network is active at a

certain iteration n with probability p_k (PT-dNLMS) [7]. In our simulations, we adjusted the parameters of all the algorithms to obtain roughly the same level of MSD during steady state. For comparison, we also present the results obtained with the original dNLMS algorithm and with the non-cooperative case. In Fig. 4(a) we present the MSD performance, and in Fig. 4(b) the number of communication processes per iteration $t(n)$. To enable the comparison with the PU-dNLMS algorithm, we scaled the number of communication processes by the ratio of data sent in each transmission in this plot. We observe that AS-dNLMS initially requires just as many transmissions as the original dNLMS algorithm, but this number drastically decreases after it converges. During steady state, it led to the lowest number of communication processes among all the solutions tested. It is interesting to note that, in a scenario where the nodes are able to broadcast their estimates to all their neighbors at once, the comparison shown in Fig. 4(b) is unfair with the AS-dNLMS and ACW-S algorithms, since in this case they are the only solutions that would lead to an actual reduction in the number of communication processes.

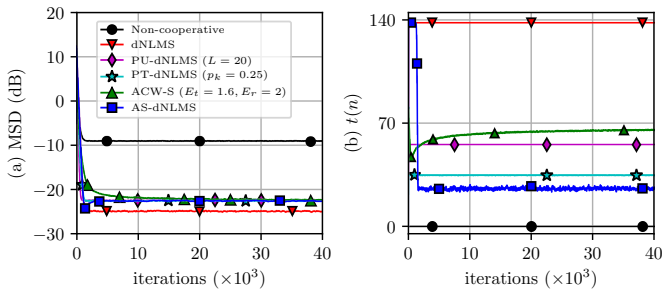

Fig. 4: Comparison between the energy-saving version of AS-dNLMS ($\beta = 0.68$, $\mu_s = 0.1571$) and other techniques found in the literature [7], [11], [13]. (a) MSD, and (b) Communications per iteration.

VI. Conclusions

In this paper, we generalize the sampling mechanism of [17] for adaptive diffusion networks. The proposed mechanism uses the information from more nodes when the error in the network is high and less nodes otherwise. Besides reducing the computational cost, it can be used to save energy by avoiding transmissions of nodes that are not sampled. We observed from simulations that AS-dNLMS maintains the convergence rate of dNLMS during transient while displaying a lower computational cost in steady state. The energy-saving version of AS-dNLMS presents a slight increase in steady-state MSD, but still exhibits a good tradeoff between performance and energy consumption. The theoretical bounds for the number of sampled nodes obtained in Sec. IV present a good agreement with simulations, and are useful for the proper choice of algorithm parameters. It should be mentioned that, although we compared the proposed AS-dNLMS algorithm with other techniques in Sec. V, it may be used in conjunction with these methods, as well as many others [9], [11]–[14], to further reduce the computational cost and the energy consumption associated with the communication processes. For future work, we intend to compare AS-dNLMS with other state-of-the-art censoring mechanisms [12] and test it with other diffusion schemes, such as decoupled algorithms [6].

References

[1] A. H. Sayed, *Adaptation, Learning, and Optimization over Networks*, vol. 7, Foundations and Trends in Machine Learning, now Publishers Inc., Hanover, MA, 2014.

[2] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, pp. 3122–3136, 2008.

[3] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, pp. 1035–1048, 2009.

[4] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 58, pp. 4795–4810, 2010.

[5] C.-K. Yu and A. H. Sayed, "A strategy for adjusting combination weights over adaptive networks," in *Proc. IEEE ICASSP*, Vancouver, Canada, 2013, pp. 4579–4583.

[6] J. Fernandez-Bes, , J. Arenas-García, M. T. M. Silva, and L. A. Azpicueta-Ruiz, "Adaptive diffusion schemes for heterogeneous networks.," *IEEE Trans. Signal Process.*, vol. 65, pp. 5661–5674, 2017.

[7] C. G. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proc.IEEE ICASSP*, Las Vegas, NV, 2008, pp. 3285–3288.

[8] X. Zhao and A. H. Sayed, "Single-link diffusion strategies over adaptive networks," in *Proc. IEEE ICASSP*, Kyoto, Japan, 2012, pp. 3749–3752.

[9] S. Xu, R. C. de Lamare, and H. V. Poor, "Adaptive link selection algorithms for distributed estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, no. 1, pp. 86, 2015.

[10] N. Takahashi and I. Yamada, "Link probability control for probabilistic diffusion least-mean squares over resource-constrained networks," in *Proc. IEEE ICASSP*, Dallas, TX, 2010, pp. 3518–3521.

[11] R. Arroyo-Valles, S. Maleki, and G. Leus, "A censoring strategy for decentralized estimation in energy-constrained adaptive diffusion networks," in *Proc. of IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Darmstadt, Germany, 2013, pp. 155–159.

[12] J. Fernandez-Bes, R. Arroyo-Valles, J. Arenas-García, and J. Cid-Sueiro, "Censoring diffusion for harvesting WSNs," in *Proc. of IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, Cancun, Mexico, 2015, pp. 237–240.

[13] R. Arablouei, S. Werner, Y. Huang, and K. Doğançay, "Distributed least mean-square estimation with partial diffusion," *IEEE Trans. Signal Process.*, vol. 62, no. 2, pp. 472–484, 2013.

[14] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Trading off complexity with communication costs in distributed adaptive learning via Krylov subspaces for dimensionality reduction," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 257–273, 2013.

[15] D. K. Berberidis, V. Kekatos, G. Wang, and G. B. Giannakis, "Adaptive censoring for large-scale regressions," in *Proc. IEEE ICASSP*, Brisbane, Australia, 2015, pp. 5475–5479.

[16] L. Yang, H. Zhu, K. Kang, X. Luo, H. Qian, and Y. Yang, "Distributed censoring with energy constraint in wireless sensor networks," in *Proc. IEEE ICASSP*, Calgary, Canada, 2018, pp. 6428–6432.

[17] D. G. Tiglea, R. Candido, and M. T. M. Silva, "An adaptive sampling technique for graph diffusion LMS algorithm," in *Proc. of European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019, pp. 1364–1368.

[18] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "Distributed diffusion adaptation over graph signals," in *Proc. IEEE ICASSP*, Brighton, UK, 2018, pp. 4129–4133.

[19] S. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, San Juan, Puerto Rico, 2011, pp. 317–320.

[20] J. Arenas-Garcia, L. A. Azpicueta-Ruiz, M. T. M. Silva, V. H. Nascimento, and A. H. Sayed, "Combinations of adaptive filters: performance and convergence properties," *Signal Processing Magazine*, vol. 33, pp. 120–140, 2016.

[21] M. Lázaro-Gredilla, L. A. Azpicueta-Ruiz, A. R. Figueiras-Vidal, and J. Arenas-Garcia, "Adaptively biasing the weights of adaptive filters," *IEEE Trans. Signal Process.*, vol. 58, pp. 3890–3895, 2010.