

On the Impact of Random Node Sampling on Adaptive Diffusion Networks

Daniel G. Tiglea, Renato Candido, and Magno T. M. Silva, *Member, IEEE*

Abstract—In this paper, we analyze the effects of random sampling on adaptive diffusion networks. These networks consist in a collection of nodes that can measure and process data, and that can communicate with each other to pursue a common goal of estimating an unknown system. In particular, we consider in our theoretical analysis the diffusion least-mean-squares algorithm in a scenario in which the nodes are randomly sampled. Hence, each node may or may not adapt its local estimate at a certain iteration. Our model shows that a reduction in the sampling probability leads to a noticeable deterioration in the convergence rate, and, if the nodes cooperate, to a slight decrease in the steady-state Network Mean-Square Deviation (NMSD), assuming that the environment is stationary and that all other parameters of the algorithm are kept fixed. Furthermore, we also investigate the effects of the random node sampling on the network stability.

Index Terms—Adaptive diffusion networks, distributed signal processing, sampling, stability, asynchronous networks.

I. INTRODUCTION

ADAPTIVE diffusion networks have attracted widespread attention in the distributed signal processing literature over the past decade and a half, and have become consolidated tools in the area [1]–[10]. They consist in a set of connected *agents*, or *nodes*, that are able to measure and process data locally, and that can communicate with other nodes in their vicinity. The network formed by these agents has a collective goal to estimate a parameter vector of interest, without the need for a central processing unit [1]–[10]. Hence, they present better flexibility, scalability, and robustness than centralized approaches, whose central unit represents a critical point of failure, and limits the area where the nodes can be deployed.

In order to enable the distributed learning of the parameters, each node usually computes its own *local* estimate in what is known as the *adaptation step*. Then, the neighboring nodes cooperate to reach a *global* estimate of the vector of interest. This stage is usually referred to as the *combination step* [1]–[11]. Due to their advantages in comparison with other distributed settings, such as the incremental [12], [13] and consensus [14], [15] strategies, adaptive diffusion networks have branched out into many different research topics. Examples include multitask networks [16]–[19], nonlinear adaptive networks [20]–[23], among others. Moreover, the field of graph signal processing (GSP) has drawn inspiration from these techniques, since its applications are usually distributed

in nature [9], [10], [24], [25]. Hence, many graph adaptive filtering algorithms can be seen as an extension of adaptive diffusion networks to domains where space, as well as time, plays a role in the development of the signals of interest [26].

For the feasibility of these solutions, it is oftentimes desirable to restrict the amount of data measured and processed by the nodes. However, this process can have a negative effect on the network performance. For this purpose, we proposed in [26] and [28] adaptive mechanisms that control the amount of data measured and processed by the nodes according to the estimation error, so as to mitigate the impacts on the performance. In both papers, we adopted a random *sampling* technique as a benchmark to compare the proposed solutions with. In this scheme, a node that is sampled adapts its estimates normally, while a node that is not sampled is exempt from performing certain calculations. When using this random sampling method, two things occur as we decrease the number of nodes sampled per iteration. In the steady state, the Network Mean-Square Deviation (NMSD), a performance indicator, slightly decreases, so long as the environment is stationary and the nodes cooperate, but the convergence rate deteriorates noticeably in the transient phase. Next, we provide some preliminary simulation results to illustrate this. The discussion on the results presented will help us motivate the present work.

A. Introductory Simulations and Motivation

We show in Fig. 1 simulation results obtained with the adapt-then-combine diffusion Least-Mean-Squares (ATC dLMS) algorithm [1]–[5] with a random sampling technique. In this setup, each node is sampled with probability p_ζ , or not sampled with probability $1 - p_\zeta$. This algorithm will be revisited in detail in Sec. II. We consider the Scenario 1 described in Sec. V, and compare the NMSD of the algorithm along the iterations for different values of $p_\zeta \in \{0.1, 0.25, 0.5, 0.75, 1\}$.

From Fig. 1, we can clearly see that as we decrease the sampling probability, the convergence rate becomes evidently slower, while the steady-state NMSD is slightly reduced. In relation to the case in which every node is sampled, i.e., $p_\zeta = 1$, the curve obtained with $p_\zeta = 0.1$ achieves a steady-state NMSD that is approximately 3.5 dB lower, but this comes at the expense of a noticeably impaired convergence rate. In order to better understand these phenomena, our goal in this paper is to investigate the effects of random sampling on adaptive diffusion networks.

This work was supported by CAPES under Grant 88887.512247/2020-00 and Finance Code 001, by CNPq under Grants 303826/2022-3 and 404081/2023-1, and by FAPESP under Grant 2021/02063-6. The authors are with the Electronic Systems Engineering Department, Escola Politécnica, University of São Paulo, São Paulo, SP, Brazil, e-mails: {dtiglea, renato-can, magno}@lps.usp.br, ph. +55-11-3091-5134.

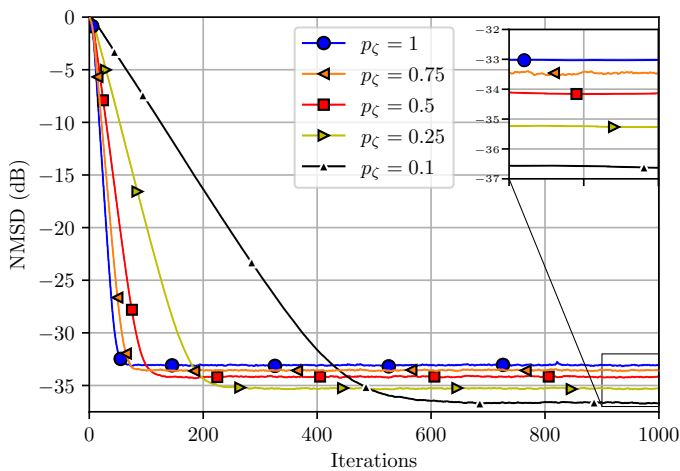


Fig. 1: Performance of the ATC dLMS algorithm with a random sampling technique, considering different sampling probabilities p_ζ , in the Scenario 1 described in Sec. V.

B. Relations with Other Works and Major Contributions

To the best of our knowledge, interpretations for these phenomena have not been provided in the literature. For example, in [29]–[31], a scenario was considered in which some of the nodes are not capable of performing the adaptation step. These are referred to as “uninformed” nodes, in contrast with the “informed” nodes, which can perform it. In those works, it was shown that, in comparison with a network in which every node is informed, the convergence rate is deteriorated, but the steady-state NMSD can decrease, increase, or remain unchanged, as we turn some of the nodes into uninformed ones. However, differently from the scenario considered in the simulations of Fig. 1, informed nodes carry out the adaptation step at every time instant, whereas uninformed nodes never do so. This differs from the behavior observed in Fig. 1, in which the nodes are sampled randomly. Furthermore, in [32]–[34], the authors study networks in which the adaptation and combination steps are not necessarily carried out simultaneously by all the nodes at every iteration. These networks are referred to as “asynchronous”, in contrast with the “synchronous” ones that appear, e.g., in [1]–[5]. From this perspective, the random-sampling network used in the simulations of Fig. 1 can be deemed as a type of asynchronous network. However, in those papers, the phenomena seen in Fig. 1 were not illustrated.

Next, we provide a list of the most relevant contributions of this paper in comparison with previous works:

- We obtain theoretical results that describe the effects of the random sampling of the nodes on the transient and steady-state behaviors of the dLMS algorithm observed in Fig. 1. Furthermore, and perhaps more importantly, our analysis helps us explain *why* these effects occur, and to what we may attribute them;
- Different from the studies conducted in [32]–[34], which are based on the Energy Conservation Argument – a powerful tool for the analysis of adaptive algorithms [35] –, our analysis utilizes a traditional statistical framework [36], [37], which facilitates the interpretation of

some results;

- Several simulation results are presented considering different scenarios, therefore abundantly illustrating the phenomena of interest and the theoretical results. In particular, we present simulation results that showcase the effects of the sampling of the nodes on the stability of the algorithm, which depends on the selection of the combination weights and on the network topology;
- We study the impact of the random node sampling on the computational cost.

We remark that our goal in this paper is not to motivate the usage of the random sampling technique of Fig. 1, but just to study the effects of the sampling of the nodes. As our analysis will show, the random sampling hinders the performance of the network in comparison with the case in which every node is sampled, if the step sizes are adjusted to obtain the same convergence rate with both versions of the algorithm. Regardless, we believe that a better understanding of the effects of sampling can lead to some insights into the inner workings of adaptive diffusion networks.

C. Organization of the Paper and Notation

The remainder of this paper is organized as follows. In Sec. II, we present the problem formulation. In Sec. III, we conduct a theoretical analysis on the behavior of adaptive diffusion networks, and on the effects of sampling on them. In Sec. IV, the impact of sampling on the computational cost is analyzed. Finally, in Secs. V and VI, we present simulation results and the main conclusions of our work, respectively.

Notation. We use normal font letters for scalars, boldface lowercase letters for vectors, and boldface uppercase letters for matrices. Moreover, $(\cdot)^T$ denotes transposition, $E\{\cdot\}$ the mathematical expectation, $[\cdot]_{\ell k}$ the element of a matrix at its ℓ -th row and k -th column, $\text{rank}[\cdot]$ the rank of a matrix, $\text{Tr}[\cdot]$ its trace, δ_{ij} the Kronecker delta function of i and j , $\|\cdot\|$ the Euclidean norm, $|\cdot|$ the absolute value, $/$ the difference between two sets, \otimes the Kronecker product, and \odot the Hadamard product. We denote the $L \times L$ identity matrix by \mathbf{I}_L , an L by M matrix of zeros by $\mathbf{0}_{L \times M}$, and an L by M matrix of ones by $\mathbf{1}_{L \times M}$. When referring to L -length column vectors of zeros or ones, we respectively write $\mathbf{0}_L$ and $\mathbf{1}_L$. We denote by $\text{vec}\{\cdot\}$ the vectorization of a matrix by stacking all of its columns to form a column vector. Lastly, we denote by $\text{diag}\{\cdot\}$ the aggregation of the arguments into a diagonal matrix. To simplify the arguments, we assume real data throughout the paper.

II. PROBLEM FORMULATION

Let us consider a network consisting of V nodes, with labels $k \in \{1, \dots, V\}$. For each node k , we call the set of nodes with which it can communicate, including node k itself, its neighborhood, and we denote it by \mathcal{N}_k . We assume that each node k has access at each iteration n to an input signal $u_k(n)$ and to a desired signal $d_k(n)$, which we model as [1]–[5]

$$d_k(n) = \mathbf{u}_k^T(n) \mathbf{w}^o + v_k(n), \quad (1)$$

where \mathbf{w}° is an M -length column vector that represents an unknown system, oftentimes referred to as the optimal system in the adaptive filtering literature [1]–[5], and $\mathbf{u}_k(n) = [u_k(n) \ u_k(n-1) \ \cdots \ u_k(n-M+1)]^T$ is a regressor vector formed by the last M samples of the input signal. Finally, $v_k(n)$ is the measurement noise at node k .

The goal of the network is to obtain an estimate \mathbf{w} of \mathbf{w}° in a distributed manner by solving [1]–[5]

$$\min_{\mathbf{w}} J_{\text{global}}(\mathbf{w}) = \min_{\mathbf{w}} \sum_{k=1}^V J_k(\mathbf{w}), \quad (2)$$

where $J_k(\mathbf{w})$ are local cost functions at each node k . A common choice for the $J_k(\mathbf{w})$, $k = 1, \dots, V$ is the mean squared error (MSE) [35], [36], in which case each node k should seek, at every iteration n , to minimize [1]–[5]

$$J_k(\mathbf{w}) \triangleq \mathbb{E}\{[d_k(n) - \mathbf{u}_k^T(n)\mathbf{w}]^2\}. \quad (3)$$

To this end, each node k computes a local estimate of \mathbf{w}° in order to minimize its individual cost function $J_k(\mathbf{w})$. For this purpose, it uses the data available locally, as well as the estimates by neighboring nodes. This is known as the adaptation step. Then, the node k cooperates with its neighbors to form a combined estimate in what is called the combination step. Depending on how the adaptation step is carried out, different algorithms can be obtained. Examples include the dLMS algorithm [1]–[5], the diffusion recursive least squares (dRLS) [6], diffusion Normalized LMS (dNLMS) [26], [28], [38], diffusion Affine Projection Algorithm (dAPA) [7], among others [39], [40]. For simplicity, we focus our analysis in this paper on the dLMS algorithm, which is obtained by adopting an LMS-type of strategy [35], [36] for the update of the local estimates. Considering a scenario in which each node k may or may not update its local estimate at each iteration n , the adaptation and combination steps of the dLMS algorithm are respectively given by [1]–[5]

$$\begin{cases} \psi_k(n) = \mathbf{w}_k(n-1) + \mu_k \zeta_k(n) \mathbf{u}_k(n) e_k(n) & (4a) \\ \mathbf{w}_k(n) = \sum_{i \in \mathcal{N}_k} c_{ik} \psi_i(n). & (4b) \end{cases}$$

where ψ_k and \mathbf{w}_k are the local and combined estimates of \mathbf{w}° at node k , respectively, $\mu_k > 0$ is a step size,

$$e_k(n) = d_k(n) - \mathbf{u}_k^T(n) \mathbf{w}_k(n-1) \quad (5)$$

is the estimation error at node k , and $\zeta_k(n) \in \{0, 1\}$ is a binary variable such that $\zeta_k(n) = 1$ if node k is sampled, and $\zeta_k(n) = 0$ otherwise. In the former case, $\psi_k(n)$ is updated as usual in the dLMS algorithm [1]–[5]. In contrast, when node k is not sampled, $\mathbf{u}_k^T(n) \mathbf{w}_k(n-1)$ and $e_k(n)$ do not need to be computed, as (4a) becomes simply $\psi_k(n) = \mathbf{w}_k(n-1)$. Lastly, c_{ik} are combination weights satisfying [1]–[5]

$$c_{ik}(n) \geq 0, \quad \sum_{i \in \mathcal{N}_k} c_{ik}(n) = 1, \quad \text{and} \quad c_{ik}(n) = 0 \text{ for } i \notin \mathcal{N}_k. \quad (6)$$

There are several possible rules for the selection of the combination weights. For instance, if we adopt $c_{ik} = 1$ if $i = k$ and $c_{ik} = 0$ otherwise, this corresponds to a setup in

which the nodes do not exchange their local estimates. This is oftentimes referred to as the non-cooperative approach in the literature [1]–[5], and can be seen as a scenario in which V adaptive filters try to solve (2) isolated from each other. Cooperative strategies include the Uniform, Metropolis, and Hastings rules, among others [1]–[3]. Moreover, several adaptive schemes have been proposed in the literature [8], [41]–[43], in which the combination weights $c_{ik}(n)$ are adjusted along the iterations. For simplicity, in our analysis we only consider static combination weights. In Table I, we provide a summary of the static rules considered in the simulations.

TABLE I: Summary of some static rules for the selection of the combination weights most widely adopted in the literature.

| Name | Equations |
|--------------------|---|
| Non-coop. [1]–[5] | $c_{ik} = \begin{cases} 1, & \text{if } i = k \\ 0, & \text{otherwise} \end{cases}$ |
| Uniform [1]–[5] | $c_{ik} = \begin{cases} \frac{1}{ \mathcal{N}_k }, & \text{if } i \in \mathcal{N}_k \\ 0, & \text{otherwise} \end{cases}$ |
| Metropolis [1]–[5] | $c_{ik} = \begin{cases} \frac{1}{\max\{ \mathcal{N}_k , \mathcal{N}_i \}}, & \text{if } i \in \mathcal{N}_k / \{k\} \\ 1 - \sum_{i \in \mathcal{N}_k} c_{ik}, & \text{if } i = k \\ 0, & \text{otherwise} \end{cases}$ |

When (4a) and (4b) are performed in this order at each iteration, this corresponds to a configuration known in the literature as *adapt-then-combine* (ATC) [1]–[5]. The order of these steps could be reversed, resulting in the *combine-then-adapt* (CTA) configuration. In this paper, we focus on the ATC protocol, but the results can be extended to the CTA dLMS.

III. THEORETICAL ANALYSIS

In our analysis, we are especially interested in the NMSD, a commonly adopted performance indicator, defined as [1]

$$\text{NMSD}(n) \triangleq \frac{1}{V} \sum_{k=1}^V \text{MSD}_k(n), \quad (7)$$

in which MSD_k is the mean-square deviation at node k , given by

$$\text{MSD}_k(n) \triangleq \mathbb{E}\{\|\tilde{\mathbf{w}}_k(n)\|^2\}, \quad (8)$$

where we have introduced the weight-error vector

$$\tilde{\mathbf{w}}_k(n) \triangleq \mathbf{w}^\circ - \mathbf{w}_k(n). \quad (9)$$

For compactness of notation, it is convenient to introduce the quantities

$$\beta_{ij}(n) \triangleq \mathbb{E}\{\tilde{\mathbf{w}}_i^T(n) \tilde{\mathbf{w}}_j(n)\} \quad (10)$$

for $i = 1, \dots, V$ and $j = 1, \dots, V$. It is worth noting that

$$\beta_{kk}(n) = \mathbb{E}\{\|\tilde{\mathbf{w}}_k(n)\|^2\} = \text{MSD}_k(n). \quad (11)$$

We then introduce the matrix $\mathbf{B}(n)$ such that $[\mathbf{B}(n)]_{ij} = \beta_{ij}(n)$, i.e.,

$$\mathbf{B}(n) = \begin{bmatrix} \beta_{11}(n) & \beta_{12}(n) & \cdots & \beta_{1V}(n) \\ \beta_{21}(n) & \beta_{22}(n) & \cdots & \beta_{2V}(n) \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{V1}(n) & \beta_{V2}(n) & \cdots & \beta_{VV}(n) \end{bmatrix}, \quad (12)$$

which allows us to recast the NMSD as

$$\text{NMSD}(n) = \frac{1}{V} \text{Tr}\{\mathbf{B}(n)\}. \quad (13)$$

Furthermore, defining the $V^2 \times 1$ vectors

$$\boldsymbol{\beta}(n) \triangleq \text{vec}\{\mathbf{B}(n)\} = [\beta_{11}(n) \ \beta_{21}(n) \ \cdots \ \beta_{VV}(n)]^T \quad (14)$$

and $\mathbf{b} \triangleq \text{vec}\{\mathbf{I}_V\}$, and recalling that

$$\text{Tr}\{\mathbf{M}_1\mathbf{M}_2\} = \text{vec}\{\mathbf{M}_1\}^T \text{vec}\{\mathbf{M}_2\} \quad (15)$$

for any arbitrary matrices \mathbf{M}_1 and \mathbf{M}_2 of compatible dimensions, (13) can be written as

$$\text{NMSD}(n) = \frac{1}{V} \text{Tr}\{\mathbf{B}(n)\} = \frac{1}{V} \mathbf{b}^T \boldsymbol{\beta}(n). \quad (16)$$

Resuming our analysis, by subtracting both sides of (4a) from \mathbf{w}_o , and replacing (1) and (5) in the resulting equation, after some algebraic manipulations, we can write

$$\begin{aligned} \tilde{\boldsymbol{\psi}}_k(n) &= [\mathbf{I}_M - \mu_k \zeta_k(n) \mathbf{u}_k(n) \mathbf{u}_k^T(n)] \tilde{\mathbf{w}}_k(n-1) \\ &\quad - \mu \zeta_k(n) \mathbf{u}_k(n) v_k(n), \end{aligned} \quad (17)$$

where we have introduced

$$\tilde{\boldsymbol{\psi}}_k(n) \triangleq \mathbf{w}_o - \boldsymbol{\psi}_k(n). \quad (18)$$

On the other hand, from (4b), we observe that

$$\tilde{\mathbf{w}}_k(n) = \sum_{i \in \mathcal{N}_k} c_{ik} \tilde{\boldsymbol{\psi}}_i(n). \quad (19)$$

If we multiply both sides of (19) by $\tilde{\mathbf{w}}_k^T(n)$ from the left, and use (4b) again, we obtain after some algebra

$$\|\tilde{\mathbf{w}}_k(n)\|^2 = \sum_{i \in \mathcal{N}_k} \sum_{j \in \mathcal{N}_k} c_{ik} c_{jk} \tilde{\boldsymbol{\psi}}_j^T(n) \tilde{\boldsymbol{\psi}}_i(n). \quad (20)$$

Replacing (17) in (20), we obtain

$$\begin{aligned} \|\tilde{\mathbf{w}}_k(n)\|^2 &= \sum_{i \in \mathcal{N}_k} \sum_{j \in \mathcal{N}_k} c_{ik} c_{jk} \\ &\cdot \left\{ [\mathbf{I}_M - \mu_j \zeta_j(n) \mathbf{u}_j(n) \mathbf{u}_j^T(n)] \tilde{\mathbf{w}}_j(n-1) \right. \\ &\quad \left. - \mu_j \zeta_j(n) \mathbf{u}_j(n) v_j(n) \right\}^T \\ &\cdot \left\{ [\mathbf{I}_M - \mu_i \zeta_i(n) \mathbf{u}_i(n) \mathbf{u}_i^T(n)] \tilde{\mathbf{w}}_i(n-1) \right. \\ &\quad \left. - \mu_i \zeta_i(n) \mathbf{u}_i(n) v_i(n) \right\}. \end{aligned} \quad (21)$$

To examine the MSD of node k , we need to take the expectations from both sides of (21). At this point, we make a few assumptions to make the analysis more tractable, all of which are common in the related literature [1]–[3], [35]–[37]:

- A1.** All the nodes in the network employ the same step size, i.e., $\mu_1 = \cdots = \mu_V = \mu > 0$;
- A2.** The weight error vectors $\tilde{\mathbf{w}}_i(n-1)$ are statistically independent of $\mathbf{u}_j(n)$ for any pair i and j . This is a multi-agent version of the independence theory, a common assumption in the adaptive filtering literature [35], [36];
- A3.** The measurement noise $v_k(n)$ is zero-mean with variance $\sigma_{v_k}^2$, independent and identically distributed (iid), and independent from any other variable for $k = 1, \dots, V$;

A4. The input signals are zero-mean and white Gaussian with variance $\sigma_{u_1}^2 = \cdots = \sigma_{u_V}^2 = \sigma_u^2 > 0$. In other words, the autocorrelation matrices $\mathbf{R}_{u_k} \triangleq \text{E}\{\mathbf{u}_k(n) \mathbf{u}_k^T(n)\}$, $k = 1, \dots, V$ are the same, and are proportional to the identity matrix, i.e., $\mathbf{R}_{u_1} = \cdots = \mathbf{R}_{u_V} = \sigma_u^2 \mathbf{I}_M$;

A5. For every node k , $\zeta_k(n)$ is independent from any other variable, and drawn from a Bernoulli distribution, such that $\zeta_k(n) = 1$ with probability p_ζ and $\zeta_k(n) = 0$ with probability $1-p_\zeta$. We remark that we are assuming that p_ζ is the same for every node in the network. Furthermore, for any pair of distinct nodes, $\zeta_i(n)$ and $\zeta_j(n)$, $i \neq j$, are statistically independent from each other;

A6. At any time instant n , $u_i(n)$ is statistically independent from $u_j(n)$ for any pair of nodes i and j , $i \neq j$.

With these assumptions at hand, we can continue with our analysis. For the sake of brevity, we shall omit here the intermediate steps and focus on the main results obtained from (21). These results are justified in detail in Appendix A. For the scenario described, using (6), we can obtain

$$\begin{aligned} \beta_{kk}(n) &= \theta \sum_{i=1}^V c_{ik}^2 \beta_{ii}(n-1) \\ &\quad + \tau \sum_{j=1}^V \sum_{\substack{\ell=1 \\ \ell \neq j}}^V c_{jk} c_{\ell k} \beta_{j\ell}(n-1) + \mu^2 p_\zeta M \sigma_u^2 \sum_{q=1}^V c_{qk}^2 \sigma_{v_q}^2, \end{aligned} \quad (22)$$

where for the sake of compactness we have introduced

$$\theta \triangleq 1 - 2\mu p_\zeta \sigma_u^2 + \mu^2 p_\zeta \sigma_u^4 (M+2) \quad (23)$$

and

$$\tau \triangleq 1 - 2\mu p_\zeta \sigma_u^2 + \mu^2 p_\zeta^2 \sigma_u^4. \quad (24)$$

Hence, we can see that $\text{MSD}_k(n)$ depends on the MSD of its neighbors at the previous iteration, as well as on the trace of the cross correlation matrices between $\tilde{\mathbf{w}}_j(n-1)$ and $\tilde{\mathbf{w}}_\ell(n-1)$, i.e., $\beta_{j\ell}(n-1) = \text{E}\{\tilde{\mathbf{w}}_j^T(n-1) \tilde{\mathbf{w}}_\ell(n-1)\}$, for every pair of nodes j and ℓ in the neighborhood of node k . The impact of each of these terms on the behavior of node k depends on μ , σ_u^2 , c_{ik} for $i \in \mathcal{N}_k$ (and, therefore, on the network topology), and, in the case of the MSD's of neighboring nodes, on the filter length M . Finally, the noise variance in the neighborhood also influences directly the MSD of node k .

From (22), it becomes evident that we also need to study how the trace of the cross-correlation matrix of $\tilde{\mathbf{w}}_j(n)$ and $\tilde{\mathbf{w}}_\ell(n)$, with $j \neq \ell$, evolves over time. Again, we focus on the main result and leave the details for the Appendix A. Using assumptions **A1** to **A6**, we can obtain the following recursion:

$$\begin{aligned} \beta_{j\ell}(n) &= \theta \sum_{t=1}^V c_{tj} c_{t\ell} \beta_{tt}(n-1) \\ &\quad + \tau \sum_{r=1}^V \sum_{\substack{s=1 \\ s \neq r}}^V c_{rj} c_{s\ell} \beta_{rs}(n-1) + \mu^2 p_\zeta M \sigma_u^2 \sum_{z=1}^V c_{zj} c_{z\ell} \sigma_{v_z}^2. \end{aligned} \quad (25)$$

Next, we analyze in Sec. III-A the special case of the non-cooperative approach, as it is more straightforward and enables us to draw some qualitative conclusions about this

type of scenario. Then, in Sec. III-B, we resume our analysis of Eqs. (22) and (25) for the general case. Later on, in Sec. III-C, we derive an approximate model for the cooperative strategies that will provide us with valuable insights.

A. The Non-Cooperative Case

For the non-cooperative case, the analysis is straightforward. Since in this case we have $c_{kk} = 1$ for any $k = 1, \dots, V$, and $c_{ik} = 0$ if $i \neq k$, we can recast (22) as

$$\beta_{kk}(n) = \theta \beta_{kk}(n-1) + \mu^2 p_\zeta M \sigma_u^2 \sigma_{v_k}^2. \quad (26)$$

Assuming $\beta_{kk}(0) = \|\mathbf{w}_o\|^2$, by recursively applying (26) we get

$$\beta_{kk}(n) = \theta^n \|\mathbf{w}_o\|^2 + \mu^2 p_\zeta M \sigma_u^2 \sigma_{v_k}^2 \sum_{n_i=0}^{n-1} \theta^{n_i}. \quad (27)$$

Assuming that $|\theta| < 1$, we have that $\lim_{n \rightarrow \infty} \theta^n = 0$. At this point, it is worth noting that we can write

$$\tau = (1 - \mu p_\zeta \sigma_u^2)^2 \geq 0 \quad (28)$$

and

$$\theta = \tau + \mu^2 p_\zeta \sigma_u^4 (M + 2 - p_\zeta) \geq \tau, \quad (29)$$

where the equality only occurs if $p_\zeta = 0$, in which case $\theta = \tau = 1$. Assuming $p_\zeta > 0$, we notice that $\theta < 1$ if, and only if

$$0 < \mu < \frac{2}{(M+2)\sigma_u^2}, \quad (30)$$

where we have incorporated the fact that $\mu > 0$. In this case, we can write $\sum_{n_i=0}^{n-1} \theta^{n_i} = \frac{1-\theta^n}{1-\theta}$. Thus, considering (7) and (11), by applying some algebraic manipulations to (27), we can write the NMSD as

$$\boxed{\text{NMSD}_{\text{nc}}(n) = (\|\mathbf{w}_o\|^2 - \chi_{\text{nc}}) \theta^n + \chi_{\text{nc}}}, \quad (31)$$

with

$$\chi_{\text{nc}} \triangleq \frac{\mu M}{2 - \mu \sigma_u^2 (M + 2)} \cdot \frac{\sum_{k=1}^V \sigma_{v_k}^2}{V}. \quad (32)$$

Taking the limit $\lim_{n \rightarrow \infty} \text{NMSD}_{\text{nc}}(\infty)$ in (31) yields

$$\boxed{\text{NMSD}_{\text{nc}}(\infty) = \chi_{\text{nc}}}. \quad (33)$$

Therefore, we can clearly see that χ_{nc} given by (32) represents the steady-state value of the NMSD for the non-cooperative strategy. It is interesting to notice that p_ζ does not appear in (32). Thus, the sampling probability does not affect the steady-state NMSD of the algorithm in the non-cooperative approach whatsoever, so long as $p_\zeta > 0$. If $p_\zeta = 0$ were chosen, we would obtain $\theta = 1$, and, from (26), we would get $\beta_{kk}(n) = \|\mathbf{w}_o\|^2$ for every iteration n . This is reasonable, since in this case the nodes would never acquire any information on \mathbf{w}_o .

There are a couple more things to notice from the previous analysis. Firstly, we remark that (32) agrees with existing results in the literature for the steady-state NMSD of non-cooperative networks when all the nodes are sampled and employ sufficiently small step sizes [1]. Moreover, taking into

account that, for a single LMS filter, it is a well-known result that its MSD can be approximated by [35], [36]

$$\chi_{\text{LMS}} \triangleq \frac{\mu M \sigma_v^2}{2 - \mu \sigma_u^2 (M + 2)} \quad (34)$$

for sufficiently small step sizes, we see from (32) that the steady-state NMSD of the non-cooperative approach is simply the average of the MSD's of V individual LMS filters working isolated from one another, with each filter k subjected to a certain noise power $\sigma_{v_k}^2$. This conclusion makes sense, since there is no cooperation between the nodes. Lastly, we remark that (30) is simply the condition for the stability of an LMS filter in the mean [35], [36]. Therefore, we can interpret (30) as follows: so long as $p_\zeta > 0$, if we pick a step size μ that leads to the individual stability of each filter in the network, the network as a whole will be stable, regardless of the value of p_ζ .

Finally, we notice that in (31) the term $(\|\mathbf{w}_o\|^2 - \chi_{\text{nc}}) \theta^n$ decays exponentially along the iterations. Assuming that this term is dominant during the transient phase in comparison with χ_{nc} , we conclude that the closer θ is to unity, the slower the convergence rate. From (23), we can clearly see that

$$\lim_{p_\zeta \rightarrow 0^+} \theta = 1. \quad (35)$$

This indicates that the lower the sampling probability, the slower the convergence.

From the previous discussion, we can summarize the effects of sampling on the behavior of non-cooperative networks as in the following result.

Result 1 (Non-cooperative networks). *In the case of the non-cooperative networks, the stability of dLMS is ensured if (30) holds and $0 < p_\zeta \leq 1$. Under these conditions, the lower the sampling probability p_ζ , the slower the convergence rate. Moreover, the steady-state NMSD is completely unaffected by p_ζ . This result follows as a direct consequence of Eqs. (31)–(33) and (35).*

B. The General Case

Let us now resume our analysis for a more general case, encompassing the cooperative strategies as well. From (22) and (25), we observe that $\beta_{kk}(n)$ can be seen as a linear combination of the $\beta_{ii}(n-1)$ and $\beta_{j\ell}(n-1)$, plus the constant term $\mu^2 M \sigma_u^2 \sum_{q \in \mathcal{N}_k} c_{qk}^2 \sigma_{v_q}^2$. This becomes clear if we expand the first two summations in (22), i.e.,

$$\begin{aligned} \beta_{kk}(n) &= \theta c_{1k}^2 \beta_{11}(n-1) + \dots + \theta c_{V_k}^2 \beta_{V_k V_k}(n-1) \\ &+ \tau c_{1k} c_{2k} \beta_{12}(n-1) + \dots + \tau c_{V_k} c_{(V-1)k} \beta_{V(V-1)}(n-1) \\ &+ \mu^2 p_\zeta M \sigma_u^2 \sum_{q=1}^V c_{qk}^2 \sigma_{v_q}^2, \end{aligned} \quad (36)$$

Analogously, the same can be said about $\beta_{j\ell}(n)$ based on (25). Hence, we should be able to write

$$\beta(n) = \Phi \beta(n-1) + \mu^2 p_\zeta M \sigma_u^2 \sigma, \quad (37)$$

where Φ is a matrix whose k -th row determines how exactly each $\beta_{ij}(n-1)$ influences the corresponding term in

the current iteration, and σ is a vector that aggregates the information from the network topology and noise variance from the constant terms that appear in (22) and (25).

Let us now aggregate the combination weights into a $V \times V$ matrix \mathbf{C} , such that $[\mathbf{C}]_{ij} = c_{ij}$. Similarly, let us collect the noise variances in a $V \times V$ diagonal matrix \mathbf{R}_v , such that its k -th element is equal to $\sigma_{v_k}^2$, i.e.,

$$\mathbf{R}_v = \begin{bmatrix} \sigma_{v_1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{v_2}^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{v_V}^2 \end{bmatrix}. \quad (38)$$

In this case, we observe that

$$\mathbf{C}\mathbf{R}_v\mathbf{C}^T = \begin{bmatrix} \sum_{k=1}^V c_{k1}^2 \sigma_{v_k}^2 & \sum_{k=1}^V c_{k1}c_{k2} \sigma_{v_k}^2 & \cdots & \sum_{k=1}^V c_{k1}c_{kV} \sigma_{v_k}^2 \\ \sum_{k=1}^V c_{k2}c_{k1} \sigma_{v_k}^2 & \sum_{k=1}^V c_{k2}^2 \sigma_{v_k}^2 & \cdots & \sum_{k=1}^V c_{k2}c_{kV} \sigma_{v_k}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{k=1}^V c_{kV}c_{k1} \sigma_{v_k}^2 & \sum_{k=1}^V c_{kV}c_{k2} \sigma_{v_k}^2 & \cdots & \sum_{k=1}^V c_{kV}^2 \sigma_{v_k}^2 \end{bmatrix}. \quad (39)$$

Then, we may write the $V^2 \times 1$ vector σ in (37) as

$$\sigma = \text{vec}\{\mathbf{C}\mathbf{R}_v\mathbf{C}^T\} = \begin{bmatrix} \sum_{i=1}^V c_{i1}^2 \sigma_{v_i}^2 \\ \sum_{i=1}^V c_{i2}c_{i1} \sigma_{v_i}^2 \\ \vdots \\ \sum_{i=1}^V c_{iV}^2 \sigma_{v_i}^2 \end{bmatrix}. \quad (40)$$

As for the matrix Φ , from (22) and (25) we obtain that

$$\Phi = \Omega \odot \Gamma, \quad (41)$$

where we have introduced

$$\Gamma \triangleq (\mathbf{C} \otimes \mathbf{C})^T \quad (42)$$

and

$$\Omega = [\Omega_1 \ \Omega_2 \ \cdots \ \Omega_V], \quad (43)$$

in which Ω_i is a $V^2 \times V$ matrix whose elements in the i -th column are all equal to θ , and whose other elements are all equal to τ , i.e.

$$\Omega_i = \underbrace{\begin{bmatrix} \tau & \cdots & \tau & \theta & \tau & \cdots & \tau \\ \tau & \cdots & \tau & \theta & \tau & \cdots & \tau \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \tau & \cdots & \tau & \theta & \tau & \cdots & \tau \end{bmatrix}}_{V \text{ columns}}. \quad (44)$$

$\begin{matrix} i\text{-th column} \\ \downarrow \end{matrix}$

More details on the matrices Φ and Ω are discussed in Appendix B, but are skipped here for the sake of brevity.

With Eqs. (39) to (44) at hand, we can continue with the analysis of Eq. (37). Considering an initial condition $\beta_0 = \beta(0)$, the recursive application of (37) leads to

$$\beta(n) = \Phi^n \beta_0 + \mu^2 p_\zeta M \sigma_u^2 \sum_{n_i=0}^{n-1} \Phi^{n_i} \sigma. \quad (45)$$

If the algorithm is initialized with $\tilde{\mathbf{w}}_k(0) = \mathbf{0}_M$ for every node k , we have that $\tilde{\mathbf{w}}_k(0) = \mathbf{w}_o$. Thus, for any i and j , we have that $\beta_{ij}(0) = \mathbb{E}\{\tilde{\mathbf{w}}_i^T(0)\tilde{\mathbf{w}}_j(0)\} = \mathbb{E}\{\mathbf{w}_o^T \mathbf{w}_o\} = \|\mathbf{w}_o\|^2$, and, consequently,

$$\beta_0 = \|\mathbf{w}_o\|^2 \mathbf{1}_{V^2}. \quad (46)$$

Using this information in (45) and noting that $\sum_{n_i=0}^{n-1} \Phi^{n_i} = [\mathbf{I}_{V^2} - \Phi]^{-1}[\mathbf{I}_{V^2} - \Phi^n]$, we get

$$\beta(n) = \|\mathbf{w}_o\|^2 \Phi^n \mathbf{1}_{V^2} + \mu^2 p_\zeta M \sigma_u^2 [\mathbf{I}_{V^2} - \Phi]^{-1} [\mathbf{I}_{V^2} - \Phi^n] \sigma. \quad (47)$$

Thus, considering (47) and (16), we can write

$$\boxed{\text{NMMSD}(n) = \frac{1}{V} \left\{ \|\mathbf{w}_o\|^2 \mathbf{b}^T \Phi^n \mathbf{1}_{V^2} + \mu^2 p_\zeta M \sigma_u^2 \mathbf{b}^T [\mathbf{I}_{V^2} - \Phi]^{-1} [\mathbf{I}_{V^2} - \Phi^n] \sigma \right\}}. \quad (48)$$

It is worth noting that although \mathbf{w}_o appears in (47) and (48), we do not need to know it beforehand. Instead, we only need its norm. This is usually not a problem, since the norm of the optimal system can be adjusted by using adaptive gain control. Furthermore, we remark that $\beta_{ij}(n) = \beta_{ji}(n)$, which means that the vector $\beta(n)$ given by (14) has $V(V-1)/2$ duplicated entries. Although we could remove these elements from our model to make it more efficient from a computational perspective, and make the appropriate modifications where needed, we opted not to make this change for the clarity of the exposition. This is due to the fact that we are not primarily concerned with the computational complexity of the proposed model, and we believe that the formulation adopted is more convenient for the calculations. However, we would like to reinforce that this change is possible, and can reduce the amount of computations significantly, especially if V is large. Lastly, it is interesting to notice that we can obtain the theoretical MSD of the LMS algorithm [35]–[37] as a special case of (48). In this situation, we have that $V = 1$, and \mathbf{b} , σ , Γ , Ω and Φ degenerate into $\mathbf{b} = 1$, $\sigma = \sigma_v^2$, $\Gamma = 1$, $\Omega = \theta$ and $\Phi = \theta$, respectively. Replacing these results in (48), we get

$$\text{MSD}(n) = (\|\mathbf{w}_o\|^2 - \chi_{\text{LMS}}) \theta^n + \chi_{\text{LMS}},$$

with χ_{LMS} given by (34).

From (47) and (48), we can see that the stability of the network in the mean-squared sense is ensured if $\lim_{n \rightarrow \infty} \Phi^n = \mathbf{0}_{V^2 \times V^2}$, which occurs if, and only if, [44]

$$\rho(\Phi) < 1, \quad (49)$$

where $\rho(\cdot)$ denotes the spectral radius of a matrix, i.e., the maximum absolute value of its eigenvalues.

If (49) holds, by taking the limit of (48) as $n \rightarrow \infty$, we conclude that the steady-state NMSD is given by

$$\text{NMSD}(\infty) = \frac{\mu^2 p_\zeta M \sigma_u^2}{V} \mathbf{b}^T [\mathbf{I}_{V^2} - \Phi]^{-1} \sigma. \quad (50)$$

As will become clear in Sec. V, the model described by (48) can be very accurate. However, it still does not allow us to draw many qualitative conclusions about the behavior of the diffusion algorithm. Thus, further approximations may come in handy to aid us in this task, as shown next.

C. An Approximate Model for the Cooperative Strategies

Since the columns of the matrix Ω are filled by either τ or θ , we could adopt $\Omega \approx \tau \mathbf{1}_{V^2 \times V^2}$ or $\Omega \approx \theta \mathbf{1}_{V^2 \times V^2}$ as an approximation. Making these replacements in (41) leads to $\Phi \approx \tau \Gamma$ or $\Phi \approx \theta \Gamma$, respectively. Due to (29), the second approximation tends to be more conservative. Replacing it in (49), and using the fact that for any real scalar α and matrix \mathbf{M} , $\rho(\alpha \mathbf{M}) = |\alpha| \rho(\mathbf{M})$, we conclude that

$$\theta \cdot \rho(\Gamma) < 1, \quad (51)$$

where we used the fact that $\theta > 0$.

Due to (6), \mathbf{C} is a left-stochastic matrix, i.e., a matrix whose entries are all non-negative, and whose columns add up to one. Consequently, $\mathbf{C} \otimes \mathbf{C}$ is also a left-stochastic matrix. By transposing it, we conclude that Γ is a right-stochastic matrix, i.e., all of its entries are non-negative, and its rows add up to one. One interesting property of such matrices is that their spectral radius is always equal to one [44], [45]. Thus, the condition established by (51) can be recast as simply $\theta < 1$. Replacing (23) in (51) and assuming that $p_\zeta > 0$, after some algebra we get (30). Hence, we can observe that our previous conclusion that if μ lies within a certain range, the sampling probability p_ζ does not affect the stability of the algorithm, so long as $p_\zeta > 0$, holds for the general case, and not just for the non-cooperative approach. We remark that (30) corresponds to ensuring that each individual filter in the non-cooperative scheme is stable. It is a well-known fact in the adaptive diffusion networks literature that, if every individual node is stable, the stability of the network as a whole in a cooperative scenario is also ensured [1]–[3]. However, we remark that (30) was obtained considering a worst-case scenario, in which $\Phi = \theta \Gamma$. In practice, (30) is not strictly necessary to ensure the stability of the algorithm if a cooperative strategy is adopted. In these cases, greater step sizes may be employed without making the algorithm unstable.

Next, we seek to derive an approximate model from (48) and (50) for the cooperative strategies. In this case, simulations show that adopting $\Phi \approx \tau \Gamma$ leads to reasonable results. It is worth noting, however, that this approximation is not applicable to the non-cooperative case. Thus, replacing Φ with $\tau \Gamma$ in (48), we conclude that, for cooperative networks, we may write

$$\text{NMSD}(n) = \frac{1}{V} \left\{ \|\mathbf{w}_o\|^2 \mathbf{b}^T \tau^n \Gamma^n \mathbf{1}_{V^2} + \mu^2 p_\zeta M \sigma_u^2 \mathbf{b}^T [\mathbf{I}_{V^2} - \Phi]^{-1} [\mathbf{I}_{V^2} - \tau^n \Gamma^n] \sigma \right\}. \quad (52)$$

We remark that the result of the multiplication $\Gamma^n \mathbf{1}_{V^2}$ is a column vector whose i -th element is the sum of the elements of the i -th row of the matrix Γ^n . Since the product of right-stochastic matrices is also right-stochastic [45], Γ^n is right-stochastic, from which we notice that $\Gamma^n \mathbf{1}_{V^2} = \mathbf{1}_{V^2}$. Moreover, using the fact that $\mathbf{b}^T \mathbf{1}_{V^2} = V$, we thus conclude that for the cooperative strategies, the NMSD is well approximated by

$$\text{NMSD}_\tau(n) = \|\mathbf{w}_o\|^2 \tau^n + \frac{\mu^2 p_\zeta M \sigma_u^2}{V} \cdot \mathbf{b}^T [\mathbf{I}_{V^2} - \tau \Gamma]^{-1} [\mathbf{I}_{V^2} - \tau^n \Gamma^n] \sigma. \quad (53)$$

Analogously to what we observed in Sec. III-A about (31), the first term in (53) decays exponentially along the iterations. Assuming that this term is dominant during the transient phase, we conclude that the closer that τ is to unity, the slower the convergence rate. From (23) and (24), we note that

$$\lim_{p_\zeta \rightarrow 0^+} \tau = 1. \quad (54)$$

Hence, much like in the non-cooperative case of Sec. III-A, the lower the sampling probability, the slower the convergence for the cooperative strategies. We remark that this result is in accordance with Fig. 1.

Finally, let us assume that $\tau < 1$, which can only occur if $p_\zeta > 0$. In this case, if we take the limit when $n \rightarrow \infty$ in (53), we conclude that the steady-state NMSD of the cooperative networks can be well approximated by

$$\text{NMSD}_\tau(\infty) = \frac{\mu^2 p_\zeta M \sigma_u^2}{V} \mathbf{b}^T [\mathbf{I}_{V^2} - \tau \Gamma]^{-1} \sigma. \quad (55)$$

Eq. (55) clearly depends on the matrix Γ , and, therefore, on the network topology and combination rule adopted. It also depends on μ , M , σ_u^2 , V , and p_ζ , as evidenced by the factor $\frac{\mu^2 p_\zeta M \sigma_u^2}{V}$ that appears in (55). However, μ , M , σ_u^2 , and p_ζ also influence the value of τ , which in its turn plays a role in Eq. (55) due to the matrix $[\mathbf{I}_{V^2} - \tau \Gamma]^{-1}$. Unfortunately, there does not seem to be a way of rewriting (55) such that the dependency of the steady-state NMSD with relation to p_ζ is completely explicit. Thus, it is hard to tell exactly how the sampling probability affects the steady-state performance in the generic case by just taking glance at Eq. (55). However, as we shall see in the simulations of Sec. V, the model of Eq. (55) accurately predicts the effects of the sampling probability on the performance of cooperative networks.

Even though it is difficult to extract direct conclusions from (55) for any arbitrary topology without calculating them explicitly, we can use it as a starting point for obtaining an upper bound for the steady-state NMSD in the generic case. Doing so, we do get an explicit relationship with the sampling probability p_ζ . If the matrix Γ is symmetric, which certainly occurs if Metropolis weights are employed, for example, one can obtain the upper bound given by (56), depicted at the top of the next page. This result is derived in Appendix C. We notice from (56) that the upper bound for the steady-state NMSD decreases as we reduce p_ζ .

Furthermore, let us also calculate the steady-state NMSD for a particular case. Next, we consider a network topology

$$\text{NMSD}_{\tau}(\infty) \leq \frac{\mu M}{2 - \mu p_{\zeta} \sigma_u^2} \cdot \left\{ \frac{1}{2V} \left[\sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 + \sqrt{V \sum_{i=1}^V \sum_{j=1}^V \left[\left(\sum_{k=1}^V c_{ki} c_{kj} \sigma_{v_k}^2 \right)^2 \right]} \right] \right\}. \quad (56)$$

represented by a complete graph, i.e., one in which every pair of nodes is directly connected by an edge. A graph with this topology and V nodes is usually denoted by K_V in the literature. An example with $V = 8$ nodes is depicted in Fig. 2.

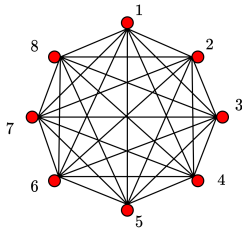


Fig. 2: A network arranged according to the K_8 topology.

For the K_V topology, the Uniform and Metropolis weights coincide, and lead to $\mathbf{C}_{K_V} = \frac{1}{V} \mathbf{1}_{V \times V}$ and $\mathbf{\Gamma}_{K_V} = \frac{1}{V^2} \mathbf{1}_{V^2 \times V^2}$, where the index K_V is adopted to refer to this type of topology with Uniform or Metropolis weights. For the aforementioned matrix $\mathbf{\Gamma}_{K_V}$, from (55), we get

$$\text{NMSD}_{\tau_{K_V}}(\infty) = \chi_{K_V} \triangleq \frac{\mu M}{2 - \mu p_{\zeta} \sigma_u^2} \frac{\sum_{k=1}^V \sigma_{v_k}^2}{V^2}. \quad (57)$$

This result is derived in Appendix D and establishes an explicit relationship between the steady-state NMSD and p_{ζ} in the case of the network topology given by Fig. 2. Comparing (32) and (33) with (56) and (57), we observe that the sampling probability does not affect the steady-state performance of the non-cooperative strategy, but does influence the steady-state NMSD of the cooperative schemes. For example, we get from (57) that

$$\frac{\mu M}{2} \frac{\sum_{k=1}^V \sigma_{v_k}^2}{V^2} < \chi_{K_V} \leq \frac{\mu M}{2 - \mu \sigma_u^2} \frac{\sum_{k=1}^V \sigma_{v_k}^2}{V^2}, \quad (58)$$

i.e., as we reduce the value of p_{ζ} , χ_{K_V} decreases as well. The first inequality is obtained by taking the limit of χ_{K_V} as $p_{\zeta} \rightarrow 0^+$. This observation is in accordance with the results from Fig. 1. This reduction in the steady-state NMSD should be more significant for relatively large values of μ and σ_u^2 . If $\mu \sigma_u^2 \ll 2$, however, the impact of sampling becomes negligible. We remark that (32) and (57) agree with the analysis of [1]–[3] for the ATC dLMS algorithm with every node sampled and small step sizes, if we consider $p_{\zeta} = 1$ in the latter. Lastly, it is worth noting that (57) only holds for $p_{\zeta} > 0$. This is because, in order to obtain this result, we assumed in our calculations that $\tau < 1$, which can only be true if $p_{\zeta} > 0$, as can be seen from (24). If we consider $p_{\zeta} = 0$, we would get $\tau = 1$ and therefore obtain from (55) that $\text{NMSD}_{\tau}(n) = \|\mathbf{w}_o\|^2$ for every iteration n , which is in accordance with our expectations.

We can summarize the results of the analysis for the cooperative networks as in the following result.

Result 2 (Cooperative networks). *In the case of the cooperative networks, the stability of dLMS is ensured in the mean-squared sense if (49) holds, which can only occur if $p_{\zeta} > 0$. If the sampling probability is different from zero, (30) is a sufficient but not strictly necessary condition for stability of cooperative networks. Furthermore, if the algorithm is stable, then the lower the sampling probability p_{ζ} , the slower the convergence rate, much like in the non-cooperative case. However, in contrast with the non-cooperative approach, in cooperative schemes the steady-state NMSD decreases as we reduce p_{ζ} . This follows as a consequence of Eqs. (49) and (53)–(55), and is better visualized from the approximate model given by (57).*

Comparing Results 1 and 2, therefore, we can see that in both the non-cooperative and cooperative schemes, as we reduce p_{ζ} , the convergence rate deteriorates in comparison with the case in which every node is sampled, assuming that the other parameters are kept fixed. However, in the non-cooperative case, this deterioration is not accompanied by any change in the steady-state NMSD whatsoever if the other parameters of the algorithm remain unchanged. Under the same circumstances, when the nodes do cooperate, the steady-state NMSD does slightly decrease, although, as we shall discuss next, it would not be adequate to treat this as a performance improvement in comparison with the case in which the nodes are kept permanently sampled. It is worth recalling that (55) is not applicable to non-cooperative networks, due to the approximations adopted in its derivation, i.e., that $\mathbf{\Phi} \approx \tau \mathbf{\Gamma}$. One possible interpretation for the different impacts of the sampling probability in both cases is as follows. The adaptation step is the process through which the algorithm acquires knowledge about its environment. For this reason, it is particularly relevant when there is little knowledge about the optimal system – e.g., during the transient phase. Thus, it makes sense that by not sampling the nodes in the transient phase, the convergence rate should deteriorate. However, the adaptation step also introduces noise into the algorithm, since it involves the acquisition of the desired signal, which is corrupted by it. In steady state, the algorithm does not gain enough information from the adaptation step to continue to decrease its NMSD, but is affected by the noise that it injects. The step size directly influences the impact of the measurement noise on the algorithms, since it multiplies $d_k(n)$, and, therefore, $v_k(n)$ in (4a). Thus, the greater the μ , the more the noise will affect the behavior of the algorithm. Conversely, if μ is small, this effect is restricted. Following this line of reasoning, if we cease to sample some of the nodes, there is less noise entering the algorithm. In a non-cooperative scheme,

if a node is not sampled, its estimate remains fixed until its sampling is resumed. Hence, there is no reason why the steady-state performance should be affected by sampling less nodes, which is predicted by our theoretical model. However, in the cooperative schemes, this is changed by the existence of the combination step. Indeed, (57) shows that we should expect some decrease in the steady-state NMSD in this scenario, even if slightly, and at the expense of a great deterioration in the convergence rate. The theoretical model attributes this to the parameter τ , which is related to the cooperation between nodes in (22), since it determines how $E\{\tilde{\mathbf{w}}_\ell^T(n-1)\tilde{\mathbf{w}}_j(n-1)\}$, for ℓ and j in the neighborhood of node k , will affect $E\{\|\tilde{\mathbf{w}}_k(n)\|^2\}$. The model also shows that when μ is large, the effects of sampling less nodes on the steady-state NMSD should be more noticeable, which supports the idea of the step size as a factor that determines the impact of the measurement noise in the algorithm. Interestingly, the diagnosis that the adaptation step injects noise in the algorithm, and that the combination step tends to remove it, has been raised in, e.g., [26], [28], [46], but lacked formal theoretical support, until now.

Lastly, it may be interesting to compare the steady-state NMSD achieved by the algorithm with a certain step size μ and sampling probability p_ζ , and the one obtained by utilizing $\mu' = \mu p_\zeta$ and maintaining all nodes sampled. Denoting these quantities by $\text{NMSD}_{\text{async.}(\infty)}$ and $\text{NMSD}_{\text{sync.}(\infty)}$, respectively, we notice from (57) that, for the topology of Fig. 2, we get

$$\text{NMSD}_{\text{sync.}(\infty)} = p_\zeta \text{NMSD}_{\text{async.}(\infty)}. \quad (59)$$

This is in accordance with [34], in which it was noticed that the steady-state NMSD of the synchronous networks should be less than or equal to that of the asynchronous one, if an adjusted step size is adopted taking p_ζ into consideration. In other words, if we adjust the step size, the network with all nodes sampled should outperform the one with random sampling, as it will present approximately the same convergence rate, with a lower steady-state NMSD.

IV. COMPUTATIONAL COST REDUCTION

In this section, we seek to estimate the effects of sampling on the expected computational cost of the dLMS algorithm. For brevity, we focus on the number of multiplications per iteration, but a similar analysis could be done for the additions.

We begin by noticing that each sampled node k needs to perform $M(2 + |\mathcal{N}_k|) + 1$ multiplications per iteration, assuming that static combination weights are employed. This is summarized in Table II, in which the number of multiplications required at each node k per iteration is denoted by \otimes_k and detailed for each calculation required. We can see that $2M + 1$ multiplications are related to the adaptation step. Assuming that none of the operations associated with this step have to be performed at node k if it is not sampled, we conclude that the total number of multiplications required at the iteration n and node k with random sampling can be estimated as

$$\otimes_k(n) = \zeta_k(n) \cdot (2M + 1) + M|\mathcal{N}_k|. \quad (60)$$

Taking the expectations from both sides in (60), we obtain

$$E\{\otimes_k\} = p_\zeta(2M + 1) + M|\mathcal{N}_k|, \quad (61)$$

where we dropped the indication of the time instant n since the right-hand side of (61) remains constant along the iterations.

TABLE II: Estimated number of multiplications per iteration at each sampled node k .

| | Calculation | Step | \otimes_k |
|-------|--|--------|------------------------------|
| 1 | $y_k(n) = \mathbf{u}_k^T \mathbf{w}_k(n-1)$ | Adapt. | M |
| 2 | $e_k(n) = d_k(n) - y_k(n)$ | Adapt. | 0 |
| 3 | $\mu \cdot e_k(n)$ | Adapt. | 1 |
| 4 | $[\mu \cdot e_k(n)] \cdot \mathbf{u}_k(n)$ | Adapt. | M |
| 5 | $\mathbf{w}_k(n-1) + \{[\mu \cdot e_k(n)] \cdot \mathbf{u}_k(n)\}$ | Adapt. | 0 |
| 6 | $\sum_{i \in \mathcal{N}_k} c_{ik} \psi_i(n)$ | Comb. | $M \mathcal{N}_k $ |
| Total | | | $M(2 + \mathcal{N}_k) + 1$ |

Summing (61) for $k = 1, \dots, V$, we can estimate the expected computational cost for the whole network as

$$E\{\otimes_{\text{total}}\} = V p_\zeta(2M + 1) + M \sum_{k=1}^V |\mathcal{N}_k|. \quad (62)$$

By replacing $p_\zeta = 1$ in (62), we obtain the number of multiplications required by the dLMS with every node sampled. Thus, denoting the number of multiplications saved per iteration due to the sampling by $\Delta \otimes_{\text{total}}$, we can thus estimate it as

$$E\{\Delta \otimes_{\text{total}}\} = V(2M + 1)(1 - p_\zeta) \quad (63)$$

by subtracting (62) from the case with $p_\zeta = 1$.

From (63), we see that the smaller the p_ζ , the greater the savings in computation, as expected. Moreover, for a given p_ζ , the number of multiplications saved increases with V and M .

V. SIMULATION RESULTS

In this section, we present simulation results to validate the theoretical analysis. They were obtained over an average of 1000 independent realizations, considering the scenarios summarized in Table III. The Scenario 1 of this table corresponds to that of Fig. 1. In every case, the coefficients of the optimal system \mathbf{w}_o are drawn from a uniform distribution in the range $[-1, 1]$, and later normalized so that \mathbf{w}_o has unit norm. Moreover, the length of the adaptive filter is always equal to that of \mathbf{w}_o . We consider the network topology presented in Fig. 3(a), which was generated randomly. The input signal $u_k(n)$ and the measurement noise $v_k(n)$ follow Gaussian distributions with zero mean for each node k , with $\sigma_{u_k}^2 = \sigma_u^2 = 1$, whereas the noise variance $\sigma_{v_k}^2$ is drawn from a uniform distribution in the range $[0.001, 0.01]$ for $k = 1, \dots, V$, as depicted in Fig. 3(b).

TABLE III: List of scenarios considered in the simulations.

| Scenario | μ | M | Combination Rule |
|------------|-------|-----|------------------|
| Scenario 1 | 0.1 | 10 | Metropolis |
| Scenario 2 | 0.01 | 10 | Metropolis |
| Scenario 3 | 0.02 | 100 | Uniform |
| Scenario 4 | 0.01 | 10 | Non-Cooperative |

This section is organized as follows. In Sec. V-A, we study the transient performance of the algorithm, in Sec. V-B, its stability, and in Sec. V-C, its steady-state NMSD.

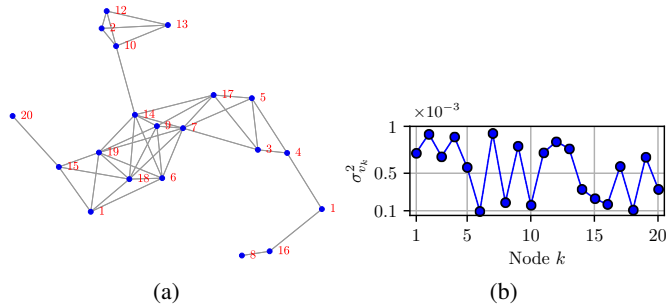


Fig. 3: (a) Network topology, and (b) noise variance profile considered in the simulations.

A. Transient Performance

In Fig. 4, we present the simulation results obtained in the Scenarios 1, 2, and 3 of Table III, considering $p_\zeta \in \{1, 0.5, 0.1\}$, and compare them to the theoretical models of Sec. III. The sub-figures in the top row present a comparison with the more precise model of Eq. (48), whereas those in the bottom row show the comparison with the approximate model of Eq. (53). Furthermore, each column of Fig. 4 refers to a scenario, with Figs. 4 (a) and (b) presenting the results obtained in Scenario 1, Figs. 4 (c) and (d) those from Scenario 2, and Figs. 4 (e) and (f) those from Scenario 3.

From Figs. 4 (a), (c), and (e), we can see that the simulation results match the theoretical curves very well for all three scenarios, and that the model accurately predicts the reduction in steady-state NMSD caused by the sampling of less nodes. Furthermore, comparing the results of Figs. 4 (c) and (e) with those of Fig. 4 (a), we can observe that the difference in performance caused by the sampling is indeed more noticeable for larger step sizes, as expected. By comparing Figs. 4 (a) and 4 (b) can see that, in Scenario 1, the approximate model is less accurate than the one described by Eq. (48). This was expected, to a certain extent. The same can be said about Scenario 3, by comparing Figs. 4 (e) and (f). However, we observe from Figs. 4 (c) and (d) that, in Scenario 2, both models practically coincide. Hence, we can conclude that the approximate model of Eq. (53) tends to be more accurate for relatively low step sizes μ and filter lengths M , and is more affected by them than the model of Eq. (48).

In order to examine the impact of sampling on the computational cost under these circumstances, in Table IV we present the average number of multiplications required per iteration in the whole network for each p_ζ considered in the simulations for $M = 10$, as in Scenarios 1 and 2, and for $M = 100$, as in Scenario 3. We also present the number of multiplications saved per iteration in comparison with the case in which every node is sampled, and compare them to the results given by (63). We can see that the average number of operations saved per iterations matches Eqs. (63), which can be attributed to the high number of realizations and iterations considered in the computations. Furthermore, it is straightforward to see that the smaller the p_ζ , the greater the savings in terms of computation, as expected. Moreover, for a fixed value of p_ζ , the computational cost and the savings increase with M , as

expected. From these experiments, we can summarize the effects of sampling as follows: smaller sampling probabilities p_ζ lead to lower steady-state NMSD and computational costs, at the expense of a deteriorated convergence rate.

TABLE IV: Average number of multiplications per iteration in the network for $p_\zeta \in \{0.1, 0.5, 1\}$ with $M = 10$ and $M = 100$.

| p_ζ | \otimes_{total} | | $\Delta \otimes_{\text{total}}$ | | Eq. (63) | |
|-----------|--------------------------|-----------|---------------------------------|-----------|----------|-----------|
| | $M = 10$ | $M = 100$ | $M = 10$ | $M = 100$ | $M = 10$ | $M = 100$ |
| 1 | ≈ 1460 | 14420 | 0 | 0 | 0 | 0 |
| 0.5 | 1250 | 12410 | 210 | 2010 | 210 | 2010 |
| 0.1 | 1082 | 10802 | 378 | 3618 | 378 | 3618 |

In the simulations of Fig. 5, we consider the Scenario 4 of Table III, and compare the simulation results to the theoretical model given by Eq. (31) for the non-cooperative scheme. Once again, the simulation results closely match the theoretical analysis. Furthermore, the simulations support the idea that the sampling probability does not affect the steady-state performance of the algorithm in the non-cooperative scheme, unlike what was observed in Fig. 4 for the cooperative rules.

B. Effects of Sampling on the Stability

From (30), we concluded that the sampling probability does not affect the stability of the algorithm so long as μ is sufficiently small and $p_\zeta > 0$. However, (30) is not strictly necessary to ensure the stability of the algorithm in the mean-squared sense. For instance, the value of μ considered in the Scenario 3 of Table III does not satisfy (30), but still leads to the stability of the dLMS algorithm for the network of Fig. 3(a) with Uniform weights and $M = 100$. Under these conditions, one obtains $\rho(\Phi) \approx 0.9628$ for $p_\zeta = 1$, which satisfies (49) and thus ensures the convergence in the mean-squared sense. For $p_\zeta = 0.5$ and $p_\zeta = 0.1$, we get $\rho(\Phi) < 1$ as well.

To verify if the sampling of the nodes influences the stability of the algorithm in the general case, we calculated the spectral radius of the matrix Φ considering $M = 100$ and the three combination policies for the network of Fig. 3(a) with $\mu = 0.1$ and several values of p_ζ . The results are shown in Fig. 6 (a), where we have highlighted with a red horizontal line the threshold $\rho(\Phi) = 1$. We can see that, for all combination policies, the adoption of $p_\zeta = 0$ leads to $\rho(\Phi) = 1$. This is expected, since in this case we get $\theta = \tau = 1$, and consequently $\Phi = \Gamma$, whose spectral radius is equal to one. Intuitively, this comes from the fact that the algorithm never acquires any knowledge on the optimal system if the nodes are never sampled. For the non-cooperative strategy, $\rho(\Phi)$ increases with p_ζ , indicating that the algorithm is unstable for any sampling probability. For the Uniform and Metropolis combination policies, however, $\rho(\Phi)$ decreases up to a certain point with the increase of p_ζ , and then begins to rise. Interestingly, for both policies, Fig. 6 (a) tells us that, under the conditions considered, the algorithm is unstable with all nodes sampled, but we can stabilize it by sampling less nodes. For the Uniform rule, we conclude from Fig. 6 (a) that the dLMS algorithm is stable for $p_\zeta \in]0, 0.71]$ (approximately), whereas for the Metropolis rule the stability occurs for $p_\zeta \in]0, 0.39]$. In order to verify these results, we ran the dLMS algorithm under the same

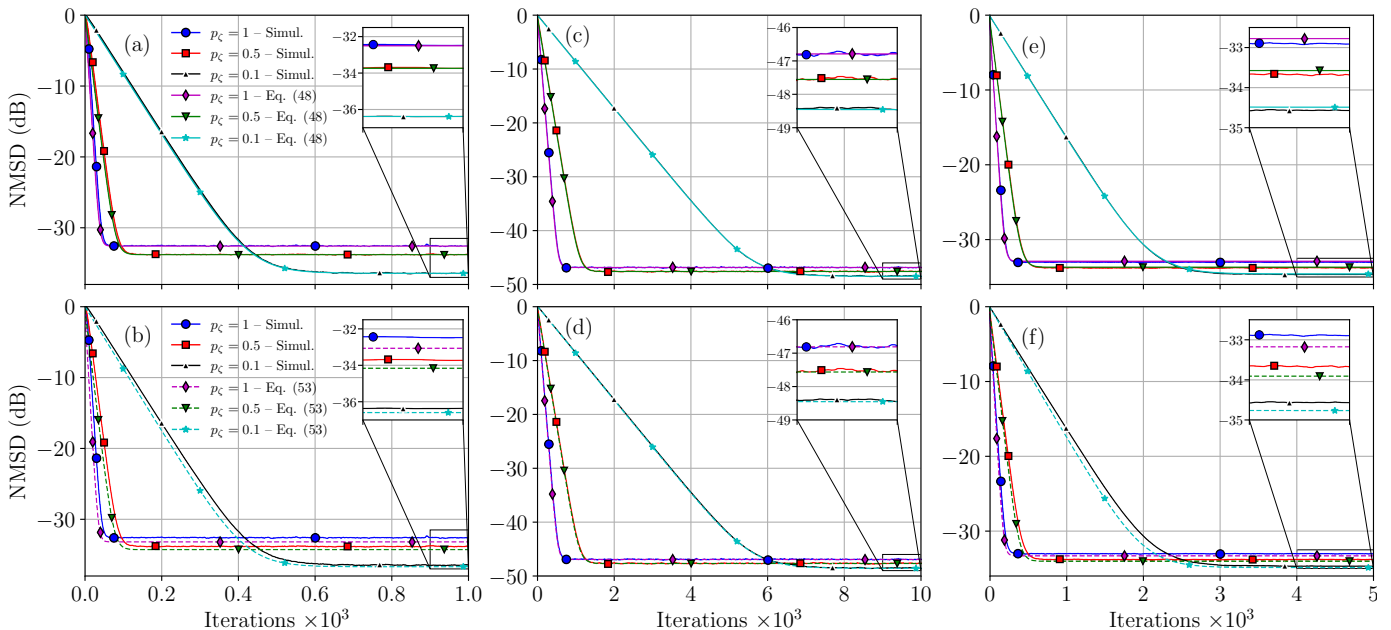


Fig. 4: The sub-figures in the top row present a comparison between the simulation results and the model of Eq. (48), whereas the ones in the bottom row show a comparison with the model of Eq. (53). The simulation results were obtained considering the Scenario 1 of Table III in sub-figures (a) and (b), Scenario 2 in (c) and (d), and Scenario 3 in (e) and (f).

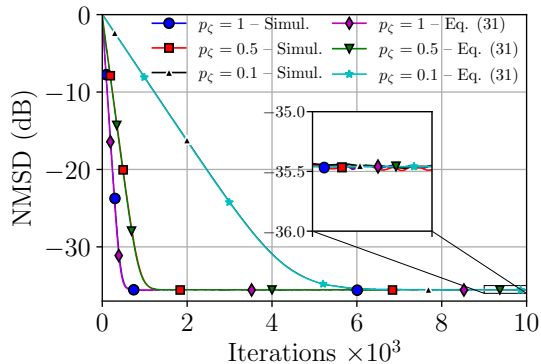


Fig. 5: Comparison between the simulation results and the model of Eq. (31) for $p_C \in \{1, 0.5, 0.1\}$ in Scenario 4.

circumstances considered in Fig. 6 (a) with different sampling probabilities in the range $[0.01, 1]$ for $200 \cdot 10^3$ iterations, which is more than necessary for the algorithm to achieve the steady state with $p_C = 0.01$ and cooperative strategies. Then, utilizing the `isnan` and `isinf` functions of MATLAB®, we calculated the percentage of realizations in which the dLMS algorithm diverged at some iteration. The results are depicted in Fig. 6 (b). We can see that, for the non-cooperative strategy, the algorithm diverges in 100% of the realizations for all values of p_C considered. For the Uniform and Metropolis rules, the percentage of realizations in which the algorithm diverges is initially zero, and increases steeply as p_C approaches the limit values of $p_C = 0.71$ and $p_C = 0.39$, respectively. For the former combination policy, the algorithm starts to diverge for $p_C > 0.68$, whereas for the latter the first divergences occur for $p_C > 0.41$. In both cases, by increasing p_C slightly further, the algorithm begins to diverge in 100% of

the realizations. Therefore, the simulation results of Fig. 6 (b) support the theoretical findings of Fig. 6 (a). It is worth noting that, although the Uniform rule leads to the stability of the algorithm for a wider range of p_C than the Metropolis rule in this case, this does not necessarily occur in all scenarios. For example, for the topology in Fig. 2, the weights coincide for the two rules and therefore there is no difference between them in terms of the stability of the algorithm.

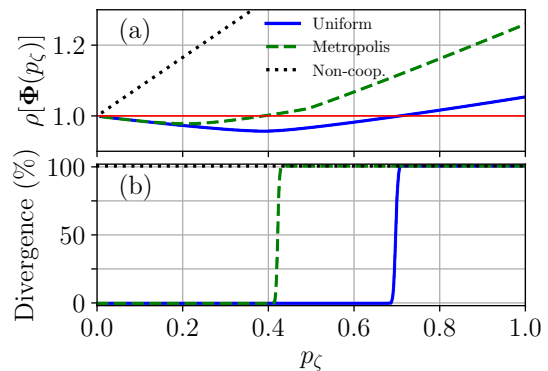


Fig. 6: (a) $\rho(\Phi)$ as a function of p_C , and (b) percentage of realizations in which the dLMS diverged with $\mu = 0.1$ and $M = 100$ for $p_C \in [0.01, 1]$ with different combination policies.

C. Steady-State Performance

Lastly, in order to verify Eqs. (50) and (55) in detail, we ran the ATC dLMS for different values of $p_C \in [0.1, 1]$, and calculated the average NMSD during the final 20% iterations of each realization. The results are shown in Figs. 7 (a), (b), and (c) for Scenarios 1, 2, and 3, respectively. In each

case, we set the total number of iterations N so that the algorithms achieved the steady state before the end of each realization, resulting in $10^3 \leq N \leq 10^5$. We remark that the simulation results match (50) very closely in Scenarios 1 and 2. In Scenario 3, there is a discrepancy between the simulation results and the theoretical curve of 0.1 dB, on average. As for the model of Eq. (55), there is a difference of approximately 0.40 dB in comparison with the simulation results in Fig. 7 (a), on average. In Fig. 7(c), this difference is of roughly 0.23 dB, whereas in Fig. 7 (b) the curve practically overlaps with the simulation results.

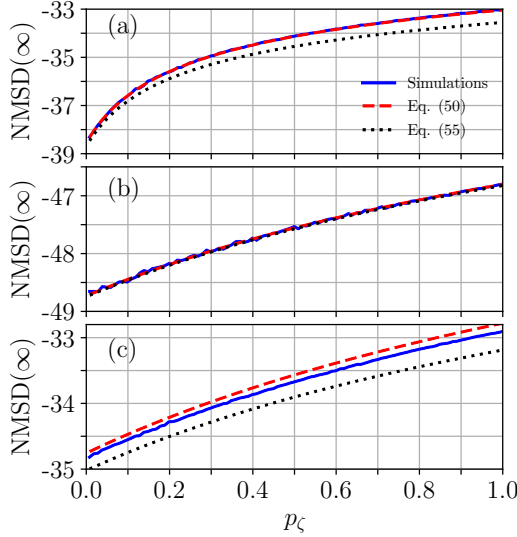


Fig. 7: Steady-state NMSD for $p_c \in [0.01, 1]$ in: (a) Scenario 1, (b) Scenario 2, and (c) Scenario 3.

VI. CONCLUSIONS

In this paper, we conducted a theoretical analysis on the ATC dLMS algorithm with random sampling in a stationary environment. It was shown that, so long as the sampling probability is greater than zero, the sampling does not hinder the stability of the algorithm in the worst-case scenario. Moreover, there is a slight reduction in the steady-state NMSD, but this is accompanied by a noticeable deterioration in the convergence rate. As a result, networks with a random sampling technique are outperformed by their counterparts in which the nodes are permanently sampled, if the step sizes are adjusted so as to obtain the same convergence rate. Lastly, as one might expect, the sampling of less nodes leads to a reduction in the computational cost. The simulations support the theoretical results obtained in the paper. Furthermore, it is interesting to notice that the effects of the sampling probability observed in them are not exclusive to the scenarios considered in this paper, as they were also noted in, e.g., [26], [28]. In future works, we intend to extend the analysis to: i) non-Gaussian and/or colored input signals, ii) to scenarios in which each node k has its own distinct step size μ_k , sampling probability p_{ζ_k} , and autocorrelation matrix \mathbf{R}_{u_k} for the input vector $\mathbf{u}_k(n)$, iii) to situations in which the optimal system varies over time, such as in random-walk scenarios, and iv)

possibly to other solutions, such as the dNLMS and dRLS algorithms. Finally, it may be interesting to investigate if this phenomenon also occurs in other types of algorithms, such as nonlinear solutions like the diffusion Kernel Least-Mean-Squares (dKLMS) algorithm [20]–[23].

APPENDIX A DERIVING (22) AND (25)

Taking the expectations from both sides of (21) and using Assumption **A1**, we get

$$E\{\|\tilde{\mathbf{w}}_k(n)\|^2\} = \beta_{kk}(n) = \sum_{i \in \mathcal{N}_k} \sum_{j \in \mathcal{N}_k} c_{ik} c_{jk} x_{ji}(n), \quad (64)$$

where we have defined

$$\begin{aligned} x_{ji}(n) \triangleq E\left\{ \left[\mathbf{I}_M - \mu \zeta_j(n) \mathbf{u}_j(n) \mathbf{u}_j^T(n) \right] \tilde{\mathbf{w}}_j(n-1) \right. \\ \left. - \mu \zeta_j(n) \mathbf{u}_j(n) v_j(n) \right\}^T \\ \cdot \left\{ \left[\mathbf{I}_M - \mu \zeta_i(n) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \right] \tilde{\mathbf{w}}_i(n-1) \right. \\ \left. - \mu \zeta_i(n) \mathbf{u}_i(n) v_i(n) \right\}. \end{aligned} \quad (65)$$

The analysis of (65) can be broken down into two cases: i) when $j = i$, and ii) when $j \neq i$. In the first situation, using **A3** and **A5**, and noting that $E\{\zeta_i(n)\} = E\{\zeta_i^2(n)\} = p_\zeta$, we get

$$\begin{aligned} x_{ii}(n) = E\{ \tilde{\mathbf{w}}_i^T(n-1) \tilde{\mathbf{w}}_i(n-1) \} \\ - 2\mu p_\zeta E\{ \tilde{\mathbf{w}}_i^T(n-1) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \tilde{\mathbf{w}}_i(n-1) \} \\ + \mu^2 p_\zeta E\{ \tilde{\mathbf{w}}_i^T(n-1) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \tilde{\mathbf{w}}_i(n-1) \} \\ + \mu^2 p_\zeta \sigma_{v_i}^2 E\{ \mathbf{u}_i^T(n) \mathbf{u}_i(n) \}. \end{aligned} \quad (66)$$

Using Assumptions **A2** and **A4**, and following similar procedures to those used in the analysis of the MSD of the LMS algorithm, we may write (see pages 803–807 of [37])

$$E\{ \tilde{\mathbf{w}}_i^T(n-1) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \tilde{\mathbf{w}}_i(n-1) \} = \sigma_u^2 \beta_{ii}(n-1) \quad (67)$$

and

$$E\{ \tilde{\mathbf{w}}_i^T(n-1) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \mathbf{u}_i(n) \mathbf{u}_i^T(n) \tilde{\mathbf{w}}_i(n-1) \} = \sigma_u^4 (M+2) \beta_{ii}(n-1). \quad (68)$$

Thus, (66) can be recast as

$$x_{ii}(n) = \theta \beta_{ii}(n-1) + \mu^2 p_\zeta M \sigma_u^2 \sigma_{v_i}^2, \quad (69)$$

with θ defined as in (23). Let us now analyze the case in which $j \neq i$. To make this distinction clearer, we shall replace the index i by ℓ in this case. From **A5**, we can observe that $E\{\zeta_j(n) \zeta_\ell(n)\} = E\{\zeta_j(n)\} E\{\zeta_\ell(n)\} = p_\zeta^2$. Thus, using assumptions **A3** and **A5**, we can rewrite (65) for $\ell \neq j$ as

$$\begin{aligned} x_{j\ell}(n) = E\{ \tilde{\mathbf{w}}_j^T(n-1) \tilde{\mathbf{w}}_\ell(n-1) \} \\ - \mu p_\zeta E\{ \tilde{\mathbf{w}}_j^T(n-1) \mathbf{u}_j(n) \mathbf{u}_j^T(n) \tilde{\mathbf{w}}_\ell(n-1) \} \\ - \mu p_\zeta E\{ \tilde{\mathbf{w}}_j^T(n-1) \mathbf{u}_\ell(n) \mathbf{u}_\ell^T(n) \tilde{\mathbf{w}}_\ell(n-1) \} \\ + \mu^2 p_\zeta^2 E\{ \tilde{\mathbf{w}}_j^T(n-1) \mathbf{u}_j(n) \mathbf{u}_j^T(n) \mathbf{u}_\ell(n) \mathbf{u}_\ell^T(n) \tilde{\mathbf{w}}_\ell(n-1) \} \end{aligned} \quad (70)$$

Using **A2**, **A4**, and **A6**, from (70) we can write

$$\begin{aligned} E\{ \tilde{\mathbf{w}}_j^T(n-1) \mathbf{u}_j(n) \mathbf{u}_j^T(n) \tilde{\mathbf{w}}_\ell(n-1) \} \\ = E\{ \tilde{\mathbf{w}}_j^T(n-1) \mathbf{u}_\ell(n) \mathbf{u}_\ell^T(n) \tilde{\mathbf{w}}_\ell(n-1) \} = \sigma_u^2 \beta_{j\ell}(n-1) \end{aligned} \quad (71)$$

for any pair of nodes ℓ and j , $\ell \neq j$. Moreover, since the fourth-order moment from (66) is absent in (70), we may write

$$\begin{aligned} & \mathbb{E}\{\tilde{\mathbf{w}}_j^T(n-1)\mathbf{u}_j(n)\mathbf{u}_j^T(n)\mathbf{u}_\ell(n)\mathbf{u}_\ell^T(n)\tilde{\mathbf{w}}_\ell(n-1)\} \\ & = \sigma_u^4\beta_{j\ell}(n-1) \end{aligned} \quad (72)$$

Therefore, with τ defined as in (24), we can write

$$x_{j\ell}(n) = \tau\beta_{j\ell}(n-1). \quad (73)$$

Thus, replacing (69) and (73) in (64) leads to (22).

As evidenced by (22) and (73), we also need to obtain a recursion for $\mathbb{E}\{\tilde{\mathbf{w}}_j^T(n-1)\tilde{\mathbf{w}}_\ell(n-1)\}$, for $j \neq \ell$, in order to analyze the evolution of the MSD of each node. Firstly, we should notice that we can rewrite $\tilde{\mathbf{w}}_j^T(n)\tilde{\mathbf{w}}_\ell(n)$ as a function of the local estimates, i.e.

$$\tilde{\mathbf{w}}_j^T(n)\tilde{\mathbf{w}}_\ell(n) = \sum_{s \in \mathcal{N}_\ell} \sum_{r \in \mathcal{N}_j} c_{s\ell}c_{rj}\tilde{\psi}_r^T(n)\tilde{\psi}_s(n). \quad (74)$$

Replacing (17) in (74), using Assumption **A1**, and taking the expectations from both sides, we arrive at

$$\beta_{j\ell}(n) = \sum_{s \in \mathcal{N}_\ell} \sum_{r \in \mathcal{N}_j} c_{s\ell}c_{rj}x_{rs}(n). \quad (75)$$

Similarly to what we did for (64), the analysis of (75) can be broken down into two cases: when $s=r=t$, and when $s \neq r$. In the first case, we have that $c_{tj} \neq 0$ and $c_{t\ell} \neq 0$ only if the node t is in $\mathcal{N}_j \cap \mathcal{N}_\ell$. Thus, in an analogous manner, we get

$$\beta_{tt}(n) = \mu^2 p_\zeta M \sigma_u^2 \sigma_{v_t}^2 + \theta\beta_{tt}(n-1). \quad (76)$$

For $s \neq r$, we can write

$$\beta_{rs}(n) = \tau\beta_{rs}(n-1) \quad (77)$$

Thus, replacing (76) and (77) in (75), we finally obtain (25).

APPENDIX B ON THE MATRIX Φ

We begin by noting that (25) can be recast as

$$\begin{aligned} \beta_{j\ell}(n) & = \sum_{r=1}^V \sum_{s=1}^V c_{rj}c_{s\ell}[(\theta - \tau)\delta_{rs} + \tau]\beta_{rs}(n-1) \\ & + \mu^2 p_\zeta M \sigma_u^2 \sum_{z=1}^V c_{zj}c_{z\ell}\sigma_{v_z}^2, \end{aligned} \quad (78)$$

for any arbitrary j and ℓ , by simply changing the order in which the elements are added. Thus, if $r = s$, $\beta_{rr}(n-1)$, which corresponds to the MSD of node r , is multiplied by θ and by $c_{rj}c_{r\ell}$. In contrast, if $r \neq s$, $\beta_{rs}(n-1)$ corresponds to the trace of the covariance matrix between $\tilde{\mathbf{w}}_r(n-1)$ and $\tilde{\mathbf{w}}_s(n-1)$, and is multiplied by τ and by $c_{rj}c_{s\ell}$. Thus, if we examine the vector $\beta(n)$ in (14), we notice that it consists of V elements between each pair of consecutive MSD's in the vector, and $V(V-1)$ elements related to cross-terms.

Thus, the matrix Φ that appears in (37) is a matrix that has V columns filled with θ , and between each pair of consecutive columns, there are V columns filled with τ . These columns are multiplied element-wise by the corresponding combination

weights. As an example, let us consider a network formed by only two connected nodes. In this case, we have

$$\Phi = \begin{bmatrix} \theta c_{11}^2 & \tau c_{21}c_{11} & \tau c_{11}c_{21} & \theta c_{21}^2 \\ \theta c_{12}c_{11} & \tau c_{22}c_{11} & \tau c_{12}c_{21} & \theta c_{22}c_{21} \\ \theta c_{11}c_{12} & \tau c_{21}c_{12} & \tau c_{11}c_{22} & \theta c_{21}c_{22} \\ \theta c_{12}^2 & \tau c_{22}c_{12} & \tau c_{12}c_{22} & \theta c_{22}^2 \end{bmatrix}. \quad (79)$$

We should notice that there are $V = 2$ columns that are related to the MSD's, and, in between them, we have also two columns, which are related to the covariances. If we focus on the combination weights, we can see that the matrix Φ carries information from $(\mathbf{C}^T) \otimes (\mathbf{C}^T) = (\mathbf{C} \otimes \mathbf{C})^T$, where the equality follows from the properties of the Kronecker product. It also carries information from τ and θ . Hence, we can see Φ as the element-wise multiplication of two matrices, as in (41): Γ , which is related to the combination weights as in (42), and Ω , which is related to θ and τ as in (43).

APPENDIX C AN UPPER BOUND FOR THE STEADY-STATE NMSD OF COOPERATIVE NETWORKS

For compactness of notation, let us introduce the quantity

$$\varepsilon \triangleq \mathbf{b}^T [\mathbf{I}_{V^2} - \tau\Gamma]^{-1} \boldsymbol{\sigma}. \quad (80)$$

Replacing ε in Eq. (55), we thus obtain

$$\text{NMSD}_\tau(\infty) = \frac{\mu^2 p_\zeta M \sigma_u^2 \varepsilon}{V}. \quad (81)$$

Since ε is a scalar, and the trace of a scalar is the scalar itself, we can write

$$\varepsilon = \text{Tr}(\varepsilon) = \text{Tr} \left\{ \mathbf{b}^T [\mathbf{I}_{V^2} - \tau\Gamma]^{-1} \boldsymbol{\sigma} \right\}. \quad (82)$$

The cyclic property of the trace operator states that we can write

$$\text{Tr}(\mathbf{M}_1\mathbf{M}_2\mathbf{M}_3) = \text{Tr}(\mathbf{M}_2\mathbf{M}_3\mathbf{M}_1) \quad (83)$$

for any arbitrary matrices \mathbf{M}_1 , \mathbf{M}_2 and \mathbf{M}_3 of appropriate dimensions. Applying this property twice to (82), we get

$$\varepsilon = \text{Tr} \left\{ \boldsymbol{\sigma} \mathbf{b}^T [\mathbf{I}_{V^2} - \tau\Gamma]^{-1} \right\}. \quad (84)$$

Defining $\mathbf{S} \triangleq \boldsymbol{\sigma} \mathbf{b}^T$ and $\mathbf{G} \triangleq [\mathbf{I}_{V^2} - \tau\Gamma]^{-1}$, (84) can be recast more compactly as

$$\varepsilon = \text{Tr}(\mathbf{S}\mathbf{G}). \quad (85)$$

At this point, it is worth noting that, if Γ is symmetric, so is the matrix $\mathbf{I} - \tau\Gamma$. Recalling that the inverse of a symmetric matrix is also symmetric, we can conclude that \mathbf{G} is also symmetric. Since $\Gamma = (\mathbf{C} \otimes \mathbf{C})^T$, the matrix Γ is symmetric if, and only if, \mathbf{C} is symmetric. This is guaranteed to occur if the Metropolis rule is adopted, regardless of the network topology, for example [1]. In the adaptive diffusion networks, rules for the selection of the combination weights that lead to a symmetric matrix \mathbf{C} regardless of the topology are sometimes referred to as ‘‘doubly stochastic policies’’ [1]. If we adopt the Uniform rule instead, this is not guaranteed, since it is not a doubly stochastic policy [1]. Nonetheless, the matrix \mathbf{C} may still be symmetric, depending on the network topology.

On the other hand, the matrix \mathbf{S} is not symmetric, and in general is not positive semi-definite. Nonetheless, it has been shown in [47] that, for a real $N \times N$ matrix \mathbf{M}_1 , and for a real symmetric $N \times N$ matrix \mathbf{M}_2 , we may write

$$\text{Tr}(\mathbf{M}_1\mathbf{M}_2) \leq \sum_{i=1}^N \lambda_i(\overline{\mathbf{M}}_1)\lambda_i(\mathbf{M}_2), \quad (86)$$

where $\lambda_i(\cdot)$ denotes the i -th largest eigenvalue, and $\overline{\mathbf{M}}_1 \triangleq \frac{1}{2}(\mathbf{M}_1 + \mathbf{M}_1^T)$. With this notation, we have that $\lambda_1(\cdot) \geq \lambda_2(\cdot) \geq \dots \geq \lambda_N(\cdot)$. In other words, $\lambda_1(\cdot)$ represents the greatest eigenvalue of its argument, i.e., $\lambda_1(\cdot) = \lambda_{\max}(\cdot)$, and $\lambda_N(\cdot)$ denotes the smallest one, i.e., $\lambda_N(\cdot) = \lambda_{\min}(\cdot)$.

In our context, if \mathbf{C} is symmetric, we can replace \mathbf{M}_2 with \mathbf{G} in (86), and $\overline{\mathbf{M}}_1$ with

$$\overline{\mathbf{S}} \triangleq \frac{1}{2}(\mathbf{S} + \mathbf{S}^T) = \frac{1}{2}(\boldsymbol{\sigma}\mathbf{b}^T + \mathbf{b}\boldsymbol{\sigma}^T). \quad (87)$$

Doing so, we straightforwardly obtain

$$\text{Tr}(\mathbf{S}\mathbf{G}) \leq \sum_{i=1}^{V^2} \lambda_i(\overline{\mathbf{S}})\lambda_i(\mathbf{G}). \quad (88)$$

It would be useful to estimate $\lambda_{\max}(\mathbf{G})$. To do so, we begin by noticing that $\boldsymbol{\Gamma}$ is a right-stochastic matrix. As mentioned in Sec. III-B, the spectral radius of these matrices is equal to one [44], [45]. Moreover, at least one of their eigenvalues is necessarily equal to 1. As a result, we know for sure that $\lambda_{\max}(\boldsymbol{\Gamma}) = 1$. Thus, we have that $\lambda_i(\boldsymbol{\Gamma}) \in [-1, 1]$ for $i = 1, 2, \dots, V^2$ and therefore conclude that $\lambda_i(\tau\boldsymbol{\Gamma}) \in [-\tau, \tau]$, $\lambda_i(\mathbf{I} - \tau\boldsymbol{\Gamma}) \in [1 - \tau, 1 + \tau]$, and, consequently,

$$\lambda_i(\mathbf{G}) \in \left[\frac{1}{1 + \tau}, \frac{1}{1 - \tau} \right]. \quad (89)$$

Moreover, since $\lambda_{\max}(\boldsymbol{\Gamma}) = 1$, we conclude from (89) that

$$\lambda_{\max}(\mathbf{G}) = \frac{1}{1 - \tau}. \quad (90)$$

We should notice that, as long as (30) holds, we have that $0 \leq \tau \leq 1$, with $\tau = 1$ only if $p_\zeta = 0$. Thus, for $p_\zeta > 0$, we observe from (90) that $\lambda_{\max}(\mathbf{G}) > 0$. Moreover, since $-1 \leq \lambda_{\min}(\boldsymbol{\Gamma}) \leq 1$, we have that

$$\frac{1}{1 + \tau} \leq \lambda_{\min}(\mathbf{G}) \leq \frac{1}{1 - \tau}. \quad (91)$$

Since $\frac{1}{1 + \tau} > 0$, we thus conclude that

$$0 < \lambda_{\min}(\mathbf{G}) \leq \lambda_{\max}(\mathbf{G}) = \frac{1}{1 - \tau}. \quad (92)$$

We remark that the matrix $\mathbf{S} \triangleq \boldsymbol{\sigma}\mathbf{b}^T$ is formed by the outer product of two nonzero vectors. The rank of such matrices is equal to one, and their non-zero eigenvalue is given by [44]

$$\lambda_1(\mathbf{S}) = \text{Tr}(\mathbf{S}) = \text{Tr}(\mathbf{b}\boldsymbol{\sigma}^T) = \mathbf{b}^T\boldsymbol{\sigma}, \quad (93)$$

where we adopted the index “1” because $\mathbf{b}^T\boldsymbol{\sigma} > 0$ and thus this is the greatest eigenvalue of \mathbf{S} . Moreover, since transposing a matrix does not change its rank, we thus notice that $\text{rank}(\mathbf{S}) = \text{rank}(\mathbf{S}^T) = 1$. Since $\text{rank}(\mathbf{M}_1 + \mathbf{M}_2) \leq \text{rank}(\mathbf{M}_1) + \text{rank}(\mathbf{M}_2)$ for any arbitrary matrices \mathbf{M}_1 and \mathbf{M}_2 of the same dimensions [44], we thus conclude that

$$\text{rank}(\overline{\mathbf{S}}) \leq \text{rank}(\mathbf{S}) + \text{rank}(\mathbf{S}^T) = 2. \quad (94)$$

In fact, from (87), we can see that $\overline{\mathbf{S}}$ will be a rank-two matrix, unless \mathbf{b}^T and $\boldsymbol{\sigma}$ are linearly dependent, in which case the rank of $\overline{\mathbf{S}}$ will be equal to one. However, comparing the vectors $\mathbf{b}^T \triangleq \text{vec}\{\mathbf{I}_V\}$ and $\boldsymbol{\sigma}$ given by (40), we notice that they cannot be linearly dependent unless $c_{ki} = 0$ for every $i \neq k$, i.e., the network is non-cooperative. Since (55) only holds as valid approximation for the cooperative scheme, we conclude that, for the scenario we are interested in analyzing, the rank of $\overline{\mathbf{S}}$ cannot be equal to one. Finally, we remark that the rank of $\overline{\mathbf{S}}$ cannot be zero either, as this would lead to the conclusion that the steady-state NMSD is equal to zero. Thus, $\overline{\mathbf{S}}$ is a rank-two matrix. Since it is also symmetric, and therefore diagonalizable, this means that $\overline{\mathbf{S}}$ has exactly two non non-zero eigenvalues, or one non-zero eigenvalue with multiplicity two. Let us denote them by $\lambda'(\overline{\mathbf{S}})$ and $\lambda''(\overline{\mathbf{S}})$. For convenience, we use $\lambda'(\overline{\mathbf{S}})$ to refer to the greatest eigenvalue of $\overline{\mathbf{S}}$, and $\lambda''(\overline{\mathbf{S}})$ to refer to the other one, i.e., $\lambda'(\overline{\mathbf{S}}) \geq \lambda''(\overline{\mathbf{S}})$.

Moreover, we notice that

$$\lambda'(\overline{\mathbf{S}}) + \lambda''(\overline{\mathbf{S}}) = \text{Tr}(\overline{\mathbf{S}}), \quad (95)$$

as all the other eigenvalues of $\overline{\mathbf{S}}$ are equal to zero. Since transposing a matrix does not change its trace, Eq. (87) leads to

$$\text{Tr}(\overline{\mathbf{S}}) = \text{Tr}(\mathbf{S}). \quad (96)$$

Thus, using (95), we may write

$$\text{Tr}(\overline{\mathbf{S}}) = \mathbf{b}^T\boldsymbol{\sigma}. \quad (97)$$

At this point, recalling (15), we may write

$$\mathbf{b}^T\boldsymbol{\sigma} = \text{Tr}(\mathbf{I}_V \mathbf{C} \mathbf{R}_v \mathbf{C}^T) = \text{Tr}(\mathbf{C} \mathbf{R}_v \mathbf{C}^T). \quad (98)$$

Replacing (39) in (98) then yields

$$\mathbf{b}^T\boldsymbol{\sigma} = \sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2. \quad (99)$$

Thus, from (97) and (99), we can write

$$\lambda'(\overline{\mathbf{S}}) + \lambda''(\overline{\mathbf{S}}) = \sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2. \quad (100)$$

Since $\sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 > 0$, we conclude that at least one of the eigenvalues of $\overline{\mathbf{S}}$ is positive. Since we are considering that $\lambda'(\overline{\mathbf{S}}) \geq \lambda''(\overline{\mathbf{S}})$, we must have

$$\lambda'(\overline{\mathbf{S}}) > 0$$

and, therefore,

$$\lambda_1(\overline{\mathbf{S}}) = \lambda'(\overline{\mathbf{S}}). \quad (101)$$

In principle, $\lambda''(\overline{\mathbf{S}})$ could be either positive or negative, provided that (100) holds. As discussed previously, it cannot be equal to zero because $\text{rank}(\overline{\mathbf{S}}) = 2$. However, Bendixson's inequality states that, for any arbitrary matrix \mathbf{M} , we have [48]

$$\lambda_{\min} \left[\frac{1}{2}(\mathbf{M} + \mathbf{M}^T) \right] \leq \text{Re}[\lambda(\mathbf{M})] \leq \lambda_{\max} \left[\frac{1}{2}(\mathbf{M} + \mathbf{M}^T) \right] \quad (102)$$

for every eigenvalue λ of \mathbf{M} . We remark that in (102), $\text{Re}[\cdot]$ denotes the real part. Thus, if we assume that $\lambda''(\overline{\mathbf{S}}) > 0$, we

$$\text{Tr}(\mathbf{S}\mathbf{G}) \leq \frac{1}{1-\tau} \cdot \left\{ \frac{1}{2} \left[\sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 + \sqrt{V \sum_{i=1}^V \sum_{j=1}^V \left[\left(\sum_{k=1}^V c_{ki} c_{kj} \sigma_{v_k}^2 \right)^2 \right]} \right] \right\}. \quad (109)$$

$$\text{NMSD}_\tau(\infty) \leq \frac{\mu^2 p_\zeta M \sigma_u^2}{V} \cdot \frac{1}{1-\tau} \cdot \left\{ \frac{1}{2} \left[\sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 + \sqrt{V \sum_{i=1}^V \sum_{j=1}^V \left[\left(\sum_{k=1}^V c_{ki} c_{kj} \sigma_{v_k}^2 \right)^2 \right]} \right] \right\}, \quad (110)$$

would have that $\lambda_{\min}(\bar{\mathbf{S}}) = 0$. In this case, from (87), (101), and (102) we should be able to write

$$0 \leq \sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 \leq \lambda'(\bar{\mathbf{S}}). \quad (103)$$

However, this leads to a contradiction. From (100) and (103) we respectively get $\lambda'(\bar{\mathbf{S}}) + \lambda''(\bar{\mathbf{S}}) = \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2$ and $\lambda'(\bar{\mathbf{S}}) \geq \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2$. Both expressions cannot be simultaneously true if $\lambda''(\bar{\mathbf{S}})$ is positive, which contradicts our initial assumption. Thus, $\lambda''(\bar{\mathbf{S}})$ is necessarily negative, and we may write

$$\lambda_{V^2}(\bar{\mathbf{S}}) = \lambda''(\bar{\mathbf{S}}) < 0. \quad (104)$$

Therefore, from (88), we get

$$\text{Tr}(\mathbf{S}\mathbf{G}) \leq \lambda_{\max}(\mathbf{G})\lambda'(\bar{\mathbf{S}}) + \lambda_{\min}(\mathbf{G})\lambda''(\bar{\mathbf{S}}). \quad (105)$$

From (92) and (104), we notice that $\lambda_{\min}(\mathbf{G})\lambda''(\bar{\mathbf{S}}) < 0$, from which we may write

$$\text{Tr}(\mathbf{S}\mathbf{G}) \leq \lambda_{\max}(\mathbf{G})\lambda'(\bar{\mathbf{S}}), \quad (106)$$

which leads to

$$\text{Tr}(\mathbf{S}\mathbf{G}) \leq \frac{\lambda'(\bar{\mathbf{S}})}{1-\tau}. \quad (107)$$

Hence, we need to determine $\lambda'(\bar{\mathbf{S}})$. At this point, it is useful to notice that the matrix $\bar{\mathbf{S}}$ can be written as $\bar{\mathbf{S}} = \frac{1}{2} \bar{\mathbf{S}}_A \cdot \bar{\mathbf{S}}_B$,

where $\bar{\mathbf{S}}_A = [\boldsymbol{\sigma} \quad \mathbf{b}]$ is a $V \times 2$ matrix and $\bar{\mathbf{S}}_B = \begin{bmatrix} \mathbf{b}^T \\ \boldsymbol{\sigma}^T \end{bmatrix}$

is a $2 \times V$ matrix. We then use the fact that the non-zero eigenvalues of $\bar{\mathbf{S}}_A \cdot \bar{\mathbf{S}}_B$ are the same as those of $\bar{\mathbf{S}}_B \cdot \bar{\mathbf{S}}_A$, which corresponds to the 2×2 matrix $\bar{\mathbf{S}}_B \cdot \bar{\mathbf{S}}_A = \begin{bmatrix} \mathbf{b}^T \boldsymbol{\sigma} & \mathbf{b}^T \mathbf{b} \\ \boldsymbol{\sigma}^T \boldsymbol{\sigma} & \mathbf{b}^T \boldsymbol{\sigma} \end{bmatrix}$.

After some algebra, we therefore conclude that

$$\lambda'(\bar{\mathbf{S}}) = \frac{1}{2} (\mathbf{b}^T \boldsymbol{\sigma} + \|\mathbf{b}\| \|\boldsymbol{\sigma}\|). \quad (108)$$

Thus, we have to evaluate $\|\mathbf{b}\|$ and $\|\boldsymbol{\sigma}\|$. From (15), we get

$$\|\mathbf{b}\|^2 = \mathbf{b}^T \mathbf{b} = \text{vec}\{\mathbf{I}_V\}^T \text{vec}\{\mathbf{I}_V\} = \text{Tr}(\mathbf{I}_V) = V, \quad (111)$$

and therefore

$$\|\mathbf{b}\| = \sqrt{V}. \quad (112)$$

As for the vector $\boldsymbol{\sigma}$, from (40) we get

$$\|\boldsymbol{\sigma}\| = \sqrt{\sum_{i=1}^V \sum_{j=1}^V \left[\left(\sum_{k=1}^V c_{ki} c_{kj} \sigma_{v_k}^2 \right)^2 \right]}. \quad (113)$$

Thus, replacing (99), (112), and (113) in (108), we obtain

$$\lambda'(\bar{\mathbf{S}}) = \frac{\sum_{i=1}^V \sum_{k=1}^V c_{ki}^2 \sigma_{v_k}^2 + \sqrt{V \sum_{i=1}^V \sum_{j=1}^V \left[\left(\sum_{k=1}^V c_{ki} c_{kj} \sigma_{v_k}^2 \right)^2 \right]}}{2}. \quad (114)$$

Lastly, replacing (114) in (107), we obtain Eq. (109) presented at the top of this page.

Replacing (109) in (85) and then in (81), we conclude that, for an arbitrary topology with doubly stochastic combination weights, the upper bound of the steady-state NMSD is given by (110), which is also presented at the top of the page. Then, after some algebraic manipulations, one finally obtains (56).

APPENDIX D OBTAINING (57)

Firstly, it is useful to note that $[\mathbf{I}_{V^2} - \tau \boldsymbol{\Gamma}]^{-1} = \sum_{n=0}^{\infty} (\tau \boldsymbol{\Gamma})^{n_i} = \mathbf{I}_{V^2} + \sum_{n_i=1}^{\infty} (\tau \boldsymbol{\Gamma})^{n_i}$. At this point, one useful property of the matrix $\boldsymbol{\Gamma}_{K_V} = \frac{1}{V^2} \mathbf{1}_{V^2 \times V^2}$ is that $\boldsymbol{\Gamma}_{K_V}^2 = \boldsymbol{\Gamma}_{K_V}$. Thus, we have that

$$[\mathbf{I}_{V^2} - \tau \boldsymbol{\Gamma}_{K_V}]^{-1} = \mathbf{I}_{V^2} + \frac{\tau}{(1-\tau)V^2} \mathbf{1}_{V^2 \times V^2}. \quad (115)$$

Multiplying (115) from the left by \mathbf{b}^T leads to

$$\mathbf{b}^T [\mathbf{I}_{V^2} - \tau \boldsymbol{\Gamma}_{K_V}]^{-1} = \mathbf{b}^T + \frac{\tau}{(1-\tau)V} \mathbf{1}_{V^2}^T, \quad (116)$$

By applying the inverse vec operator to the right-hand side of (116), we obtain

$$\text{vec}^{-1} \left\{ \mathbf{b}^T + \frac{\tau}{(1-\tau)V} \mathbf{1}_{V^2}^T \right\} = \mathbf{I}_V + \frac{\tau}{(1-\tau)V} \mathbf{1}_{V \times V}. \quad (117)$$

Thus, using (15), (116), and (117), and introducing $\xi \triangleq \mathbf{b}^T [\mathbf{I}_{V^2} - \tau \boldsymbol{\Gamma}_{K_V}]^{-1} \text{vec}\{\mathbf{C}\mathbf{R}_v \mathbf{C}^T\}$, we can write

$$\begin{aligned} \xi &= \text{Tr} \left\{ \left[\mathbf{I}_V + \frac{\tau}{(1-\tau)V} \mathbf{1}_{V \times V} \right] \frac{1}{V^2} \mathbf{1}_{V \times V} \mathbf{R}_v \mathbf{1}_{V \times V} \right\} \\ &= \frac{1}{V^2} \left(1 + \frac{\tau}{1-\tau} \right) \text{Tr}\{\mathbf{1}_{V \times V} \mathbf{R}_v \mathbf{1}_{V \times V}\}, \end{aligned} \quad (118)$$

where we used the fact that $\mathbf{C}_{K_V} = \mathbf{C}_{K_V}^T = \frac{1}{V} \mathbf{1}_{V \times V}$ and that $\mathbf{1}_{V^2 \times V^2}^2 = V \mathbf{1}_{V \times V}$. Moreover, from (83), we get

$$\text{Tr}\{\mathbf{1}_{V \times V} \mathbf{R}_v \mathbf{1}_{V \times V}\} = V \text{Tr}\{\mathbf{R}_v \mathbf{1}_{V \times V}\} = V \sum_{k=1}^V \sigma_{v_k}^2, \quad (119)$$

where we took advantage from the fact that \mathbf{R}_v is a diagonal matrix. Thus, we obtain

$$\xi = \frac{1}{V^2} \left(1 + \frac{\tau}{1-\tau} \right) \cdot V \cdot \sum_{k=1}^V \sigma_{v_k}^2 = \frac{1}{1-\tau} \frac{\sum_{k=1}^V \sigma_{v_k}^2}{V}. \quad (120)$$

Finally, replacing (120) in (55) leads to (57).

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] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Library in Signal Processing: array and statistical signal processing*, R. Chellapa and S. Theodoridis, Eds., vol. 3, chapter 9, pp. 323–456. Academic Press, 2014.
- [3] A. H. Sayed, "Adaptive networks," *Proc. IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014.
- [4] 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, Jun. 2008.
- [5] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Oct. 2009.
- [6] 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, Apr. 2008.
- [7] L. Li and J. A. Chambers, "Distributed adaptive estimation based on the APA algorithm over diffusion networks with changing topology," in *Proc. IEEE SSP*, 2009, pp. 757–760.
- [8] 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, Nov. 2017.
- [9] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "Distributed diffusion adaptation over graph signals," in *Proc. IEEE ICASSP*, 2018, pp. 4129–4133.
- [10] F. Hua, R. Nassif, C. Richard, H. Wang, and A. H. Sayed, "A preconditioned graph diffusion lms for adaptive graph signal processing," in *Proc. EUSIPCO*, 2018, pp. 111–115.
- [11] P. Di Lorenzo, S. Barbarossa, and A. H. Sayed, "Bio-inspired decentralized radio access based on swarming mechanisms over adaptive networks," *IEEE Trans. Signal Process.*, vol. 61, no. 12, pp. 3183–3197, Apr. 2013.
- [12] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis," in *Proc. IEEE ICASSP*, 2006, vol. III, pp. 584–587.
- [13] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, pp. 4064–4077, Aug. 2007.
- [14] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proc. IEEE Conf. Decision Control (CDC)*, 2005, pp. 6698–6703.
- [15] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Dynamic consensus on mobile networks," in *IFAC World Congress*, 2005, pp. 1–6.
- [16] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Trans. Signal Process.*, vol. 62, no. 16, pp. 4129–4144, Jun. 2014.
- [17] J. Chen, C. Richard, and A. H. Sayed, "Diffusion LMS over multitask networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2733–2748, Mar. 2015.
- [18] J. Plata-Chaves, N. Bogdanović, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," *IEEE Trans. Signal Process.*, vol. 63, no. 13, pp. 3448–3460, 2015.
- [19] V. C. Gogineni, S. P. Talebi, and S. Werner, "Performance of clustered multitask diffusion LMS suffering from inter-node communication delays," *IEEE Trans. Circuits Syst. II Express Briefs*, Jan. 2021.
- [20] W. Gao, J. Chen, C. Richard, and J. Huang, "Diffusion adaptation over networks with kernel least-mean-square," in *Proc. IEEE Int. Workshop on Comput. Adv. in Multi-Sensor Adaptive Process. (CAMSAP)*, 2015, pp. 217–220.
- [21] S. Chouvardas and M. Draief, "A diffusion kernel LMS algorithm for nonlinear adaptive networks," in *Proc. IEEE ICASSP*, 2016, pp. 4164–4168.
- [22] B.-S. Shin, M. Yukawa, R. L. G. Cavalcante, and A. Dekorsy, "Distributed adaptive learning with multiple kernels in diffusion networks," *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5505–5519, Aug. 2018.
- [23] P. Bouboulis, S. Theodoridis, and S. Chouvardas, "A random Fourier features perspective of KAFs with application to distributed learning over networks," in *Adaptive Learning Methods for Nonlinear System Modeling*, pp. 149–172. Elsevier, 2018.
- [24] P. Di Lorenzo, P. Banelli, S. Barbarossa, and S. Sardellitti, "Distributed adaptive learning of graph signals," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4193–4208, May 2017.
- [25] P. Di Lorenzo, E. Isufi, P. Banelli, S. Barbarossa, and G. Leus, "Distributed recursive least squares strategies for adaptive reconstruction of graph signals," in *Proc. EUSIPCO*, 2017, pp. 2289–2293.
- [26] D. G. Tiglea, R. Candido, and M. T. M. Silva, "A low-cost algorithm for adaptive sampling and censoring in diffusion networks," *IEEE Trans. Signal Process.*, vol. 69, pp. 58–72, Nov. 2020.
- [27] P. Di Lorenzo, P. Banelli, E. Isufi, S. Barbarossa, and G. Leus, "Adaptive graph signal processing: Algorithms and optimal sampling strategies," *IEEE Trans. Signal Process.*, vol. 66, no. 13, pp. 3584–3598, Jul. 2018.
- [28] D. G. Tiglea, R. Candido, and M. T. M. Silva, "An adaptive algorithm for sampling over diffusion networks with dynamic parameter tuning and change detection mechanisms," *Digit. Signal Process.*, vol. 127, pp. 103587, Jul. 2022.
- [29] S.-Y. Tu and A. H. Sayed, "On the influence of informed agents on learning and adaptation over networks," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1339–1356, Nov. 2012.
- [30] A. H. Sayed, S.-Y. Tu, and J. Chen, "Online learning and adaptation over networks: More information is not necessarily better," in *IEEE Inf. Theory Appl. Workshop (ITA)*, 2013, pp. 1–8.
- [31] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, Apr. 2013.
- [32] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part I: Modeling and stability analysis," *IEEE Trans. Signal Process.*, vol. 63, no. 4, pp. 811–826, Dec. 2014.
- [33] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part II: Performance analysis," *IEEE Trans. Signal Process.*, vol. 63, no. 4, pp. 827–842, Dec. 2014.
- [34] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part III: Comparison analysis," *IEEE Trans. Signal Process.*, vol. 63, no. 4, pp. 843–858, Dec. 2014.
- [35] A. H. Sayed, *Adaptive Filters*, John Wiley & Sons, NJ, 2008.
- [36] S. Haykin, *Adaptive Filter Theory*, Pearson, Upper Saddle River, 5th edition, 2014.
- [37] V. H. Nascimento and M. T. M. Silva, "Adaptive filters," in *Signal Processing Theory and Machine Learning*, P. S. R. Diniz, Ed., chapter 12, pp. 717–868. Academic Press, 2024.
- [38] C. G. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proc. IEEE ICASSP*, 2008, pp. 3285–3288.
- [39] L. Lu and H. Zhao, "Diffusion leaky LMS algorithm: Analysis and implementation," *Signal Process.*, vol. 140, pp. 77–86, Nov. 2017.
- [40] S. Werner, Y.-F. Huang, M. L. R. De Campos, and V. Koivunen, "Distributed parameter estimation with selective cooperation," in *Proc. IEEE ICASSP*, 2009, pp. 2849–2852.
- [41] 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, no. 9, pp. 4795–4810, Jun. 2010.
- [42] C.-K. Yu and A. H. Sayed, "A strategy for adjusting combination weights over adaptive networks," in *Proc. IEEE ICASSP*, 2013, pp. 4579–4583.
- [43] S.-Y. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *Proc. IEEE Int. Workshop on Comput. Adv. in Multi-Sensor Adaptive Process. (CAMSAP)*, 2011, pp. 317–320.
- [44] C. D. Meyer, *Matrix analysis and applied linear algebra*, vol. 2, Siam, 2000.
- [45] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. IEEE Conf. Decision Control (CDC)*, 2005, pp. 2996–3000.
- [46] J.-W. Lee, J.-T. Kong, W.-J. Song, and S.-E. Kim, "Data-reserved periodic diffusion LMS with low communication cost over networks," *IEEE Access*, vol. 6, pp. 54636–54650, Sep. 2018.
- [47] J. B. Lasserre, "A trace inequality for matrix product," *IEEE Transactions on Automatic Control*, vol. 40, no. 8, pp. 1500–1501, 1995.
- [48] L. Mirsky, *An Introduction to Linear Algebra*, Dover Books on Mathematics. Dover Publications, 2012.