Adaptive Diffusion Networks: An Overview

Daniel Gilio Tiglea*, Renato Candido, Magno T. M. Silva
Escola Politécnica, University of São Paulo, São Paulo, Brazil

Abstract

This work provides a comprehensive overview of adaptive diffusion networks, from the first papers published on the subject to state-of-the-art solutions and current challenges. These networks consist of a collection of agents that can measure and learn from streaming data locally, and cooperate to improve the overall performance. Since their inception, adaptive diffusion networks have consolidated themselves as interesting tools for distributed estimation and learning, and have spun several types of solutions for these problems. We begin by discussing the technological advances that led to their emergence, and present the many ramifications of the area. We also discuss some of the most critical limitations of these types of networks in practical situations, such as energy consumption, and show techniques that have been proposed to cope with them. Finally, simulations with real-world data are presented in order to illustrate in practice the opportunities and challenges that they pose.

Keywords: Diffusion networks, distributed estimation, distributed signal processing, multitask learning, kernel adaptive filtering, graph signal processing.

1. Introduction

More than one decade after their inception, adaptive diffusion networks have become a consolidated tool in the signal processing literature [1–6]. Widely regarded as an effective tool for distributed parameter estimation and signal processing, they gained widespread attention due to their advantages over centralized approaches and branched out into many different research topics, such as multitask networks [7–18], nonlinear adaptive networks [19–27], among others. Moreover, the field of graph signal processing (GSP) has oftentimes been inspired by these techniques, since it deals with applications that are usually distributed in nature [28–31]. As a result, 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.

Yet, there are still many open research topics on the area of adaptive diffusion networks, whose relevance is only increased by the multitude of applications and solutions inspired by them. For instance, the energy

*Corresponding author.
Email addresses: dtiglea@lps.usp.br (Daniel Gilio Tiglea), renatocan@lps.usp.br (Renato Candido), magno.silva@usp.br (Magno T. M. Silva)

and computation constraints involved in distributed signal processing tasks still pose a challenge that is a topic of research to this day. Moreover, additional challenges arise when the nodes have potentially different tasks, or when there are significant nonlinearities in the environment, for instance. Consequently, new algorithms continue to be proposed to deal with these types of circumstances, as well as other challenging scenarios.

Inspired by the influence of adaptive diffusion networks on distributed signal processing over the last decade, the goal of this paper is to provide a review on these powerful tools. Starting from the emergence of adaptive diffusion networks in the literature and some of the seminal algorithms, we study the many ramifications that these solutions have spun. The opportunities and challenges posed by these tools are examined, and we seek to point out what type of problems each one of the solutions presented in the paper is best suited for. It is worth noting that we are primarily concerned with adaptive diffusion networks designed for distributed linear and nonlinear adaptive filtering. In recent years, the term "adaptive networks" has sometimes been employed in a wider sense to refer to networked strategies for, e.g., optimization [32, 33] and social learning [34–36]. However, covering these topics would turn an already wide scope into an unmanageable one, at least for a single paper.

Organization of the paper and major contributions

This paper is organized as follows. In Sec. 2, we examine the historical development of adaptive diffusion networks in the literature. In Sec. 3, we review the single-task adaptive diffusion networks for linear adaptive filtering. In Sec. 4, we provide an overview of techniques for restricting the number of communication processes between the nodes of a network, which is important for their feasibility in practical applications. In Sec. 5, we review multitask adaptive diffusion networks, whereas in Sec. 6 we study kernel-based adaptive diffusion networks. In Sec. 7, we discuss adaptive diffusion networks that incorporate aspects from the GSP framework. Finally, in Sec. 8 we present simulation results to illustrate the behavior of the solutions covered, and Sec. 9 closes the paper with the conclusions.

As can be attested from the organization of the paper, this work aims to be as comprehensive as possible within the scope of adaptive diffusion networks. Consequently, it distinguishes itself from other review papers published on the field, such as [3, 37–40], by presenting simultaneously all of the following features:

1. **Technological background**: in addition to reviewing the algorithms proposed for adaptive diffusion networks, we also examine the technological developments that lead to the emergence of adaptive diffusion networks from a hardware perspective, including the evolution of wireless communication networks, embedded systems, low-power devices, and so on. Unlike the present paper, this discussion is not covered in [3, 37–40];

2. **Comprehensiveness**: By addressing multitask [7–14] and kernel-based [19–27] networks, as well as diffusion solutions for GSP [28–31] in a single paper, the present work is more comprehensive than others published previously. For instance, the focus of [3, 37] is on the single-task networks for linear adaptive filtering covered in Sec. 3. The same can be said about [39], which specifically emphasizes robust algorithms for these networks. In contrast, in [38, 40], the authors focus mainly on multitask solutions. Kernel-based adaptive diffusion networks are not covered in [3, 37–40]. By gathering information on all of these different solutions in one paper, it becomes easier for the reader to understand what types of problems each of the solutions is aimed at, and to obtain a general overview of the area;

3. **Concerns with practical implementations**: the attention devoted to restrictive communication policies – a concern that arises from the energy consumption due to the communication between nodes, which poses a critical constraint in battery-operated devices – is another difference in comparison with other papers. These techniques are not covered in [3, 37–40];

4. **Real-world data**: the simulations presented in Sec. 8 are based on real-world data, enabling us to we compare the several solutions addressed in this paper in a realistic scenario. This is different from, e.g., [3, 37, 39].

2. Brief History of Adaptive Diffusion Networks

In this section, we present an overview of the timeline of adaptive diffusion networks, as well as correlated research areas and technological advances. For ease of reading, we have divided it in the following subsections. In Sec. 2.1, we explore the context that lead to the inception of adaptive diffusion networks, which is then addressed in Sec. 2.2, along with other strategies for distributed signal processing. In Sec. 2.3, we examine the consolidation of these tools in the literature and some of the research topics that were spanned by it in the following years after its emergence. Finally, in Sec. 2.4, we address other innovations that happened in parallel with the consolidation of adaptive diffusion networks, and the birth of other research fields that drew inspiration from them at some point.

2.1. Technological Background: The Emergence of Wireless Sensor Networks

The late 1990's and early 2000's were marked by rapid progress in the fields of electronics and wireless communications. Advances in commercial integrated circuit fabrication and in very large-scale integration (VLSI) technologies enabled the combination of wireless transceivers, processors, and sensors on a single chip and amplified the usage of computing and communication devices in commercial applications [41, 42]. There were also remarkable breakthroughs in wireless communications technologies. Just to put things in

perspective, in 1998, the Bluetooth technology was launched as an open standard for wireless communications [42]. Another project, initiated in 1990 by the IEEE, would lead to the release of a wireless networking standard in 1997: the IEEE 802.11 [43]. This, in its turn, would become the basis for wireless local access areas (WLANs), and give rise to a family of wireless network protocols that received the brand name of Wi-Fi™ in 1999 [44]. In 2001, the first commercial 3G mobile service was launched in Japan [66].

Such advances would lead to the development of low-cost, low-power micro-sensors with embedded processors and radios [46–49]. Thus, these devices were capable of sensing and processing data, as well as communicating untethered over short distances [50, 51]. This made them promising mainly for three reasons:

1. Their low cost facilitated the deployment of numerous sensors over an area in order to monitor a signal of interest, which is particularly beneficial when the exact location of such signal is not known beforehand. In this case, the placement of many sensors can lead to improved opportunities for line of sight as well as signal-to-noise ratio (SNR) [51].

2. Their capability of wireless communications enabled their use in areas where wired networking was impractical.

3. Their low power consumption allowed them to run on small energy sources, dismissing the need for a continuous connection to the energy infrastructure (e.g., the power grid).

These features enabled the use of such micro-sensors in remote regions and made them suitable for environmental monitoring, precision agriculture, smart homes, military applications, among many other applications [50, 51].

Thus, although energy consumption and production costs were still too high to make them feasible for large-scale applications in the short term [52], expectations arose that Wireless Sensor Networks (WSN's) would emerge in the following years or decades and revolutionize our relations with complex systems [51]. Indeed, over the 2000's, some of the initial shortcomings were addressed. For instance, although Bluetooth and WLAN are not specifically suitable for low-power networks, other standardization attempts would arise in order to meet the needs of WSN's, such as ZigBee [53], WirelessHART [54], and SP100.11a [55]. Moreover, the 6LoWPAN standard would be proposed in order to make WSN's compatible with the internet [56].

Among the technical challenges posed by WSN's, energy consumption was the most demanding. This was mainly due to their reliance on small power sources and on wireless communications. Besides presenting short range, the minimum output power required to transmit a signal over a distance d is proportional to d^n, where $2 \leqslant n < 4$ [50]. For low-lying antennae and near-ground channels, as is the case in most sensor networks, the exponent n is close to 4 due to ground reflections [49–51].

As a result, distributed signal processing techniques were soon perceived to be more appropriate than centralized approaches for WSN's [57–60]. In a distributed setting, each sensor node only communicates with its immediate neighbors. On the other hand, in the centralized framework, every node must send its data to a central processing unit, sometimes called a *fusion center*. However, due to the energy constraints, it is desirable to process the data as much as possible within the network, in order to limit the amount of bits transmitted over long distances [51]. Another disadvantage associated with this approach resides in the fact that it inserts a critical point of failure in the setting, namely, the fusion center. Furthermore, the central unit must be capable of dealing with large amounts of data, meaning that it requires more sophisticated processors and communication modules. Lastly, the short range of the radio components in the sensors limits the scalability of the network in the centralized setting.

Furthermore, there was a concern that the mass production of micro-sensors and their large-scale usage in future applications would create an impracticable demand on cable installation and network bandwidth [49]. Thus, by processing as much data as possible locally, the financial, computational, and management burden on communication systems and networks could be significantly alleviated.

The quickly-developing nature of this topic sparked the interest of the academic community. In 2000, the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) had a section entitled "Signal Processing and Protocols for Wireless Sensors". It was the first edition of the conference to dedicate an entire section to this topic. In the following year, there were three sections dedicated to wireless networks, communications, and systems in ICASSP. That same year, a workshop entitled *Collaborative Signal Processing* was held in Palo Alto, California. This would later be renamed as the *IEEE/ACM International Conference on Information Processing in Sensor Networks*, a symposium that has been held every year since then. In 2002, the Association for Computing Machinery held its *1st ACM International Workshop on Wireless Sensor Networks and Applications*. In 2004, the European Association for Signal Processing (EURASIP) launched an open-access journal entitled *Eurasip Journal on Wireless Communications and Networking*, which places an emphasis on signal processing techniques for these technologies [61]. It is interesting to note that, around the same time, the interest of the scientific community in complex networks began to rise for a number of factors. Firstly, the study of complex systems in general was aided by the increasing availability of powerful computers. In particular, this facilitated computations involving networks with millions of nodes. Moreover, the popularization of the internet also played a role, by enabling the exchange of databases among researchers. In fact, the internet itself was an example of a complex network, and was the subject of a handful of studies [62–64].

The prominence that the aforementioned conferences gave to the topic of wireless sensor networks is an indication of the interest that it sparked among the engineering and computer science communities. This enthusiasm would not fade in the following years – rather, it would skyrocket. As depicted in Fig. 1, between 2002 and 2005 the number of academic publications with the words "wireless sensor networks" in

their title escalated from a few tens to more than a thousand, including conference and journal papers, editorials, books and book chapters, among others [65]. Furthermore, attention on the topic continued to grow steadily throughout the 2000's, and it maintained the widespread interest of the scientific community during the 2010's.

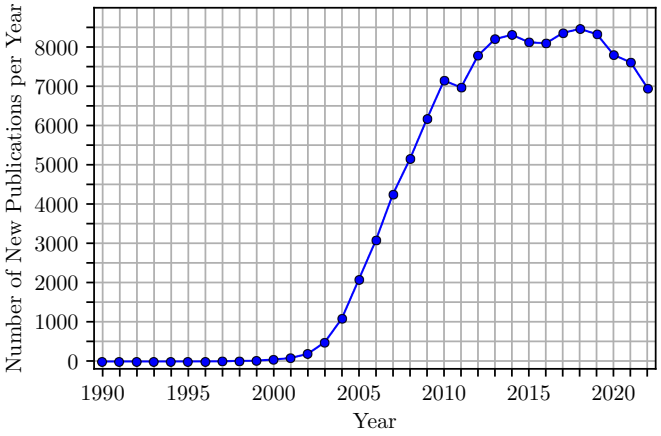

Figure 1: Number of publications with the words "wireless sensor networks" in their title from 1991 through 2022 [65].

2.2. How to Distribute the Processing?

Up until the mid 2000's, wireless sensor networks were oftentimes considered as a tool for directly estimating a certain signal of interest. Thus, the main tasks of WSN's were typically to filter the measurement noise out using, e.g., linear filtering, and to estimate a parameter of interest utilizing the maximum likelihood method, for instance. The results corresponding to different spatial locations were then averaged [67–71]. In the second half of the decade, *adaptive sensor networks with distributed processing* were proposed [1–6, 72, 73]. These solutions consisted in the extension of well-established adaptive filtering algorithms, such as least-mean-squares (LMS) and recursive least-squares (RLS) algorithms [74, 75], to distributed problems over sensor networks. With this fusion of ideas, WSN's could be endowed with the ability to solve optimization problems in an adaptive manner, which enabled them to follow changes in the environment, improved their overall flexibility, and broadened their scope of applications, including, e.g., the study of biological and social networks, system identification problems, distributed spectrum sensing in cognitive radio networks, environmental monitoring, among others [1, 9, 10, 20, 76].

In parallel, another question was the subject of much scrutiny throughout the 2000's: *how* exactly to carry out the processing in a distributed manner. More specifically, researchers sought to determine how to organize the communication between nodes and how to incorporate their cooperation in the processing of the data in an efficient manner, ensuring satisfactory performance as well as low power consumption and computational cost. Some of the earliest approaches to data fusion in sensor networks were the so-called

flooding techniques, such as the ones used in conventional *ad hoc* networks. In its purest form, flooding works as follows: each node keeps a table where it stores all its known data. At every time instant, each node then broadcasts this table to its neighbors. In a stationary environment, this means that eventually every node will be able to act as a fusion center and estimate the parameters of interest [67]. Naturally, this procedure demands high storage and communication capabilities and generates a large overhead [68]. More efficient versions of flooding were proposed over the years [77–80], but the potential for increased node density in WSN's and the technical limitations of each individual sensor made this technique unappealing in the field.

In the mid 2000's, two approaches for communication and cooperation between nodes began to attract attention in the context of wireless sensor networks: the *consensus* [67–69, 81–84] and *incremental* [60, 72, 73, 85] strategies. Both approaches predate the emergence of WSN's, hailing from the fields of statistics [86, 87] and optimization theory [85], respectively. However, as will be seen in Sec. 3, they present some shortcomings that hindered their potential for application in the specific context of WSN's. The incremental strategy is not robust to link and node failures, since the breakdown of a single node or link halts the entire learning process due to the disruption of the information chain. Moreover, it requires that the nodes be arranged in a Hamiltonian cycle, which is, in general, an NP-hard problem [88]. In contrast, in consensus techniques, the nodes are allowed to communicate with each other according to a predefined topology. As a result, they may cooperate with multiple peers, which eliminates some of the issues associated with the incremental strategy. However, they also present some drawbacks. Initially, they required the existence of two time-scales: one for the processing of the data, and the other for reaching a consensus between the nodes at each iteration of the adaptation problem [68, 89]. Furthermore, the step sizes in traditional consensus techniques diminish over time. It was noted that both of these traits could be problematic when dealing with online applications with streaming data. As a result, adjustments were proposed to the consensus schemes in order to alleviate some of these challenges, but some discrepancies remained [90, 91]. A more direct technique for adaptation over networks was proposed earlier leading to the *diffusion strategies* [4–6, 60, 92]. Later, it was shown that the diffusion strategies provided better stability than their consensus counterparts for adaptation with in-network processing [93]. As a result, diffusion strategies remained the predominant technique for the use in adaptive networks, although incremental and consensus strategies continued to be considered over the following years [94–99]. For this reason, in this paper, we focus mainly on diffusion techniques. An excellent, in-depth comparison of incremental, consensus, and diffusion strategies can be seen in [3].

Thus, by endowing the networks with the ability to adapt to the signals of interest at hand, and by adopting diffusion strategies to disseminate information throughout the network, the backbones of adaptive diffusion networks were laid out [4–6, 60, 92], and they began to popularize.

2.3. Diffusion strategies consolidate and are extended

Diffusion strategies soon became the most widely used protocol in adaptive sensor networks [1–6]. From this point on, much of the research done on adaptive sensor networks focused on them – either expanding their methods to other fields of the distributed signal processing area, or attempting to address some of their limitations. For example, despite their advantages over centralized approaches and other distributed schemes, diffusion networks may require a high number of communications. For this reason, over the 2010s decade, several techniques were proposed to reduce the energy consumption associated with the communication processes. Some aimed to reduce the amount of information sent in each transmission [100–108], whereas others shut links off according to selective communication policies [109–117]. Last but not least, there is a group of solutions known as *censoring* techniques. They seek to cut the transmission from certain nodes to any of their neighbors [118–128], hence allowing censored nodes to turn their transmitters off. This saves energy and reduces the amount of information used in the processing [120, 121]. The search for efficient mechanisms that reduce the energy consumption associated with the communication between nodes while preserving the performance of adaptive diffusion networks is a topic that continues to inspire the distributed signal processing community to this day [106–108, 124, 125, 127–130].

Meanwhile, other works sought to investigate the impact of some forms of uncertainty on the behavior of adaptive diffusion networks, which may be inevitable in practical implementations. These uncertainties included changes in the network topology during the operation of the adaptive algorithms [109, 111], random link failures, random data arrival times, and agents turning on and off randomly [131–133]. Since these effects are asynchronous in nature due to their randomness, networks where these occurrences can be observed were named as *asynchronous networks* in the literature [1, 3, 131–133].

Furthermore, from the mid 2010's onward, extensions of adaptive diffusion networks began to be proposed. For instance, the topic of *multitask networks* received considerable attention from 2014 onward [7–18]. These networks can be seen as a generalization of the solutions that had been previously studied, which, in the multitask literature, are named as *single-task networks* for distinction. In contrast with single-task solutions, the multitask approach considers a network of nodes that do not share the same exact objective, but rather have overlapping (albeit distinct) estimation interests [10]. Hence, there are multiple parameter vectors to be inferred simultaneously and in a collaborative manner [9]. This situation is oftentimes observed in distributed temperature estimation problems where the parameters that determine the evolution of temperature over time vary in space [9]. Other examples of applications include target tracking problems in which there are multiple targets to be tracked simultaneously, and spectrum sensing over cognitive radio networks, for instance [9, 10].

Around the same period, diffusion networks with nonlinear processing began to be proposed. In particular, solutions based on *kernel methods* became popular in the diffusion networks literature [19–23]. Some of the first solutions of this kind can be viewed as an extension of kernel adaptive filters [134–136] to the con-

text of diffusion networks [19–21]. Kernel adaptive filters map the input signal to a vector space of higher dimension (typically through a nonlinear mapping function), where a linear adaptive filter is employed. Thus, nonlinear filtering can be achieved by the use of linear techniques on the mapped signals. Moreover, by making use of the kernel trick [134, 136], kernel adaptive filters typically do not calculate the mapping of the inputs explicitly, which saves computational power. Evidently, the utility of these solutions stems from the fact that nonlinear functions are oftentimes better models for physical phenomena than linear ones. This also holds in many sensor networks applications. One possible example is that of environmental monitoring, where a network of sensors is deployed to observe an oftentimes nonlinear diffusion field [137]. Thus, kernel-based adaptive networks seek to address the needs of applications where the signal processing must be simultaneously nonlinear and carried out in a distributed manner [19–23].

The aforementioned research topics are not isolated from each other. For example, there have been proposals of multitask kernel networks [138] and multitask asynchronous networks [139] in the literature. Finally, one cannot review the mid 2010's in the adaptive diffusion networks literature and not mention one of the main milestones achieved by the area around this time. In 2015, the first large-scale journal dedicated specifically to the subject was launched: the *IEEE Transactions on Signal and Information Processing over Networks* [140], which covers topics such as distributed algorithms for filtering, detection, estimation, adaptation and learning, model selection, data fusion, diffusion or evolution of information over networks, applications of distributed signal processing, among others.

2.4. Other Advances Span Correlated Tools

Evidently, while the theory and many solutions of adaptive diffusion networks were being developed, technological and scientific advances in other areas continued to unfold. During this period, smartphones stormed the mobile phone market and the number of mobile users across the globe skyrocketed [141], social media were developed and rapidly grew [142], big data and data mining applications became commonplace [143, 144], and the technologies that enable 5G communication networks were continuously developed [145], facilitating the implementation of the internet of things (IoT) in many applications [146, 147]. Moreover, in this period, the usage of WSN's in practical applications was heavily explored. Examples include, e.g., wireless body area networks (WBANs) [148, 149], which rose as appealing solutions in, e.g., healthcare and security applications [150]. Among these, one could mention for instance the wireless electroencephalography (EEG) sensor networks (WESNs), which were proposed for neuromonitoring applications [151]. Furthermore, WSN's were also employed in agriculture [152], speech, video and image enhancement, spectrum sensing, power system state estimation, among many other applications (see, e.g., [153] and the references therein).

Due to the inherent connectivity between the elements that comprise these technologies, they share an interesting trait: they can be well represented by graphs. For example, in the context of social networks, each user can be represented by a node, and the existence of an edge connecting two nodes can be used to

indicate that they are friends on that platform [154]. Analogously, each end user in a communication network can be represented by a node, and an edge linking two users could indicate that they are geographically close to one another, and, therefore, the power of the signal received by them must be similar [155].

As a result, a field of research emerged and quickly attracted widespread attention in the signal processing community: the field of graph signal processing [28–31, 154–161]. Broadly speaking, GSP techniques seek to explore the relationships between elements of a networked system and from (potentially partial) observations collected by them to extract useful information. In some cases, the goal is to reconstruct a certain signal defined over the graph (also known as graph signal in the literature) [30, 31, 155], in others, it is to infer the underlying graph topology from the resulting graph signal [157–159], and in others, it is even to identify the system that dictates the dynamics of a signal defined over a spatially distributed set of sensors [28, 29]. The field of GSP has grown to become a vast research area, with many variations and nuances. Since in this paper our focus lies on adaptive diffusion networks, we restrict our review to the topics of GSP that have a clear interface with those networks.

For example, the solutions aimed at system identification for graph signals usually assume that there is a graph shift operator [154] that influences the behavior of a signal of reference over time. Hence, both the topology of the network and the temporal factor play a major role in how the reference signal unfolds at each node. For this reason, this approach has enjoyed success in meteorology applications [28]. Furthermore, these solutions also took heavy inspiration from adaptive diffusion networks. In fact, they explicitly sought to develop diffusion versions of classical adaptive filtering algorithms for use in GSP [28, 29]. Consequently, these solutions bear striking similarities to many previous diffusion adaptive algorithms, despite the obvious differences that arise from the GSP context. The use of adaptive diffusion strategies in the GSP field represents yet another reason why these strategies are as relevant today as they have ever been. In fact, they have become commonplace in papers of the area [26, 28–31]. This is only one of the factors that makes them such an interesting subject.

More recently, some strategies have been proposed for the distributed training of neural networks [162–165]. The reasoning behind this concept is that, even when the training of neural networks is done offline, it may be advantageous to carry it out in a distributed manner if the scale of the problem is too large or if there are privacy concerns involved. In the former case, hardware and software limitations can make it infeasible for one single device to handle massive amounts of data. At the same time, the growing availability of devices offers an opportunity for distributing the learning task among many agents. In the latter case, sending a great deal of raw sensitive information to a single processing unit may be undesirable due to security concerns. Examples include, e.g., data related to personal behaviors or medical conditions [163, 166]. Yet, we would still like to train the network using all the data available in these cases. Furthermore, it has been shown that diffusion strategies can escape from saddle points in non-convex optimization problems [167, 168]. Since neural networks oftentimes struggle with the existence of local minima in their respective loss functions, this

is another reason why the topic has a great potential to be explored in future research.

These are some of the primary concerns of the field of *Federated Learning* [166, 169–173], a promising and relatively young research area – its name was coined in a 2016 paper [170] –, which emerged in the wake of the same technological developments previously mentioned, i.e., the increasing ubiquity of smart devices and the auspicious promise of IoT. In the past few years, it has attracted a growing interest in the machine learning and signal processing communities. In Fig. 2, we depict a timeline with some of the main developments in wireless communication technology, network applications, and some milestones for the adaptive diffusion networks and related fields.

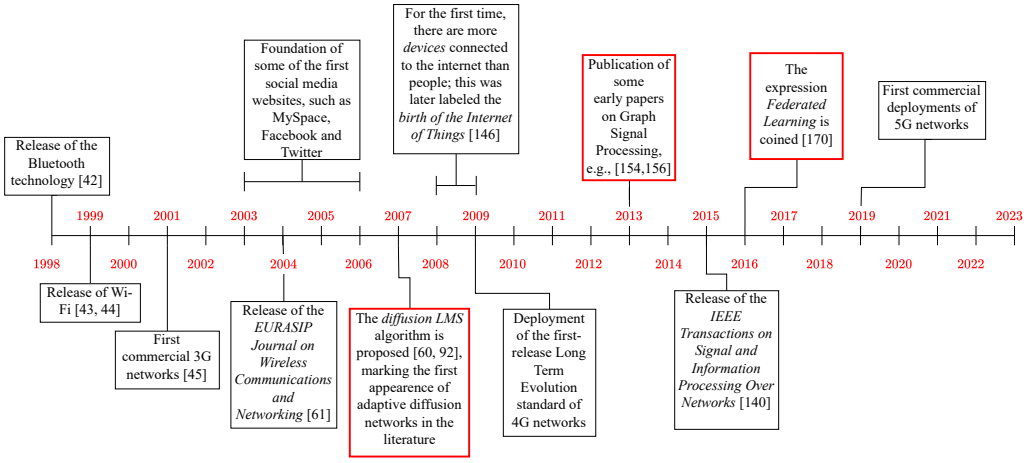

Figure 2: A timeline with several of the main events related to wireless communication technology and network applications, as well as some milestones of the adaptive diffusion network and graph signal processing fields.

Overall, many research topics remain open in the area of distributed learning. Many possible applications are becoming more viable, and the growing potential of the correlated fields is deemed promising. Efficient techniques for reducing the communication costs in adaptive diffusion networks and for sampling over diffusion GSP solutions continue to inspire the search for novel solutions, as well as the extension of these networks to more complex and challenging scenarios. For this reason, it is only fair we devote more time and space to their understanding.

Notation. We use lowercase normal font letters to denote scalars, boldface lowercase letters for vectors, boldface uppercase letters for matrices, and calligraphic fonts for sets. To simplify the arguments, we assume real data throughout the paper. In Tab. 1, we provide a summary of the notation used in the paper.

Table 1: Summary of the notation used in the paper.

Symbol	Meaning
$[\mathbf{X}]_{ij}$	(i,j)-th entry of the matrix $\mathbf{X}$
$(\cdot)^{\mathrm{T}}$	Transposition operator
$(\cdot)^*$	Complex conjugate
$\mathbf{I}_N$	$N \times N$ identity matrix
$\mathbf{1}_N$	$N \times 1$ vector whose entries are all equal to 1
$\mathbf{0}_N$	$N \times 1$ vector whose entries are all equal to 0
$\mathrm{E}\{\cdot\}$	Mathematical expectation
$\|\cdot\|$	Euclidean norm
$\lvert\cdot\rvert$	Cardinality if the argument is a set, or absolute value if the argument is a scalar
$\mathcal{X}/\mathcal{Y}$	Difference between the sets $\mathcal{X}$ and $\mathcal{Y}$
$\exp(\cdot)$	Exponential function

3. Adaptive Diffusion Networks

Let us consider a collection of V labeled *nodes* or *agents*. As illustrated in Fig. 3, in most adaptive diffusion network applications we consider that each node k, $k = 1, \cdots, V$, has access at each time instant n to an input signal $u_k(n)$ and to a desired signal $d_k(n)$, modeled as [1–5]

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

where $\mathbf{w}^{\mathrm{o}}$ and $\mathbf{u}_k(n)$ are M-length column vectors that represent respectively an unknown system and the input vector at node k. In particular, $\mathbf{u}_k(n)$ oftentimes represents a regressor vector formed by the last M samples of the input signal $u_k(n)$, i.e., $\mathbf{u}_k(n) = [u_k(n)\ u_k(n-1)\ \cdots\ u_k(n-M+1)]$, although this is not necessarily the case in every application [1–3]. Furthermore, $v_k(n)$ is the measurement noise at node k, which is assumed to be independent and identically distributed (iid), zero-mean with variance $\sigma_{v_k}^2$, and independent of the other variables.

The goal of the agents is to obtain an estimate $\mathbf{w} = [w_0, \cdots, w_{M-1}]^{\mathrm{T}}$ of $\mathbf{w}^{\mathrm{o}}$. To this end, it is customary to introduce a cost function J such that [2]

$$\mathbf{w}^{\mathrm{o}} = \arg\min_{\mathbf{w}} J(\mathbf{w}). \tag{2}$$

For this reason, the coefficient vector $\mathbf{w}^{\mathrm{o}}$ is oftentimes referred to as the "optimal system" in the literature. One of the most widely adopted cost functions is the mean-squared error (MSE). In the case of single-agent

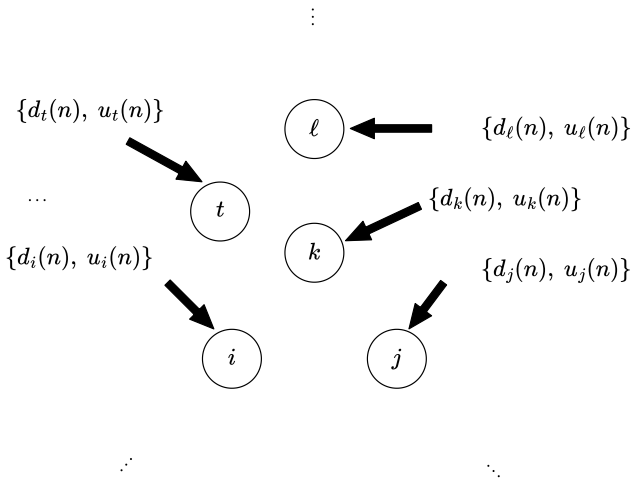

Figure 3: A collection of nodes with their respective desired signals d and input signals u.

learning, this cost function is given by $J(\mathbf{w}) \triangleq \mathrm{E}\{[d(n) - \mathbf{u}^{\mathrm{T}}(n)\mathbf{w}]^2\}$, where we have dropped the index due to the existence of only one agent, and is used in the derivation of many classical adaptive filtering algorithms, such as those of the LMS and RLS types, for example. Since we are interested in working with the multiple agents that form the network, we introduce a local cost function $J_k(\mathbf{w})$ for each node k, which is then given by [2, 74]

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

The first idea that may come to mind in face of (3) is to minimize $J_k(\mathbf{w})$ in each node k by implementing a stochastic gradient descent algorithm in each agent using only the locally available information. Following this approach, if we denote the gradient of J_k with respect to $\mathbf{w}$ by [75]

$$\nabla_{\mathbf{w}} J_k(\mathbf{w}) \triangleq \frac{\partial J_k(\mathbf{w})}{\partial \mathbf{w}} = \left[\frac{\partial J_k(\mathbf{w})}{\partial w_0}, \cdots, \frac{\partial J_k(\mathbf{w})}{\partial w_{M-1}} \right]^{\mathrm{T}}, \tag{4}$$

and its approximate value evaluated at node k by $\widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)]$, we get, for $k = 1, \cdots, V$, [2]

$$\mathbf{w}_k(n) = \mathbf{w}_k(n-1) - \mu_k \widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)], \tag{5}$$

where μ_k is a step size and $\mathbf{w}_k(n)$ the coefficient vector at node k. The expression for $\widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)]$ depends on the approximations made in the stochastic descent algorithm. For example, following an LMS approach, (5) could be recast as [2, 74]

$$\mathbf{w}_k(n) = \mathbf{w}_k(n-1) + \mu_k e_k(n) \mathbf{u}_k(n), \tag{6}$$

where

$$e_k(n) = d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_k(n-1) \tag{7}$$

is the estimation error.

We should notice that in (5) we are not assuming any form of communication between the different nodes. In fact, each node is acting as an individual adaptive filter, and they are not working together as a network. For this reason, this approach is referred to as the *non-cooperative* setup in the literature [1–5]. Although this solution can be effective in many situations, if the nodes have the ability to communicate with one another, not doing so can be seen as a waste of potential. After all, it is expected that the sharing of data between the nodes should improve their performances, since this means that more information is being used in the update process.

Hence, we now seek to analyze configurations in which the nodes cooperate. To do so, we describe the objective of the network as a whole by the optimization problem [1–5]

$$\min_{\mathbf{w}} J_{\text{global}}(\mathbf{w}) = \min_{\mathbf{w}} \sum_{k=1}^{V} J_k(\mathbf{w}). \tag{8}$$

Let us now assume that the nodes can communicate with one another through a given topology. For any node k, the subset of nodes it can communicate with (including node k itself) is called its *neighborhood*, which is denoted by $\mathcal{N}_k$. Furthermore, the nodes that it can communicate with are called its *neighbors*. An example is depicted in Fig. 4. It is important to note that a common assumption in the literature is that the network is *strongly connected*, i.e., given any pair of nodes k and i, there is a path from node k to node i and vice-versa, and at least one node has a self-loop [1–3].

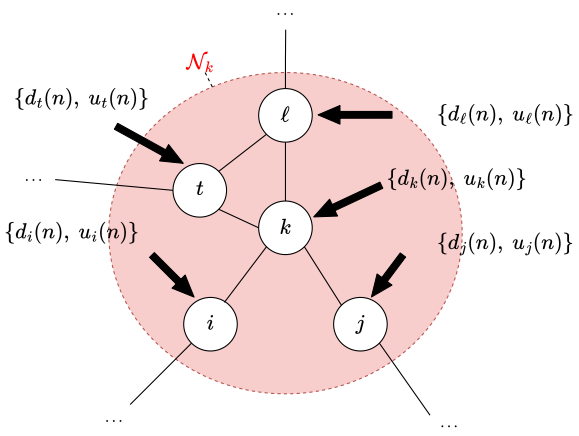

Figure 4: A network of nodes with their respective desired signals d and input signals u and a predefined topology. In particular, the neighborhood of node k is highlighted.

In this scenario, if we employ a stochastic gradient approach to solve (8), the cooperation between the

nodes can be enforced by adopting [1–3]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) - \mu_k \widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)] & \text{(9a)} \\ \mathbf{w}_k(n) = \sum\limits_{i \in \mathcal{N}_k} c_{ik} \boldsymbol{\psi}_i(n), & \text{(9b)} \end{cases}$$

where $\{c_{ik}\}$ are convex combination weights satisfying the following set of conditions:

$$\begin{cases} c_{ik} = 0 \text{ if } i \notin \mathcal{N}_k & \text{(10a)} \\ \sum\limits_{i \in \mathcal{N}_k} c_{ik} = 1 & \text{(10b)} \\ c_{ik} \geqslant 0, \ \forall i,k. & \text{(10c)} \end{cases}$$

Evidently, (10a) incorporates the constraints related to the network topology. On the other hand, (10b) is used to ensure that the combined estimates $\mathbf{w}_k(n)$ are unbiased estimators of $\mathbf{w}^o$.

Eqs. (9a) and (9b) are known as the *adaptation* and *combination* steps, respectively of adaptive diffusion networks. The order of (9a) and (9b) characterize a configuration that is known as *adapt-then-combine* (ATC) in the literature [1–6]. One could also switch their order: in this case, the adaptation step precedes the combination one, leading to the *combine-then-adapt* (CTA) configuration, given by

$$\begin{cases} \mathbf{w}_k(n-1) = \sum\limits_{i \in \mathcal{N}_k} c_{ik} \boldsymbol{\psi}_i(n-1) & \text{(11a)} \\ \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) - \mu_k \widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)]. & \text{(11b)} \end{cases}$$

It is worth mentioning that the ATC protocol is the one most commonly adopted in the literature [100, 102, 106, 109, 111, 113–115, 118–121]. For example, the ATC diffusion LMS (dLMS) algorithm is given by [1–5]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) + \mu_k e_k(n) \mathbf{u}_k(n) & \text{(12a)} \\ \mathbf{w}_k(n) = \sum\limits_{i \in \mathcal{N}_k} c_{ik} \boldsymbol{\psi}_i(n). & \text{(12b)} \end{cases}$$

For convenience, we also provide a pseudo-code representation of the ATC dLMS as Algorithm 1 next.

Besides the dLMS algorithm, diffusion versions of other adaptive algorithms were proposed in the literature, such as diffusion RLS [6, 175–177], diffusion Normalized LMS (dNLMS) [109, 178, 179], the diffusion Affine Projection Algorithm (dAPA) [180], among others [96, 110].

Eqs. (11) and (9) are the basis for diffusion adaptive networks, and form the groundwork for many diffusion solutions that were proposed over the following years. They allow us to attain the good performance of centralized solutions in a fully distributed way, without many of their limitations. The difference between the diffusion strategies and the non-cooperative approach described by (5) lies in the existence of the combination step, which allows the knowledge gained by a node to disseminate throughout the whole

Algorithm 1 The ATC dLMS Algorithm of Eq. (12).

% Initialization - for each node $k = 1, \cdots, V$, select a step size μ_k and combination weights c_{ik} satisfying (10) for $i = 1, \cdots, V$, and set $\boldsymbol{\psi}_k(0) \leftarrow \mathbf{0}_M$ and $\mathbf{w}_k(0) \leftarrow \mathbf{0}_M$
for $n = 1, 2, \cdots$ **do**
    % Adaptation Step
    for $k = 1, \cdots, V$ **do**
        Update $\mathbf{u}_k(n)$
        % Calculating the estimation error:
        $e_k(n) \leftarrow d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_k(n-1)$
        % Adapting the local estimate $\boldsymbol{\psi}_k(n)$:
        $\boldsymbol{\psi}_k(n) \leftarrow \mathbf{w}_k(n-1) + \mu_k e_k(n)\mathbf{u}_k(n)$
    end for
    % The nodes transmit their local estimates $\boldsymbol{\psi}$ to their neighbors
    % Combination Step
    for $k = 1, \cdots, V$ **do**
        % Forming the combined estimate $\mathbf{w}_k(n)$:
        $\mathbf{w}_k(n) \leftarrow \mathbf{0}_M$
        for $i \in \mathcal{N}_k$ **do**
            $\mathbf{w}_k(n) \leftarrow \mathbf{w}_k(n) + c_{ik}\boldsymbol{\psi}_i(n)$
        end for
    end for
end for

network, enabling a better performance. It is worth noting, nonetheless, that the potentially high number of communication processes between the nodes can pose a challenge in terms of bandwidth requirements and especially energy consumption – which will be addressed in Sec. 4.

There is a generalized form of the diffusion strategies of (9) and (11) that is worth mentioning. In addition to the exchange of local estimates $\boldsymbol{\psi}_i$ prior to the combination step, we could also allow each node to share its approximate gradient $\widehat{\nabla}_{\mathbf{w}} J_i[\mathbf{w}_i(n–1)]$ before the adaptation step. Then, each node k can perform a combination of these approximate gradients in the update of $\boldsymbol{\psi}_k(n)$. Following the ATC configuration, this leads to [1]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) - \mu_k \sum\limits_{i \in \mathcal{N}_k} b_{ik} \widehat{\nabla}_{\mathbf{w}} J_i[\mathbf{w}_k(n-1)] & \text{(13a)} \\ \mathbf{w}_k(n) = \sum\limits_{i \in \mathcal{N}_k} c_{ik}\boldsymbol{\psi}_i(n), & \text{(13b)} \end{cases}$$

where b_{ik} are combination weights satisfying (10). Evidently, an analogous version of (13) can be obtained for CTA. These schemes are sometimes referred to as *diffusion strategies with enlarged cooperation* [1, 3]. It should be noted that (13) implies that the nodes communicate twice per iteration: first, before (13a) is executed, each node i must share its local data $\{d_i(n),\mathbf{u}_i(n)\}$ with its neighbors to enable the calculation of $\widehat{\nabla}_{\mathbf{w}} J_i[\mathbf{w}_k(n-1)]$. After (13a) is carried out, they must then share their local estimates $\boldsymbol{\psi}$ before (13b) is run. Since this can be very costly from an energetic point of view, the diffusion strategies of the forms (9) and (11) are more common in the literature, see e.g. [100, 102, 106, 109, 111, 113–115, 118–121].

It can be shown that the stability of the standard diffusion LMS algorithm is ensured in the mean sense if each step size μ_k satisfies [2, 5]

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(\mathbf{R}_{u_k}\right)}, \tag{14}$$

where $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue, and $\mathbf{R}_{u_k} \triangleq \mathrm{E}\{\mathbf{u}_k(n)\mathbf{u}_k^{\mathrm{T}}(n)\}$ is the autocorrelation matrix of the input signal at node k. There is a clear analogy between (14) and the stability condition for a single LMS adaptive filter (see, e.g., [74, 75]). In fact, adaptive diffusion networks inherit many of the traits of the adaptive filters that inspired them. Typically, the greater the step sizes μ_k, the faster the convergence rate of the network in the transient phase. After a while, the performance of the network stabilizes and a steady state is achieved. In general, the greater the step sizes, the worse the steady-state performance [1–3]. This will become clear in Sec. 4.2, when we introduce some performance indicators and present theoretical predictions for them based on the step sizes, filter length M, and combination weights. Moreover, due to their adaptive nature, the algorithms of Eqs. (9) and (11) are able to track changes in the environment. In this regard, it is interesting to notice that this is only possible if the step sizes do not diminish indefinitely over time – which was a trait of the first consensus strategies that proved detrimental to their usage in adaptive networks.

As is known since the inception of the first adaptive filtering algorithms [74, 75], adaptation is a powerful feature for online learning. The ability to track changes in the environment, along with the fact that we do not need to have prior knowledge on the statistics of the signals involved, is one of the many features that made the extension of adaptive signal processing techniques to WSN's so appealing.

Besides the standard diffusion strategies shown so far, modified versions of these schemes continued to be proposed over the years with different purposes. Next, we mention a few of these modified diffusion protocols. For instance, a group of adaptive algorithms for diffusion networks was proposed in [181]. These algorithms sought to improve the robustness of the networks in a scenario where nodes can fail and the data collected are very noisy. To this end, each node projects its combined estimate onto a hyperslab or a halfspace, and then uses the latest q projections in the adaptation. By adopting cost functions common in the robust statistics field, such as a modified version of the Huber cost [182], the combined estimates can be projected on halfspaces that simultaneously reduce the MSE and mitigate the effect of outliers in the

data, which may be present due to malfunctioning nodes or noisy measurements. Other robust diffusion algorithms based on the Huber cost function were also proposed in [183, 184]. Furthermore, several robust diffusion techniques have also been proposed to deal with errors-in-variable (EIV) models based on the total least-squares approach [185]. EIV models can be used to represent, e.g., situations in which the input signal is subject to noise, as well as the desired signal. Representative examples include, e.g., the solutions proposed in [186–190]. The interested reader can find a review focused primarily on robust diffusion solutions in [39].

In addition to the distributed estimation algorithms that have been discussed so far, diffusion-based detection solutions have also been proposed [191–194]. In these cases, the goal of each agent in the network is to provide a decision about the state of a system, which can vary over time. It was shown in [191] that, by adopting a similar strategy to the diffusion schemes for estimation, one can achieve the same performance of a centralized stochastic-gradient approach in terms of detection error exponents. Thus, despite their peculiarities, these solutions consist in i) updating the estimates about the current state of the system locally at each node, and ii) enabling the agents to exchange and combine their estimates [191–194].

Furthermore, one could mention the *sparse diffusion solutions* [177, 195–198], which aim to take advantage of the inherent sparsity of many signals and system models, i.e., the presence of only a small number of nonzero entries. When the optimal system $\mathbf{w}^{o}$ is sparse, these solutions present a faster convergence rate and can outperform the standard diffusion solutions of (11) and (9).

Lastly, it is worth noting that, in the solutions examined thus far, the raw data are distributed among the nodes, since each node k has access to $d_k(n)$ and $u_k(n)$. However, other solutions have been recently proposed in which certain features are first extracted from the available data, and it is the information from these features that is distributed throughout the network, rather than the unprocessed data. Thus, if we aggregated the information from the features into a feature vector, each node would only have access to a certain block of entries in that vector. This approach can be adopted in scenarios in which there are privacy concerns, for example, or in which the data are already collected in a distributed manner, as is the case in spatial filters and sensor array processing. Examples of these solutions include, e.g., [199–201]. In particular, it is worth noting that in [199] a diffusion-based approach is adopted to address this scenario in which the features are distributed among the nodes, in a similar manner to Eq. (9).

3.1. The selection of the combination weights and the steady-state performance

There are several possible rules for the selection of $\{c_{ik}\}$, which can play a significant role in the behavior of diffusion adaptive networks. Among the most widely adopted strategies found in the literature are the Uniform or Averaging [202], Laplacian [67], Metropolis [67, 203, 204], Maximum-Degree [67] and Relative Degree [6] rules. For ease of reference, they are summarized as rules 1–5 in Table 2.

Although it is not the goal of this paper to provide an in-depth theoretical analysis of each solution that we will review, it may be interesting to compare the performances of the non-cooperative and diffusion

Table 2: Summary of the some rules for the selection of the combination weights most widely adopted in the literature.

Name	Equations
1) Uniform or Average [202]	$c_{ik} = \begin{cases} \frac{1}{\lvert\mathcal{N}_k\rvert}, \text{ if } i \in \mathcal{N}_k \\ 0, \quad \text{otherwise} \end{cases}$
2) Metropolis [67, 203, 204]	$c_{ik} = \begin{cases} \frac{1}{\max\{\lvert\mathcal{N}_k\rvert,\lvert\mathcal{N}_i\rvert\}}, \text{ if } i \in \mathcal{N}_k/\{k\} \\ 1 - \sum_{i\in\mathcal{N}_k} c_{ik}, \text{ if } i = k \\ 0, \text{ otherwise} \end{cases}$
3) Relative-Degree [6]	$c_{ik} = \begin{cases} \frac{\lvert\mathcal{N}_l\rvert}{\sum_{i\in\mathcal{N}_k}\lvert\mathcal{N}_i\rvert}, \text{ if } i \in \mathcal{N}_k \\ 0, \text{ otherwise} \end{cases}$
4) Laplacian [67]	$c_{ik} = \begin{cases} \frac{1}{\max_{i=1,\cdots,V}\lvert\mathcal{N}_i\rvert}, \text{ if } i \in \mathcal{N}_k/\{k\} \\ 1 - \frac{\lvert\mathcal{N}_k\rvert - 1}{\max_{i=1,\cdots,V}\lvert\mathcal{N}_i\rvert}, \text{ if } i = k \\ 0, \text{ otherwise} \end{cases}$
5) Maximum-Degree [82]	$c_{ik} = \begin{cases} \frac{1}{V}, \text{ if } i \in \mathcal{N}_k/\{k\} \\ 1 - \frac{\lvert\mathcal{N}_k\rvert - 1}{V}, \text{ if } i = k \\ 0, \text{ otherwise} \end{cases}$
6) Hastings [205]	$c_{ik} = \begin{cases} \frac{\sigma^2_{v_k}}{\max\{\lvert\mathcal{N}_k\rvert\sigma^2_{v_k},\lvert\mathcal{N}_i\rvert\sigma^2_{v_i}\}}, \text{ if } i \in \mathcal{N}_k/\{k\} \\ 0, \text{ if } i \notin \mathcal{N}_k \\ 1 - \sum_{i\in\mathcal{N}_k} c_{ik}, \text{ if } i = k \end{cases}$
7) Relative Variance [206–208]	$c_{ik} = \begin{cases} \frac{(\sigma^2_{v_i})^{-1}}{\sum_{i\in\mathcal{N}_k}(\sigma^2_{v_i})^{-1}}, \text{ if } i \in \mathcal{N}_k \\ 0, \text{ otherwise} \end{cases}$
8) Adaptive Combination Weights (ACW) [206]	$\widehat{\sigma}^2_{ik}(n) = (1-\nu_k)\widehat{\sigma}^2_{ik}(n-1) + \nu_k\lVert\boldsymbol{\psi}_i(n) - \mathbf{w}_k(n-1)\rVert^2$ $c_{ik}(n) = \begin{cases} \frac{[\widehat{\sigma}^2_{ik}(n)]^{-1}}{\sum_{i\in\mathcal{N}_k}[\widehat{\sigma}^2_{ik}(n)]^{-1}}, \text{ if } i \in \mathcal{N}_k \\ 0, \text{ otherwise} \end{cases}$

approaches. Thus, we will present some of the main results of the analysis on these solutions without extensively deriving them. For a detailed theoretical study of the techniques addressed in this paper, we suggest the reader refers to [1–3] and references therein. For simplicity, we will focus on implementations of

the LMS type in the form of (6). Furthermore, we limit our attention to a performance indicator commonly adopted in the adaptive networks literature, the network mean-square deviation (NMSD), given by [1]

$$\text{NMSD}(n) = \frac{1}{V}\sum_{k=1}^{V} \text{E}\{\|\widetilde{\mathbf{w}}_k(n)\|^2\}, \tag{15}$$

where we have introduced the weight-error vector

$$\widetilde{\mathbf{w}}_k(n) \triangleq \mathbf{w}^{\text{o}} - \mathbf{w}_k(n). \tag{16}$$

Since the non-cooperative case (5) corresponds to a situation in which there are V adaptive filters running separately, the NMSD defined by (15) will be equal to the average of the MSDs of each individual filter. This is reasonable, since they act as isolated agents. For sufficiently small step sizes μ, it can be shown that the MSD of a single LMS adaptive filter is given by [74, 75]

$$\text{MSD}_k(\infty) = \frac{\mu_k M}{2}\sigma_{v_k}^2, \tag{17}$$

where $\text{MSD}_k(n) \triangleq \text{E}\{\|\widetilde{\mathbf{w}}_k(n)\|^2\}$. For simplicity, let us now assume that the step sizes μ_k are the same for every node k, i.e., $\mu_1 = \cdots = \mu_V = \mu$, since this will make the comparison with the other strategies easier to understand. In this case, summing the right-hand side of (17) for $k = 1, \cdots, V$ leads to [1–3]

$$\text{NMSD}_{\text{ncoop}}(\infty) = \frac{\mu M}{2}\left(\frac{1}{V}\sum_{k=1}^{V}\sigma_{v_k}^2\right). \tag{18}$$

In contrast, assuming that the input signals have the same positive definite autocorrelation matrix $\mathbf{R}_{u_k}$ at every node k, i.e., $\mathbf{R}_{u_k} = \mathbf{R}_u > 0$ for $k = 1, \cdots, V$, it can be shown that, when the diffusion LMS algorithm is implemented with Uniform or Metropolis combination weights, its steady-state NMSD can be well approximated by [1, 3]

$$\text{NMSD}_{\text{diff.}}^{\text{unif.}}(\infty) \approx \text{NMSD}_{\text{diff.}}^{\text{metr.}}(\infty) \approx \frac{\mu M}{2V}\left(\frac{1}{V}\sum_{k=1}^{V}\sigma_{v_k}^2\right). \tag{19}$$

We remark that the difference between Eqs. (18) and (19) lies in the presence of a factor of V in the denominator of the fraction outside the parentheses in (19), which is absent in (18). Hence, in comparison with the non-cooperative scheme, the NMSD is reduced by a factor of $\frac{1}{V}$. Furthermore, if $c_{ik} = c_{ki}$, for any k and i, it can be shown that the steady-state NMSD of the diffusion strategies is similar to that of the centralized and incremental approaches [1, 3, 205].

As can be seen from Table 2, the aforementioned rules use information from the network topology to determine the combination weights. As evidenced by (19), this can already lead to a significant improvement in performance in comparison with the non-cooperative scheme. However, the network topology is evidently not the only factor that influences the performance of the diffusion algorithms. If we incorporate more

information into the $\{c_{ik}\}$, we should be able to assign greater weights to the nodes that are somehow expected to perform better. For example, it is intuitive that we should privilege nodes that are subject to lower noise powers in the combination step, since they are expected to present the more refined local estimates $\boldsymbol{\psi}$. For this reason, some policies that incorporate information from the noise profile across the network were proposed. One possible example is the Hastings rule [1, 3, 205]. Assuming that the noise variances $\sigma_{v_1}^2, \cdots, \sigma_{v_V}^2$ are not all equal, i.e., at least one $\sigma_{v_k}^2$ is different from the others, the Hastings weights can be obtained by seeking to minimize the Network Excess MSE (NEMSE), which is given by [1, 3, 205]

$$\text{NEMSE} = \lim_{n\to\infty} \sum_{k=1}^{V} \text{E}\{[e_k(n) - v_k(n)]^2\}. \tag{20}$$

Incorporating the constraints of (10), this leads to an optimization problem that can be solved using a procedure proposed by Hastings [204, 209], thus yielding Rule 6) of Table 2 [205]. It is easy to see from the table that the Hastings rule assigns greater weights to the least noisy neighbors, which is in accordance with our expectations. If all the nodes are subject to the same noise power, i.e., $\sigma_{v_k}^2 = \sigma_v^2$ for $k = 1, \cdots, V$, the Hastings weights coincide with the Metropolis ones.

It can be shown that, if the Hastings weights are used, the diffusion LMS generally outperforms the same strategy with Uniform or Metropolis weights, i.e., [1, 3, 205]

$$\text{NMSD}_{\text{diff.}}^{\text{Hastings}}(\infty) < \text{NMSD}_{\text{diff.}}^{\text{unif.}}(\infty) < \text{NMSD}_{\text{ncoop}}(\infty). \tag{21}$$

One important aspect of the Hastings rule is that it requires the *a priori* knowledge of the noise variances across all nodes, i.e., $\sigma_{v_1}^2, \cdots, \sigma_{v_V}^2$. However, this information may not always be known beforehand. For this reason, several *adaptive combination weights* (ACW) algorithms were proposed for the selection of the combination weights in an adaptive manner, while seeking to incorporate information from the noise profile in the network. Some early examples of this strategy were proposed in [174] and [206].

For instance, the algorithm of [206], which was later analyzed in more detail in [207], seeks to minimize $\text{Tr}(\mathbf{C}^{\text{T}}\mathbf{R}_v\mathbf{C})$ subject to (10), where $\text{Tr}(\cdot)$ denotes the trace of a matrix, $\mathbf{C}$ is a $V \times V$ matrix aggregating the combination weights, such that $[\mathbf{C}]_{ik} = c_{ik}$, and $\mathbf{R}_v$ is a diagonal matrix whose entries are equal to $\sigma_{v_1}^2, \cdots, \sigma_{v_V}^2$. Hence, it incorporates information from the noise power profile directly into the combination weights, similarly to the Hastings rule. It can be shown that the solution to this optimization problem leads to Rule 7) of Table 2 [206, 207], which is named as *relative variance rule* in the literature [206–208]. In order to eliminate the need for *a priori* knowledge of the noise powers, in the algorithm of [206], each node k implements Rule 8) of Table 2, where $0 < \nu_k < 1$ for $k = 1, \cdots, V$ [206] is a parameter to aid in the estimation of the noise variance. Oftentimes, $\nu_k = 0.2$ is adopted in the literature [118, 206, 208]. The improvement in NMSD brought by the adoption of this adaptive algorithm for the selection of the combination weights comes at the expense of a slower convergence rate in comparison with static combination rules [208, 210].

Furthermore, it has been noted that if the nodes use different step sizes μ_k, the algorithm of [206] can privilege the slower nodes in the combination step [211], and that the performance of the algorithm may be affected by the initial values assigned to each $\widehat{\sigma}_{ik}^2$ [212]. Nonetheless, due to its relative simplicity and the improvement in steady-state performance that it produces, the algorithm of [206] is widely used in the literature [113, 118, 120, 205, 207, 210, 212, 213].

Many other techniques were proposed over the years for the adaptive selection of the combination weights. As mentioned previously, in [213], the $\{c_{ik}\}$ are updated at every iteration following the APA algorithm or a LS method in order to minimize instantaneous approximations of the MSE at node k and time instant n. In [208] and [210], adaptive algorithms were proposed to take advantage of the improvement in steady-state NMSD due to the adaptive combination weights while seeking to mitigate the deterioration in the convergence rate that arises from its adoption. For this purpose, they implemented switching mechanisms that adapt the combination weights in steady state, but lead to the adoption of static rules in the transient. In [212], an algorithm for the selection of the combination weights was proposed to minimize the steady-state NMSD based on the consensus propagation technique, which is typically used for averaging results across a network of agents [214], and was shown capable of outperforming the algorithm of [206]. In [215], an algorithm for the selection of the $\{c_{ik}\}$ that takes the communication channel distortion into consideration was proposed. These are only a few of the solutions that can be found in the literature, and although it is out of the scope of this paper to provide an exhaustive list, the reader is encouraged to consult, e.g., [211, 216–218] and the references therein.

3.2. Simulations – Exploring the Theoretical Results

In this subsection, we provide some simulation results to illustrate the main arguments made so far. We consider the network of Fig. 5, which has $V = 20$ nodes, and was generated randomly according to the Erdös-Renyi model [28]. The average neighborhood size throughout the network is $\frac{1}{V}\sum_{k=1}^{V}|\mathcal{N}_k| = 4.7$. Furthermore, we consider distributed implementations of the LMS algorithm.

The simulation results were obtained over an average of 100 independent realizations in a system identification setup. We set the length of both the filter and the optimal system $\mathbf{w}^{\mathrm{o}}$ to $M = 64$. Moreover, the coefficients of $\mathbf{w}^{\mathrm{o}}$ are generated randomly following a uniform distribution in the range $[-1,1]$, and later normalized so that $\mathbf{w}^{\mathrm{o}}$ has unit norm. We consider a white Gaussian distribution for $u_k(n)$ with zero mean and unit variance. Lastly, we consider a white Gaussian distribution for $v_k(n)$ with zero mean and a different noise variance $\sigma_{v_k}^2$ for $k=1,\cdots,V$, drawn from a uniform distribution in the range $[10^{-3},10^{-2}]$. As a performance indicator, we use the NMSD given by (15).

In order to validate (18), (19), and (21), in Fig. 6, we present the steady-state NMSD (in dB) obtained in the simulations considering the non-cooperative LMS strategy of (5) and the ATC dLMS of (9), respectively. For the latter, we consider the Metropolis rule, as well as the ACW algorithm depicted in Table 2. In all

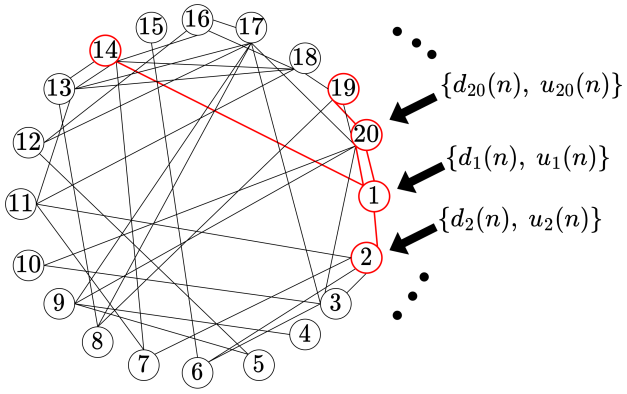

Figure 5: Network used in all simulations with synthetic data. The neighborhood of node 1 is highlighted in red.

cases, we employ the same step size $\mu_k = \mu$ for every node k, $k = 1, \cdots, 20$. Moreover, as a benchmark, we also consider a centralized solution in which a single unit employs the data from all of the nodes to adapt its model, given by

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mu \cdot \frac{1}{V} \sum_{k=1}^{V} [d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}(n-1)]\mathbf{u}_k(n).$$

The results are presented for different values of μ in the range $[10^{-4}, 10^{-2}]$. We considered a stationary environment and $120 \cdot 10^3$ iterations per realization. The results presented were obtained by calculating the average NMSD during the last $24 \cdot 10^3$ iterations of each realization, after the algorithms have converged. In addition to the simulation results, we also show the theoretical steady-state NMSD levels obtained from (18) and (19), which are presented in dashed lines.

We can observe from Fig. 6 that, for every value of μ, the non-cooperative strategy is outperformed by the cooperative schemes, as expected. Furthermore, the simulation results match well with Eqs. (18) and (19), especially for lower values of μ. This is reasonable, since (18) and (19) were obtained under the assumption of small step sizes. As μ increases, the simulation results for the non-cooperative and diffusion strategies deteriorate progressively in comparison with the theoretical predictions. Finally, we remark that the diffusion strategy with ACW clearly outperforms all other techniques for $\mu > 7 \cdot 10^{-4}$. This is in accordance with (21), if we take into account the fact that ACW aims at implementing the Hastings weights of Rule 7) of Table 2 without prior knowledge of the noise variance in the network. For lower values of μ, the difference in performance entailed by the adoption of ACW is marginal.

From Fig. 6, we can see that the adoption of small step sizes leads to lower steady-state levels of NMSD. On the other hand, this also slows down the convergence rate. To illustrate this, in Fig. 7 we show the NMSD along the iterations obtained with the same techniques used in the simulations of Fig. 6, considering two values for the step sizes: $\mu = 10^{-3}$ in Fig. 7(a) and $\mu = 10^{-2}$ in Fig. 7(b). We consider $50 \cdot 10^3$

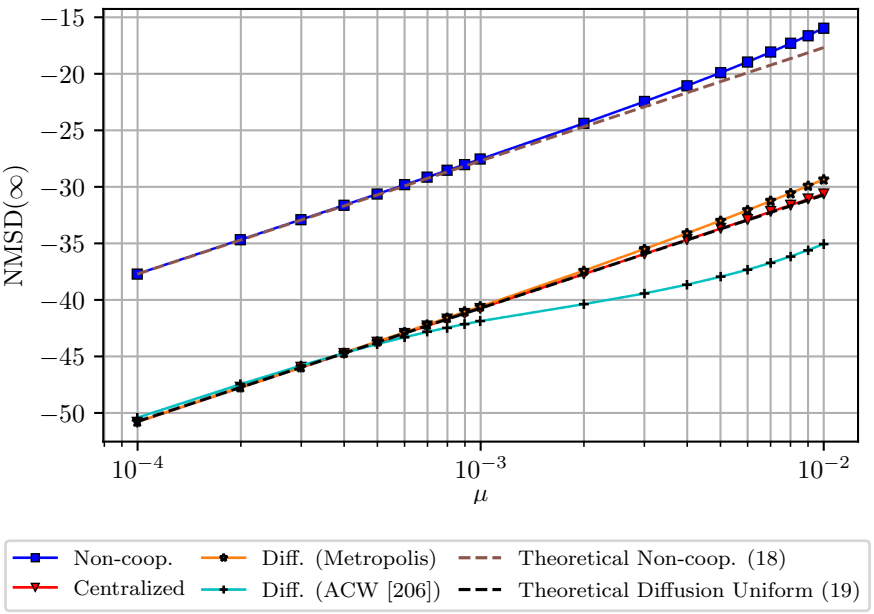

Figure 6: Steady-state NMSD (in dB) versus step size μ for various strategies, as well as the theoretical results from (18) and (19).

iterations in each realization and, to simulate an abrupt change in the environment, in the middle of each experiment we flip the vector $\mathbf{w}^o$. From Figs. 7(a) and (b) we can also see that the diffusion strategy with ACW presents a slightly slower convergence rate in comparison with the same technique with Metropolis weights. Nevertheless, with a step size of $\mu = 10^{-2}$, the diffusion algorithm with ACW achieves a noticeably lower level of steady-state NMSD in comparison with the case in which Metropolis weights are employed, as can be seen from Fig. 7(b).

4. Restricting Communication Policies

From the previous discussion, we observe from Eqs. (18), (19) and (21) that the exchange of information between the nodes can reduce the steady-state NMSD by a factor of $\frac{1}{V}$ or more. Unfortunately, allowing the permanent cooperation between the nodes can be challenging in practice. This is because energy consumption is oftentimes the most critical constraint in WSN's, and the communication between different agents is frequently the most energy-demanding task associated with the learning process. Furthermore, a high number of communication processes between the nodes demands the allocation of sufficient bandwidth for the exchange of information between them. For this reason, several solutions were proposed in the literature over the years to seek a compromise between the energy consumption and the benefits of cooperation to the performance. They allow the nodes to communicate with one another, but attempt to restrict the

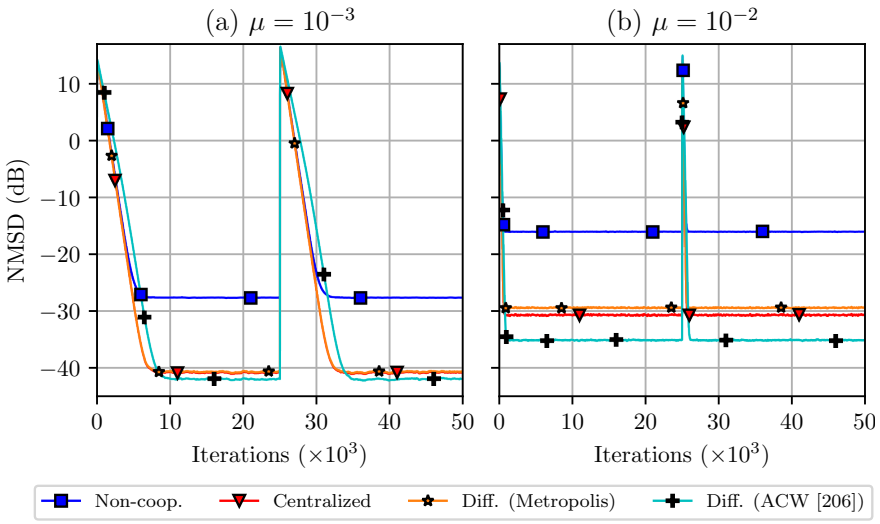

Figure 7: NMSD along the iteration obtained with various strategies, considering two different step sizes.

communication processes across the network in diffusion strategies. Although there is not an "official" distinction in the literature between these solutions, we classify them in three general categories: the *packet size reducing techniques*, the *link selection policies*, and the *censoring strategies*, which we review in detail in the following subsections. The goal of this section is not to provide an exhaustive review of every solution proposed in the literature for restricting communication policies, but rather to showcase some of the main ideas behind them, and to illustrate some of the most prominent characteristics that are common to most of these solutions.

4.1. Packet Size Reducing Techniques

The packet size reducing schemes aim to reduce the amount of information sent in each transmission by the nodes. Thus, by reducing the length of the messages, they enable a reduction in energy consumption and bandwidth usage, since the energy associated with the transmission of a package oftentimes scales linearly with its size in wireless communications [219, 220]. Examples of this strategy include, e.g., [100–108].

For instance, in [100], the authors propose to select $L < M$ entries of the local estimates $\boldsymbol{\psi}$ for transmission at every iteration. For this purpose, they introduce an $M \times M$ entry-selection matrix $\boldsymbol{\Delta}_k(n)$ in each node k, $k = 1, \cdots, V$. The matrix $\boldsymbol{\Delta}_k(n)$ is diagonal, and its elements are equal to 1 or 0. If $[\boldsymbol{\Delta}_k(n)]_{m,m} = 1$, the node k will send the m-th entry of $\boldsymbol{\psi}_k(n)$ to its neighbors. Otherwise, $[\boldsymbol{\psi}_k(n)]_m$ will not be sent, and is replaced by the corresponding entry of the local estimate of the receiving node. Thus, following an ATC strategy, (9b) is replaced with [100]

$$\mathbf{w}_k(n) = c_{kk}\boldsymbol{\psi}_k(n) + \sum_{i \in \mathcal{N}_k/\{k\}} c_{ik}\{\boldsymbol{\Delta}_i(n)\boldsymbol{\psi}_i(n) + [\mathbf{I}_M - \boldsymbol{\Delta}_i(n)]\boldsymbol{\psi}_k(n)\}. \tag{22}$$

In principle, each node should inform which entries it is sending to its neighbors. However, two methods for the selection of the matrices $\boldsymbol{\Delta}_k(n)$ are suggested in [100] that eliminate the need for this additional overhead. One of them consists in a stochastic approach in which the nodes use pseudorandom number generators (PRNGs) for the entry selection process, and share their PRNG seeds with their neighbors once, before the adaptation begins. The other approach consists in selecting the entries sequentially in a round-robin manner over the time instants n. Thus, the entries are placed in M groups of size L such that each entry is present in L groups, which are then ordered in a predetermined sequence. These groupings and their sequence are the same across all nodes, which allows them to identify which entries they are receiving [100].

As one could expect, this reduction in data exchange leads to a deterioration in steady-state performance in comparison with the standard diffusion procedure. The lower the value of L in comparison with M, the higher the steady-state NMSD becomes. Thus, there is a trade-off between energy savings and performance. This trait is not specific to the algorithm of [100], but rather common to most restrictive communication policies. The goal of these procedures is to limit the amount of information transmitted across the network while maintaining the performance of the standard diffusion schemes as much as possible.

Furthermore, many of the solutions that fit into this category seek to somehow compress the local estimates before transmitting them [103, 104, 106–108]. For example, in [107, 108], the authors consider an adapt-compress-then-combine protocol. In this solution, the compression stems from the usage of a quantized version of the local estimate. This quantized estimate is initialized in an arbitrary manner at $n = 0$, and then updated along the iterations with the usage of a certain compression operator. The choice of this operator is up to the filter designer, with possible solutions such as, e.g., [221, 222]. Moreover, the compression operator is applied to the difference between the current compressed estimate and the local uncompressed one produced by the adaptation step. Then, the output of the compression operator is scaled by a factor between zero and one, and added to the previous compressed estimate.

4.2. Link Selection Policies

As their name suggests, link selection policies seek to turn certain communication links on or off, according to predefined criteria, in order to reduce the traffic of information throughout the network [109–117]. For this, (9b) is replaced by

$$\mathbf{w}_k(n) = \sum_{i \in \mathcal{N}_k(n)} c_{ik}(n) \boldsymbol{\psi}_i(n), \tag{23}$$

in which the neighborhood $\mathcal{N}_k(n)$ varies along the iterations due to the possible deactivation of the links. In this case, the combination weights also change from one time instant to another, since (10a)–(10c) need to hold at every iteration.

The possibility of deactivating communication links was first analyzed in [109], where networks with time-varying topologies were considered. In that work, the availability of the communication links between

two neighboring nodes i and k is modeled as Bernoulli random variable with a success probability $p_{ik} = p_{ki}$. Although the main goal of [109] was to analyze the behavior of the network with link failures occurring at random, this idea was later widely employed as a benchmark for comparing the performance of link selection policies. For example, building upon this concept, a mechanism was proposed in [111] for controlling the probability of success of each link in an adaptive manner. In this solution, each node k assigns a probability $p_{\min} \leqslant p_{jk}(n) \leqslant p_{\max}$ to each link associated with the nodes $j \in \mathcal{N}_k(n)/\{k\}$, where $p_{\min}$ and $p_{\max}$ must be selected by the filter designer. It is worth noting that $p_{kk}(n) = 1$ for every n. The probabilities $p_{jk}(n)$ are updated based on the performance gain associated with it and on the resource constraints of the network, such as the amount of energy available at each node.

4.3. Censoring Strategies

Censoring strategies aim to prevent certain nodes from transmitting their local estimates to any of their neighbors, enabling them to temporarily turn their transmitters off and, consequently, save energy [118–127]. Usually, it is assumed that the node k still receives the data from its uncensored neighbors even when it does not send its own local estimate to them [118, 119, 124, 126, 127]. However, stricter versions in which the nodes cut their communications completely when they are censored have also been proposed [119].

Generically, some of these techniques can also be described by (23) with the additional restriction that $c_{jk}(n) = 0$ for every node $j \in \mathcal{N}_k(n)/\{k\}$ if node k is censored at iteration n [118, 124]. Another approach consists in assuming that the nodes can store the past local estimates from their neighbors [119, 127]. In this case, (52b) can be interpreted as

$$\mathbf{w}_k(n) = \sum_{i \in \mathcal{N}_k} c_{ik} \bar{t}_i(n) \boldsymbol{\psi}_i(n) + [1 - \bar{t}_i(n)] \bar{\boldsymbol{\psi}}_i(n), \tag{24}$$

where $\bar{t}_i(n) = 0$ if node i is censored at iteration n and $\bar{t}_i(n) = 1$ otherwise, and $\bar{\boldsymbol{\psi}}_i(n)$ is the last estimate received from node i.

One example of this technique was proposed in [119], which uses game theory, as well as information about the energy level at each node and performance indicators to determine whether each node should be censored or not. Another solution was proposed in [127] and later extended in [128]. In this technique, the nodes share their local estimates while the squared error is high in magnitude and are censored otherwise. To control the censoring of the nodes, the binary variable $\bar{t}_k(n) \in \{0,1\}$ is introduced in (9a) at each node k. Hence, (9a) is recast as [127]

$$\boldsymbol{\psi}_k(n) = [1 - \bar{t}_k(n)] \boldsymbol{\psi}_k(n-1) + \bar{t}_k(n) \left\{ \mathbf{w}_k(n-1) - \mu_k \widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)] \right\}. \tag{25}$$

Thus, when $\bar{t}_k(n) = 1$, (25) coincides with (9a). On the other hand, when $\bar{t}_k(n) = 0$, $\boldsymbol{\psi}_k(n)$ is kept fixed. Moreover, $\mu_k \widehat{\nabla}_{\mathbf{w}} J_k[\mathbf{w}_k(n-1)]$ does not need to be calculated, which saves computation, and $d_k(n)$ does

not have to be sampled. Hence, the algorithm can also be used as a sampling mechanism for, e.g., GSP applications [127]. If the nodes can store past information sent by their neighbors, there is no need for node k to re-transmit its local estimate. In this case, its neighbors should simply use the latest $\boldsymbol{\psi}_k$ that they have received from node k in (9b), enabling it to shut its transmitter off [127].

Instead of directly adapting $\bar{t}_k(n)$, an auxiliary variable $\alpha_k(n) \in [-\alpha^+,\alpha^+]$ is introduced such that $\bar{t}_k(n)=0$ for $\phi[\alpha_k(n)]<0.5$ and $\bar{t}_k(n)=1$ otherwise, with $\phi[\cdot]$ given by [223]

$$\phi[\alpha_k(n)] \triangleq \frac{\text{sgm}[\alpha_k(n)] - \text{sgm}[-\alpha^+]}{\text{sgm}[\alpha^+] - \text{sgm}[-\alpha^+]}, \tag{26}$$

where $\text{sgm}[x]=[1+\exp(-x)]^{-1}$ is a sigmoidal function. In the literature, $\alpha^+=4$ is usually adopted [223]. For compactness of notation, we write $\phi[\alpha_k(n)]$ as $\phi_k(n)$. Then, the following cost function is introduced [127, 128]:

$$J_{\alpha_k}(n)=\phi_k(n)\beta\bar{t}_k(n)+\big[1-\phi_k(n)\big]\sum_{i\in\mathcal{N}_k}c_{ik}(n)e_i^2(n), \tag{27}$$

where $\beta>0$ is a parameter that controls how much the censoring of the nodes is penalized. When the error is high in magnitude, $J_{\alpha_k}(n)$ is minimized by making $\phi_k(n)$ close to one, and thus node k is not censored. The same applies when node k is censored ($\bar{t}_k=0$), which ensures that the nodes do not remain censored permanently. On the other hand, when node k is not censored and the error is small in magnitude, $J_{\alpha_k}(n)$ is minimized by making $\phi_k(n)$ closer to zero, and the algorithm eventually censors node k [127].

The censoring mechanism is then obtained by taking the derivative of (27) with respect to $\alpha_k(n)$. Since $e_i(n)$ may be unknown when $\bar{t}_i(n)=0$, $e_i(n)$ is replaced by its latest measurement $\varepsilon_i(n)$, i.e. $\varepsilon_i(n)=\bar{t}_i(n)e_i(n)+[1-\bar{t}_i(n)]\varepsilon_i(n-1)$. Thus, the following stochastic gradient descent rule is obtained [127]:

$$\alpha_k(n+1)=\alpha_k(n)+\mu_t\phi_k'(n)\left[\sum_{i\in\mathcal{N}_k}c_{ik}(n)\varepsilon_i^2(n)-\beta\bar{t}_k(n)\right]. \tag{28}$$

where $\mu_t>0$ is a step size and [223]

$$\phi_k'(n)=\frac{d\phi[\alpha_k(n)]}{d\alpha_k(n)}=\frac{\text{sgm}[\alpha_k(n)]\{1-\text{sgm}[\alpha_k(n)]\}}{\text{sgm}[\alpha^+]-\text{sgm}[-\alpha^+]}. \tag{29}$$

From (28) we can see that the algorithm requires that every uncensored node i is required to transmit $\varepsilon_i^2(n)=e_i^2(n)$ to its neighbors. Nonetheless, this information can be sent bundled with the local estimates $\boldsymbol{\psi}_i$ so as to not increase the number of transmissions. In [128], modifications were proposed to address some of the weaknesses of the original algorithm. Firstly, it was observed in [127] that prior knowledge of the measurement noise power $\sigma_{v_k}^2$ for $k=1,\cdots,V$ was needed for a proper selection of the parameter β. This was addressed in [128] by allowing local and time-varying parameters $\beta_k(n)$ at each node k, which are updated based on estimates of $\sigma_{v_i}^2$ for $i\in\mathcal{N}_k$. Furthermore, a change detection mechanism was incorporated to improve the tracking capability of the algorithm [128].

4.4. Simulations – Effects of Restrictive Communication Policies

In this subsection, we present simulation results to illustrate some of the techniques reviewed in Secs. 4.1–4.3, and to investigate their impacts on the performance of the ATC dLMS algorithm. We consider the same scenario of Sec. 3.2. Furthermore, we adopt uniform combination weights [1], and four techniques for the reducing the amount of data exchange between the nodes: the Partial-Diffusion LMS of [100], the Link Probability Control algorithm of [111], the Energy-Aware algorithm of [119], and the Adaptive-Sampling-and-Censoring dLMS of [127]. For comparison, we also show the results obtained with dLMS without any restriction on the communications. In every case, we employ the same step size $\mu_k = 0.01$ for every node k, $k = 1, \cdots, 20$. In Fig. 8(a) we present the NMSD curves, and in Fig. 8(b) the total number of coefficients transmitted by the nodes throughout the network. In the middle of each realization, we flip the vector $\mathbf{w}^o$ to simulate an abrupt change in the environment. To enable a fair comparison, we adjusted the parameters so as to obtain roughly the same savings in communications for every algorithm. The resulting values of the parameters adopted are shown in Table 3. We can see from Fig. 8 that the restrictions on the communication between nodes led to a deterioration in the steady-state NMSD in every solution. In this regard, the PDLMS algorithm of [100] was the most affected, followed by the algorithms of [111], [119], and [127], respectively. In terms of the convergence rate, the solutions of [100], [111], and [127] followed closely the unrestricted dLMS algorithm, whereas the solution of [119] displayed a slower convergence rate. Overall, the simulations of Fig. 8 illustrate the trade-off between performance and energy savings due to the reduction of the amount of information exchange between the nodes. We remark that a theoretical analysis of adaptive diffusion networks in scenarios with imperfect communications between the nodes can be seen in, e.g. [207].

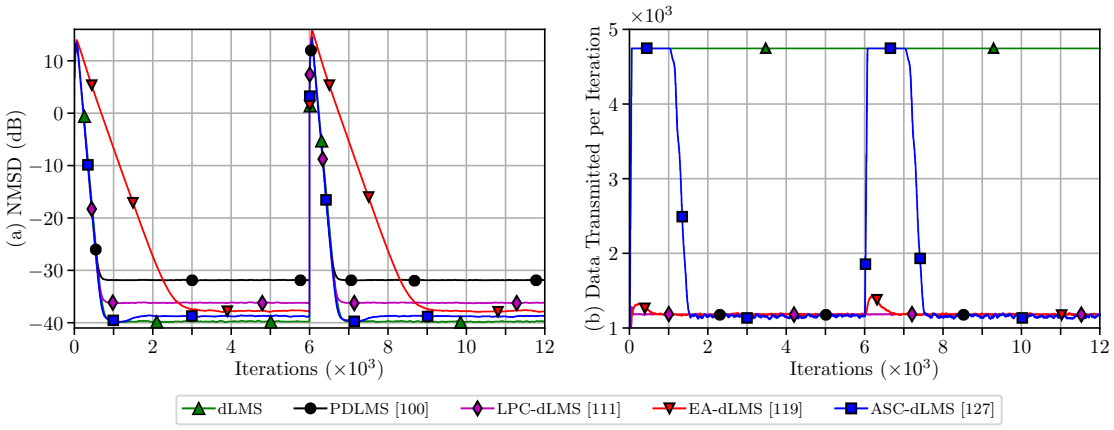

Figure 8: (a) NMSD Curves and (b) Number of coefficients transmitted throughout the network using the ATC dLMS algorithm in conjunction with the solutions of [100, 111, 119, 127], as well as the unrestricted algorithm.

Table 3: Parameters of the algorithms used in the simulations of Fig. 8.

Solution	Parameters
PDLMS [100]	$L = 16$, round-robin configuration
LPC-dLMS [111]	$p_{\min} = 0.25$, $p_{\max} = 1$, $a_{ik} = 1 \forall i,k$, $\alpha_k = 3$ for $k = 1, \cdots, V$, $\alpha_k = 3$ for $k = 1, \cdots, V$, $\nu_k(n) = 0.01/[10^{-4} + \|\widehat{Q}_k(n) q_k(n) + \widehat{r}_k(n)\|^2]$, uniform initialization
EA-dNLMS [119]	$E_{\text{Act}} = 33.5966 \cdot 10^{-3}$, $E_{\text{Tx}} = 15.16 \cdot 10^{-3}$, $K_{i,1} = 2$, $K_{i,2} = 0.5$, $K_g = 2$, $\gamma_g = 2, \gamma_i = 2$, $\delta = 0.5$, $\rho = 0.01$, $r = 2$
ASC-dLMS [127]	$\beta = 0.0148$, $\mu_t = 8.3638$

5. Multitask Adaptive Diffusion Networks

In Sec. 3, it was assumed that every node in the network has the common objective of estimating the same parameter vector $\mathbf{w}^{\text{o}}$, as described by (1). However, an interesting problem arises when different nodes try to estimate different parameters. Applications of this type are usually referred to as *multitask* estimation problems in the literature, in opposition to the *single-task* scenario modeled by (1). Multitask models can be useful for representing situations in which groups of agents have distinct but correlated objectives, or for modeling regional variations in the system to be identified. This diversity in the optimal system can play an important role in, e.g., meteorological applications, where the temperature can be governed by different dynamics at different geographic points [9, 12]. Hence, to reflect this, we recast (1) as

$$d_k(n) = \mathbf{u}_k^{\text{T}}(n) \mathbf{w}_k^{\text{o}} + v_k(n). \tag{30}$$

Eq. (1) can be seen as a special case of (30) in which $\mathbf{w}_1^{\text{o}} = \cdots = \mathbf{w}_V^{\text{o}} = \mathbf{w}^{\text{o}}$. This corresponds to an extreme case in which the parameter vector is the same for all nodes. The other extreme occurs when $\mathbf{w}_k^{\text{o}}$ is different for each node k. In many applications of multitask networks, an in-between case is considered, in which there are groups or *clusters* of nodes whose parameter vectors share a certain degree of similarity. In some works, networks that fall into the latter category are referred to as *clustered multitask networks*, and the expression "multitask networks" is reserved for the case where $\mathbf{w}_k^{\text{o}}$ is different for each node k [7, 13, 17]. For the sake of generality, in this paper we apply the term "multitask networks" to any network in which the optimal system is not the same for all nodes, and explicitly differentiate between the clustered and non-clustered scenarios when the context requires.

It can be shown that if the single-task diffusion LMS algorithm with static combination weights is used in a multitask environment, it produces biased estimates of the optimal parameter vectors $\mathbf{w}_k^{\text{o}}$ at each node k [9]. If the vectors $\mathbf{w}_k^{\text{o}}$ are similar across the network, this bias is small in magnitude and may be acceptable in many applications. However, as the spatial variations between the $\mathbf{w}_k^{\text{o}}$ increase in magnitude, the bias introduced by the diffusion can cause a visible deterioration in performance. In fact, depending on

the local discrepancies in the optimal system, the non-cooperative approach may outperform the diffusion strategies [9]. Intuitively, this is because each node can estimate its own parameter vector much better than its neighbors in this scenario, and the cooperation between nodes does not disseminate any useful information for the learning task of each individual node. For this reason, several different approaches were proposed for the development of efficient diffusion networks for multitask problems in different scenarios [7–16].

In [7], the diffusion LMS algorithm for multitask networks is derived. It is assumed that the network is divided into Q clusters $\mathcal{C}_1, \cdots, \mathcal{C}_Q$, and that each cluster $\mathcal{C}_i$ has the same optimal system $\mathbf{w}_{\mathcal{C}_i}^{\mathrm{o}}$, i.e., $\mathbf{w}_k^{\mathrm{o}} = \mathbf{w}_{\mathcal{C}_i}^{\mathrm{o}}$ for every node $k \in \mathcal{C}_i$. Furthermore, if the clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ are connected, i.e., there is at least one link between any nodes $k \in \mathcal{C}_i$ and $i \in \mathcal{C}_j$, it is assumed that their optimal systems are similar in some way. Clusters that are connected to each other are also called neighbors, analogously to the notion of neighborhood between nodes. The diffusion LMS algorithm for multitask networks is obtained by adding a regularization term to the cost function (3) to enforce the similarity between neighboring clusters. Using the squared Euclidean distance as a regularizer, this results in the following cost function [7]:

$$J_{\mathrm{global}}^{\mathrm{multitask}}(\mathbf{w}_{\mathcal{C}_i}, \cdots, \mathbf{w}_{\mathcal{C}_Q}) = \sum_{k=1}^{V} \mathrm{E}\{[d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_{\mathcal{C}(k)}]^2\} + \eta \sum_{k=1}^{V} \sum_{i \in \mathcal{N}_k/\mathcal{C}(k)} \rho_{ki} \left\| \mathbf{w}_{\mathcal{C}(k)} - \mathbf{w}_{\mathcal{C}(i)} \right\|^2, \qquad (31)$$

where $\mathcal{C}(k)$ denotes the cluster to which node k belongs. An example is shown in Fig. 9, in which, for instance, $\mathcal{C}(i) = \mathcal{C}_1$ and $\mathcal{C}(k) = \mathcal{C}_2$. Furthermore, $\mathcal{N}_k/\mathcal{C}(k)$ is the set of the neighbors of node k that do not belong to the cluster $\mathcal{C}(k)$, $\eta > 0$ is a regularization parameter, and the $\{\rho_{ki}\}$ are non-negative weights that adjust the regularization strength for each pair of nodes k and i. In [7], convex weights $\{\rho_{ki}\}$ are adopted.

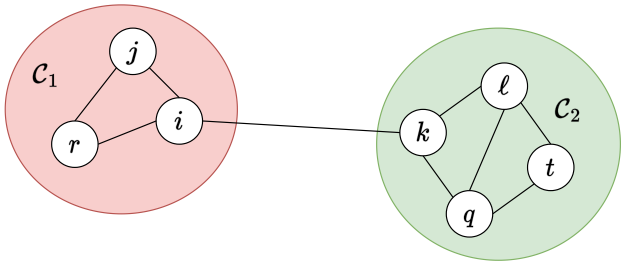

Figure 9: Example of a clustered network structure.

The cost function (31) promotes a stronger degree of similarity between the estimates of clusters that have many connections to each other, since in these cases the cardinalities of the sets $\mathcal{N}_k/\mathcal{C}(k)$ are large. Furthermore, it enforces symmetric regularization, i.e., two neighboring clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ promote the same level of similarity between their estimates due to the summation over the V nodes of the regularization term and to the symmetry of the term $\left\| \mathbf{w}_{\mathcal{C}(k)} - \mathbf{w}_{\mathcal{C}(i)} \right\|^2$ with respect to the vectors $\mathbf{w}_{\mathcal{C}(k)}$ and $\mathbf{w}_{\mathcal{C}(i)}$. Since it may be desirable to allow asymmetric regularization terms, (31) was modified in [7] and formulated as a sum of Nash equilibrium problems, one for each cluster $\mathcal{C}_i$. By slightly relaxing the cost function to enable its

minimization at each node using only the data available locally, and by applying a steepest-descent approach to the estimation of the equilibrium points of these problems and adopting stochastic approximations of the LMS type, the diffusion LMS for clustered multitask networks in an ATC configuration is given by [7]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) + \mu_k \left\{ \sum_{i \in \mathcal{N}_k \cap \mathcal{C}(k)} b_{ik}[d_i(n) - \mathbf{u}_i^{\mathrm{T}}(n)\mathbf{w}_k(n-1)]\mathbf{u}_i(n) + \eta \sum_{i \in \mathcal{N}_k / \mathcal{C}(k)} \rho_{ki}[\mathbf{w}_i(n-1) - \mathbf{w}_k(n-1)] \right\} & \text{(32a)} \\ \mathbf{w}_k(n) = \sum_{i \in \mathcal{N}_k \cap \mathcal{C}(k)} c_{ik} \boldsymbol{\psi}_i(n). & \text{(32b)} \end{cases}$$

Analyzing (32a), we can see that if $\eta = 0$ is chosen, that is equivalent to running the standard ATC dLMS within each cluster $\mathcal{C}_i$, without exchange of information between different clusters. Furthermore, the algorithm for the single-task scenario can be seen as a special case of (32) in which $\mathcal{N}_k \cap \mathcal{C}(k) = \mathcal{N}_k$ and $\mathcal{N}_k / \mathcal{C}(k) = \varnothing$ for $k = 1, \cdots, V$. Finally, the non-clustered multitask case can be analyzed by making $\mathcal{N}_k \cap \mathcal{C}(k) = \{k\}$ and $\mathcal{N}_k / \mathcal{C}(k) = \mathcal{N}_k / \{k\}$ for $k = 1, \cdots, V$ [7]. A pseudo-code description of the multitask version of the ATC dLMS for clustered networks is provided next in Algorithm 2.

A different approach was adopted in [9] where it was assumed that there is no prior knowledge about the clusters in the network. Hence, an unsupervised algorithm for the adaptive clustering of diffusion networks was proposed. The concept behind this solution is to only promote the cooperation between each node k and its neighbors $i \in \mathcal{N}_k$ with parameter vectors $\mathbf{w}_i^{\mathrm{o}}$ sufficiently similar to its own $\mathbf{w}_k^{\mathrm{o}}$. Since there is no *a priori* knowledge of the optimal systems $\mathbf{w}_i^{\mathrm{o}}$, $i \in \mathcal{N}_k$, an approximation is obtained by adjusting the combination weights $\{c_{ik}\}$ based on the norm of the difference between their local estimates $\boldsymbol{\psi}$. This is achieved by solving at each node k

$$\begin{aligned} \mathbf{c}_k = \arg \min_{\mathbf{c}_k \in \mathbb{R}^{V \times 1}} & \sum_{i=1}^{V} c_{ik}^2 \left\| \widehat{\mathbf{w}_k^{\mathrm{o}}} - \boldsymbol{\psi}_i(n) \right\|^2 \\ \text{subject to } & c_{ik} \geqslant 0,\ \mathbf{1}_V^{\mathrm{T}} \mathbf{c}_k = 1,\ c_{ik} = 0 \text{ if } i \notin \mathcal{N}_k, \end{aligned} \tag{33}$$

where $\widehat{\mathbf{w}_k^{\mathrm{o}}}$ is an estimate of $\mathbf{w}_k^{\mathrm{o}}$ and $\mathbf{c}_k$ is defined as

$$\mathbf{c}_k \triangleq [c_{1k} \ \cdots \ c_{Vk}]^{\mathrm{T}}. \tag{34}$$

In particular, the algorithm proposed in [8] uses

$$\widehat{\mathbf{w}_k^{\mathrm{o}}}(n) = \boldsymbol{\psi}_k(n) - \mu_k \hat{\nabla}_{\mathbf{w}} J_k[\boldsymbol{\psi}_k(n)] = \boldsymbol{\psi}_k(n) + \mu_k \mathbf{g}_k(n), \tag{35}$$

with

$$\mathbf{g}_k(n) = [d_k(n) - \mathbf{u}_k^{\mathrm{T}}(n)\boldsymbol{\psi}_k(n)]\mathbf{u}_k(n). \tag{36}$$

Algorithm 2 The ATC dLMS Algorithm of Eq. (32) for Clustered Multitask Networks.

% Initialization - For each node $k = 1, \cdots, V$, determine the cluster $\mathcal{C}(k)$ to which it belongs, select a step size μ_k as well as the parameter η and the weights ρ_{ki}, b_{ik}, and c_{ik} for $i = 1, \cdots, V$, and set $\boldsymbol{\psi}_k(0) \leftarrow \mathbf{0}_M$ and $\mathbf{w}_k(0) \leftarrow \mathbf{0}_M$

for $n = 1, 2, \cdots$ **do**

  for $k = 1, \cdots, V$ **do**

    Update $\mathbf{u}_k(n)$

  end for

  % The nodes transmit their local signals $\mathbf{u}$ and d and their combined estimates $\mathbf{w}$ to their neighbors

  % Adaptation Step

  for $k = 1, \cdots, V$ **do**

    % Adapting the local estimate $\boldsymbol{\psi}_k(n)$:

    $\boldsymbol{\psi}_k(n) \leftarrow \mathbf{w}_k(n-1)$

    for $i \in \mathcal{N}_k \cap \mathcal{C}(k)$ **do**

      $\boldsymbol{\psi}_k(n) \leftarrow \boldsymbol{\psi}_k(n) + \mu_k b_{ik}[d_i(n) - \mathbf{u}_i^{\mathrm{T}}(n)\mathbf{w}_k(n-1)]\mathbf{u}_i(n)$

    end for

    for $i \in \mathcal{N}_k / \mathcal{C}(k)$ **do**

      $\boldsymbol{\psi}_k(n) \leftarrow \boldsymbol{\psi}_k(n) + \mu_k \eta \rho_{ki}[\mathbf{w}_i(n-1) - \mathbf{w}_k(n-1)]$

    end for

  end for

  % The nodes transmit their local estimates $\boldsymbol{\psi}$ to their neighbors

  % Combination Step

  for $k = 1, \cdots, V$ **do**

    % Forming the combined estimate $\mathbf{w}_k(n)$:

    $\mathbf{w}_k(n) \leftarrow \mathbf{0}_M$

    for $i \in \mathcal{N}_k \cap \mathcal{C}(k)$ **do**

      $\mathbf{w}_k(n) \leftarrow \mathbf{w}_k(n) + c_{ik}\boldsymbol{\psi}_i(n)$

    end for

  end for

end for

Thus, one possible solution for (33) is given by

$$c_{ik}(n) = \begin{cases} \dfrac{\left(\left\|\widehat{\mathbf{w}}_k^{\mathrm{o}}(n) - \boldsymbol{\psi}_i(n)\right\|^2\right)^{-1}}{\displaystyle\sum_{i\in\mathcal{N}_k}\left(\left\|\widehat{\mathbf{w}}_k^{\mathrm{o}}(n) - \boldsymbol{\psi}_i(n)\right\|^2\right)^{-1}}, \text{ if } i\in\mathcal{N}_k, \\ 0, \text{ otherwise,} \end{cases} \tag{37}$$

with $\widehat{\mathbf{w}}_k^{\mathrm{o}}(n)$ given by (35). Incorporating (37) into (12) yields the ATC diffusion LMS with Adaptive Clustering for multitask problems. It is interesting to notice that there is a certain similarity between (37) and the ACW algorithm of Rule 8) in Table 2, despite the differences in context that motivate each solution.

As mentioned previously, multiple solutions were proposed for multitask estimation problems over diffusion networks. In [8], an ATC diffusion LMS algorithm was obtained for the case where the optimal systems $\mathbf{w}_k^{\mathrm{o}}$ can be described by the sum of a common component and a node-specific one for $k = 1, \cdots, V$ under some restrictions so as to ensure a unique optimal solution to the estimation problem. In [10], it is assumed that there are parameters that are of global interest to all nodes in the network, others that are of interest to a subset of nodes, and others that are of local interest to specific nodes, and a diffusion algorithm of the LMS type is derived for this problem. In [11], the authors propose a diffusion strategy that promotes the sparsity of the vector difference $\mathbf{w}_{\mathcal{C}_i} - \mathbf{w}_{\mathcal{C}_j}$ of neighboring clusters $\mathcal{C}_i$ and $\mathcal{C}_j$. Attempting to cover all of these solutions in detail in this paper would not be reasonable due to the wide scope of this work. An excellent review paper focused specifically on multitask adaptive diffusion networks can be found in [38].

6. Kernel Adaptive Diffusion Networks

In Secs. 3–5, we have discussed adaptive solutions aimed at solving linear estimation problems, as becomes evident from the modeling of the desired signal in (1) and (30). However, it is not unusual to encounter problems that are nonlinear in nature. To deal with this, kernel-based adaptive diffusion networks have been proposed in the literature and attracted significant attention [19–23]. In comparison with (1), the kernel framework for adaptive diffusion networks considers a modified model for the desired signal $d_k(n)$, given by [19, 22, 224]

$$d_k(n) = \varphi^{\mathrm{o}}\left[\mathbf{u}_k(n)\right] + v_k(n), \tag{38}$$

where $\varphi^{\mathrm{o}}(\cdot)$ typically denotes a nonlinear transformation of the input vector $\mathbf{u}_k(n)$. For simplicity, we restrict our review to the single-task scenario, but an analogous model can be obtained for the multitask one if we consider different mapping functions for different nodes [224].

Kernel-based adaptive diffusion networks are largely based on *kernel adaptive filters*, which over the years became established tools for nonlinear signal processing [134–136]. The idea behind these techniques is to

apply a nonlinear transformation $\boldsymbol{\Phi} : \mathbb{R}^M \rightarrow \mathbb{F}$ to the input vectors, where $\mathbb{F}$ is usually a higher-dimensional space, named *feature space*. Then, linear signal processing techniques are applied to the vectors mapped in $\mathbb{F}$ [134–136, 225]. This way, we can perform filtering tasks that are nonlinear in $\mathbb{U}$ while only employing linear procedures in the feature space.

Once a nonlinear mapping function $\boldsymbol{\Phi}(\cdot)$ is adopted, the Mercer kernel $\kappa : \mathbb{R}^M \times \mathbb{R}^M \rightarrow \mathbb{R}$ associated with it is defined as the inner product [225]

$$\kappa(\mathbf{u},\mathbf{u}') = \boldsymbol{\Phi}^{\mathrm{T}}(\mathbf{u})\boldsymbol{\Phi}(\mathbf{u}'). \tag{39}$$

Depending on how the mapping function $\boldsymbol{\Phi}(\cdot)$ is defined, we may be able to calculate $\kappa(\mathbf{u},\mathbf{u}')$ without explicitly knowing $\boldsymbol{\Phi}(\cdot)$. This is known as the *kernel trick*, which is a cornerstone of many kernel-based methods, such as kernel adaptive filters and support vector machines (SVMs) [134–136, 225, 226]. One of the most widely adopted kernels is the Gaussian kernel, given by

$$\kappa(\mathbf{u},\mathbf{u}') = \exp\left(\frac{-\left\|\mathbf{u}-\mathbf{u}'\right\|^2}{2h^2}\right), \tag{40}$$

where h is the Gaussian kernel bandwidth [134–136, 225, 226]. It is worth mentioning that the feature space $\mathbb{F}$ associated with the Gaussian kernel is an infinite-dimensional space. Other examples of commonly used kernels include sigmoidal, homogeneous and non-homogeneous polynomial kernels, among many others [134, 225].

In their original forms, the computational costs of these algorithms increase linearly with every iteration, which makes their implementation infeasible [134–136]. To deal with this, the concept of *dictionaries* was introduced in the kernel literature. The idea is to use only a limited set of data in the processing, which restricts the computational cost. Evidently, this leads to a deterioration in performance in comparison with the case where the dictionary grows larger at every iteration, but it is crucial to enable the use of these algorithms in practice. Several techniques have been proposed over the years for the selection of the dictionaries, aiming to limit the growth of the computational cost while maintaining performance as much as possible [134–136].

We then seek to estimate the desired signal by obtaining an approximation φ_k for $\varphi^{\mathrm{o}}[\cdot]$ in (38) in a distributed manner in each node k. Considering a global cost function of the form (8), with [19]

$$J_k(\varphi_k) = \mathrm{E}\left\{\left|d_k(n) - \varphi_k\left[\mathbf{u}_k(n)\right]\right|^2\right\}, \tag{41}$$

and applying the stochastic gradient descent, many different algorithms can be obtained. For example, in [19], following an LMS approach and adopting an ATC diffusion strategy with enlarged cooperation, and assuming that there is a dictionary $\mathcal{D} = \{\mathbf{u}_{\mathcal{D}_1}, \cdots, \mathbf{u}_{\mathcal{D}_D}\}$ of cardinality D common to all nodes in the

network, the Functional ATC diffusion Kernel LMS algorithm is obtained as [224]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) + \mu_k \sum\limits_{i \in \mathcal{N}_k} b_{ik}\Big[d_i(n) - \mathbf{w}_k^{\mathrm{T}}(n-1)\boldsymbol{\kappa}_i(n)\Big]\boldsymbol{\kappa}_i(n) & \text{(42a)} \\ \mathbf{w}_k(n) = \sum\limits_{i \in \mathcal{N}_k} c_{ik}\boldsymbol{\psi}_i(n), & \text{(42b)} \end{cases}$$

where

$$\boldsymbol{\kappa}_k(n) = \Big[\kappa\left(\mathbf{u}_k(n),\mathbf{u}_{\mathcal{D}_1}\right) \;\cdots\; \kappa\left(\mathbf{u}_k(n),\mathbf{u}_{\mathcal{D}_D}\right)\Big]^{\mathrm{T}} \tag{43}$$

is a vector comprised of the kernel values computed between the current input vector $\mathbf{u}_k(n)$ and the elements of the dictionary $\mathcal{D}$ [224].

Multikernel solutions have also been proposed for diffusion-based adaptive learning [22, 227, 228]. As their name suggests, these techniques employ more than one kernel simultaneously in the learning task. The idea is that by using two or more kernels, we may be able to grasp nuances in the desired signal due to, e.g., the existence of high and low frequency components in the nonlinear function. A version of this technique was employed in [22] for environmental monitoring.

The fact that the dictionary $\mathcal{D}$ is shared among all nodes has some implications on the practical implementations of kernel adaptive diffusion networks. The simplest way to ensure all nodes have the same dictionary would be to define its elements and inform them to all nodes before the learning process begins [224]. Nonetheless, a general-case rule for the selection of this dictionary remains an open research topic. Furthermore, this approach can cause a degradation in performance, especially if the scenario changes in such a way that the predefined dictionary ceases to be representative of the streaming data. To circumvent these issues, one might allow a time-varying dictionary using criteria such as the ones presented in [134–136]. However, in this case the necessity of sharing the dictionary with every node in the network in an online and distributed manner poses a challenge for diffusion strategies [24]. A hybrid solution, in which each node has access to both a shared and fixed dictionary and to a local and time-varying one was proposed in [229]. The idea is that the shared dictionary should provide the nodes with a rough estimate of the nonlinear function over the whole network, whereas the local part could allow them to detect particularities of the signals in its vicinity. Despite this, the handling of the dictionary in diffusion strategies is still regarded as an open research topic [24, 224]. In order to circumvent these difficulties, *Random Fourier Features* (RFF) approaches have been proposed for kernel-based adaptive diffusion networks [24–27].

In RFF solutions, instead of using the kernel trick, we apply an RFF map $\mathbf{z} : \mathbb{R}^M \to \mathbb{R}^D$, with $D > M$, to the regressor vector $\mathbf{u}_k(n)$ based on Bochner's Theorem [24, 230]. This theorem ensures that, if the kernel is shift-invariant and positive definite, i.e., $\kappa(\mathbf{u},\mathbf{u}')$ depends exclusively on $\mathbf{u} - \mathbf{u}'$, and $\kappa(\mathbf{u},\mathbf{u}') > 0$ for any $\mathbf{u}$ and $\mathbf{u}'$, the Fourier transform $p(\boldsymbol{\omega})$ of the kernel is a probability density function such that [24, 230]

$$\kappa(\mathbf{u}-\mathbf{u}') = \int p(\boldsymbol{\omega}) e^{j\boldsymbol{\omega}^{\mathrm{T}}(\mathbf{u}-\mathbf{u}')} d\boldsymbol{\omega}, \tag{44}$$

where we have written $\kappa(\mathbf{u},\mathbf{u}')$ as $\kappa(\mathbf{u}-\mathbf{u}')$ for compactness [24]. We should notice that the Gaussian kernel given by (40) is shift-invariant and positive definite.

Since $p(\boldsymbol{\omega})$ and $\kappa(\mathbf{u}-\mathbf{u}')$ are real, the integral (44) converges when the complex exponentials are replaced by cosines. Hence, the real-valued mapping $z_{\boldsymbol{\omega},\theta}[\mathbf{u}] = \sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\mathbf{u} + \theta)$ also satisfies (44) if $\boldsymbol{\omega}$ is drawn from $p(\boldsymbol{\omega})$ and θ is uniformly distributed in the range $[0, 2\pi]$ [24]. Thus, $\kappa(\mathbf{u} - \mathbf{u}')$ can be computed as $\kappa(\mathbf{u} - \mathbf{u}') = \mathrm{E}\{z_{\boldsymbol{\omega},\theta}[\mathbf{u}]z_{\boldsymbol{\omega},\theta}[\mathbf{u}']\}$. To reduce the variance of this estimate, a sample average of D randomly chosen $z_{\boldsymbol{\omega},b}[\cdot]$ is used, i.e.,

$$\kappa(\mathbf{u}, \mathbf{u}') \approx \frac{1}{D}\sum_{i=1}^{D} z_{\boldsymbol{\omega}_i,\theta_i}[\mathbf{u}]z_{\boldsymbol{\omega}_i,\theta_i}[\mathbf{u}']. \tag{45}$$

Thus, the vector $\mathbf{u}_k(n)$ can be mapped to the following D-dimensional RFF vector [24, 25]:

$$\mathbf{z}_k(n) = \sqrt{\frac{2}{D}}\begin{bmatrix} \cos[\boldsymbol{\omega}_1^{\mathrm{T}}\mathbf{u}_k(n) + \theta_1] \\ \vdots \\ \cos[\boldsymbol{\omega}_D^{\mathrm{T}}\mathbf{u}_k(n) + \theta_D] \end{bmatrix}, \tag{46}$$

For the Gaussian kernel, $\boldsymbol{\omega}_i$, $i = 1, \cdots, D$ are drawn from the multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{I}_D/h^2$ [24, 230].

Since the RFF space has a finite dimension D, we can estimate $d_k(n)$ at node k by directly using a similar strategy to that of the linear ATC dLMS of (9). Thus, the ATC RFF-dKLMS consists in two steps given by [24, 25]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) + \mu_k[d_k(n) - \mathbf{z}_k^{\mathrm{T}}(n)\mathbf{w}_k(n-1)]\mathbf{z}_k(n) & \text{(47a)} \\ \mathbf{w}_k(n) = \sum_{i\in\mathcal{N}_k} c_{ik}\boldsymbol{\psi}_i(n). & \text{(47b)} \end{cases}$$

A pseudo-code description of ATC RFF-dKLMS is presented for convenience as Algorithm 3.

The idea behind the RFF approach is that the features are selected randomly according to the aforementioned distributions. This only needs to be done once, before the beginning of the learning task. Subsequently, the selected features are informed to the nodes. Due to Bochner's Theorem, as long as $\boldsymbol{\omega}$ and b are drawn from the appropriate distributions, this approach avoids the issue of how to select the elements of the dictionary $\mathcal{D}$ of the non-RFF kernel solutions. As one might expect, the more features are used, the better the performance of the RFF diffusion algorithms tends to be [24–27]. On the other hand, the computational cost also increases as more features are used.

7. Graph Signal Processing and Adaptive Diffusion Networks

So far, we have been assuming that the signals at a certain node do not influence the desired signal at the remainder of the network in any way. For instance, even if nodes k and ℓ are neighbors, $u_k(n)$ only influences $d_k(n)$, and $u_\ell(n)$ will affect $d_\ell(n)$, in its turn. This becomes clear when we analyze Eqs. (1), (30),

Algorithm 3 The ATC RFF-dLMS Algorithm of Eq. (47).

% Initialization - draw D scalars $\theta_1, \cdots, \theta_D$ from a uniform distribution in the range $[0, 2\pi]$ and D vectors $\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_D$ from a multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{I}_D/h^2$. These parameters shall be common to every node in the network. Then, for each node $k = 1, \cdots, V$, select a step size μ_k and combination weights c_{ik} satisfying (10) for $i = 1, \cdots, V$, and set $\boldsymbol{\psi}_k(0) \leftarrow \mathbf{0}_M$ and $\mathbf{w}_k(0) \leftarrow \mathbf{0}_M$

for $n = 1, 2, \cdots$ **do**

  % Adaptation Step

  for $k = 1, \cdots, V$ **do**

    Update $\mathbf{u}_k(n)$

    % Mapping $\mathbf{u}_k(n)$ to the RFF vector $\mathbf{z}_k(n)$:

    $$\mathbf{z}_k(n) \leftarrow \sqrt{\frac{2}{D}} \begin{bmatrix} \cos[\boldsymbol{\omega}_1^{\mathrm{T}} \mathbf{u}_k(n) + \theta_1] \\ \vdots \\ \cos[\boldsymbol{\omega}_D^{\mathrm{T}} \mathbf{u}_k(n) + \theta_D] \end{bmatrix}$$

    % Adapting the local estimate $\boldsymbol{\psi}_k(n)$:

    $$\boldsymbol{\psi}_k(n) \leftarrow \mathbf{w}_k(n-1) + \mu_k [d_k(n) - \mathbf{z}_k^{\mathrm{T}}(n) \mathbf{w}_k(n-1)] \mathbf{z}_k(n)$$

  end for

  % The nodes transmit their local estimates $\boldsymbol{\psi}$ to their neighbors

  % Combination Step

  for $k = 1, \cdots, V$ **do**

    % Forming the combined estimate $\mathbf{w}_k(n)$:

    $\mathbf{w}_k(n) \leftarrow \mathbf{0}_M$

    for $i \in \mathcal{N}_k$ **do**

      $\mathbf{w}_k(n) \leftarrow \mathbf{w}_k(n) + c_{ik} \boldsymbol{\psi}_i(n)$

    end for

  end for

end for

and (38). However, there may be situations in which the signals measured at a node do have an impact on its neighborhood. Broadly speaking, the goal of Graph Signal Processing (GSP) is exactly to model scenarios in which there is an underlying relationship between the data distributed over a certain domain (such as space). For this reason, as we will see next, extensions of GSP have also been applied to adaptive diffusion networks, in order to address problems in which the signals are spatially related to one another.

Before going forward, it is important to remark that GSP has become a broad field, with many ramifications. Extensions of several classical ideas of the signal processing field to the graph framework have

been proposed in the literature, such as the graph Fourier transform, graph signal convolution, graph filters, among many others [31, 154–161]. In this paper, rather than providing a comprehensive overview of GSP, we focus on how it can be used in conjunction with adaptive diffusion networks to model certain situations that cannot be easily represented using the theory of Secs. 3–6. For a much broader view of the GSP field, we suggest that the reader refers to, e.g., [156, 157, 161, 231] and their references. With this in mind, we begin our exposition on diffusion solutions for GSP in the following manner. Firstly, in Sec. 7.1, we provide some preliminary concepts that are fundamental for the understanding of the adaptive diffusion networks for GSP, which are reviewed in Sec. 7.2.

7.1. Preliminaries to Adaptive Diffusion over Graphs

Graphs are structures commonly used to represent interactions between objects of interest, consisting of a set of points, called *nodes*, and a set of lines connecting certain pairs of points, called *edges* [232]. Typically, nodes in a graph represent objects and/or agents, edges describe relationships between objects or agents, and weights represent the strength of the relation [232, 233]. Moreover, weights can be assigned to each edge, to represent the "strength" of the link between any pair of nodes. Finally, a graph is said to be *directed* if there is an orientation associated with each edge, i.e., from node i to node j or vice-versa, and if the weights associated with one direction are different from those associated with the other [233].

The adaptive networks presented in Section 3 can be considered examples of graphs. This is not a coincidence. Graphs are mathematical abstractions capable of representing a wide variety of real-world situations [233]. A relevant property of graphs is that they admit matrix representations, which makes them computationally manipulable [233]. One of the ways to represent them in this way is through the *adjacency matrix* of the graph. For a simple graph with V nodes, the adjacency matrix $\mathbf{A}$ is a $V \times V$ matrix whose element $[\mathbf{A}]_{ij}$ is equal to the value of the weight associated with the edge connecting the nodes i and j. It is interesting to note that the adjacency matrix of an undirected graph is symmetric with respect to its main diagonal.

The field of GSP is based upon the assumption that, at each node k and iteration n, there is a value of interest, whose meaning may vary from one application to the other. For example, these values can represent the temperature measured at a certain location, the power spectral density of a radio signal that a device receives from a certain source, the affinity of a social network user for certain types of content, and so on. The collection of these values of interest can be understood as a *signal defined over a graph*, or *graph signal* for brevity [156]. Thus, given a graph $\mathcal{G}$ with V nodes labeled $k = 1,\ 2,\ \cdots,\ V$ and adjacency matrix $\mathbf{A}$, a signal defined over $\mathcal{G}$ is represented by a vector of the form $\mathbf{u}(n) = [u_1(n)\ u_2(n)\ \cdots\ u_V(n)]$, where each element $u_k(n)$ is indexed by a node k [154].

Based on this definition, the concept of filters for graph signals, or *graph filters* for short, was proposed in [154]. For this, an analogy is drawn between the adjacency matrix and the unit delay operator in

discrete time signal processing. To illustrate this idea, let us consider a periodic sequence with V elements $u_1,\ u_2,\ \cdots,\ u_V$. This sequence could be represented by a graph in which each sample u_k is associated with the corresponding node k, and there is a directed edge from node k to its successor, node $k+1$ (unless $k = V$, in which case there is an edge connecting it to node 1). This concept is depicted in Fig. 10.

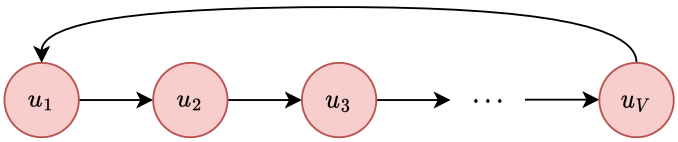

Figure 10: A discrete periodic time sequence represented as a graph.

Assigning unit weights to the edges of the graph of Fig. 10, its adjacency matrix is given by [154]

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

Collecting the samples of the sequence into a vector $\mathbf{u} = [u_1\ u_2\ \cdots\ u_V]$, we observe that by left-multiplying $\mathbf{u}$ by $\mathbf{A}$, we obtain the vector $\bar{\mathbf{u}} = \mathbf{A}\mathbf{u} = [u_V\ u_1\ \cdots\ u_2]$. Taking into account that $\mathbf{u}$ is a vector representation of a discrete time sequence, $\bar{\mathbf{u}}$ can be understood as a version of $\mathbf{u}$ delayed by one time unit. Thus, the adjacency matrix fulfills a role similar to the time delay operator in discrete time signal processing. Although this analogy is based specifically on the graph of Fig. 10, this notion is generalized for any adjacency matrix $\mathbf{A}$ in the GSP framework. In the general case, it can be understood that the multiplication of a graph signal $\mathbf{u}$ by $\mathbf{A}$ represents a spatial shift along the graph. For this reason, the adjacency matrix is considered a *graph-shift operator*. Over time, the usage of other matrices as graph shift operators was proposed in the literature, such as the Laplacian matrix [28, 29]. Although the analogy with discrete-time signal processing is less clear in this case, it allows for different forms of modeling the effects of the graph on the evolution of the signals defined over it. For the sake of generality, in this paper we denote the graph shift operator by $\mathbf{A}$, but we assume that it can represent any suitable choice for this operator, not just the adjacency matrix.

Thus, we can define a *linear shift-invariant graph filter* as a system given by [154]

$$\mathbf{H} \triangleq \sum_{m=0}^{M-1} w_m^{\mathrm{o}} \mathbf{A}^m, \tag{48}$$

where M is the filter length and $\{w_m^{\mathrm{o}}\}_{m=0}^{M-1}$ denotes its coefficients. Hence, if a graph signal $\mathbf{u}(n)$ is processed by this system, its output $\mathbf{h}(n)$ can be described as [154]

$$\mathbf{h}(n) = \sum_{m=0}^{M-1} w_m^{\mathrm{o}} \mathbf{A}^m \mathbf{u}(n), \tag{49}$$

as illustrated in Fig. 11.

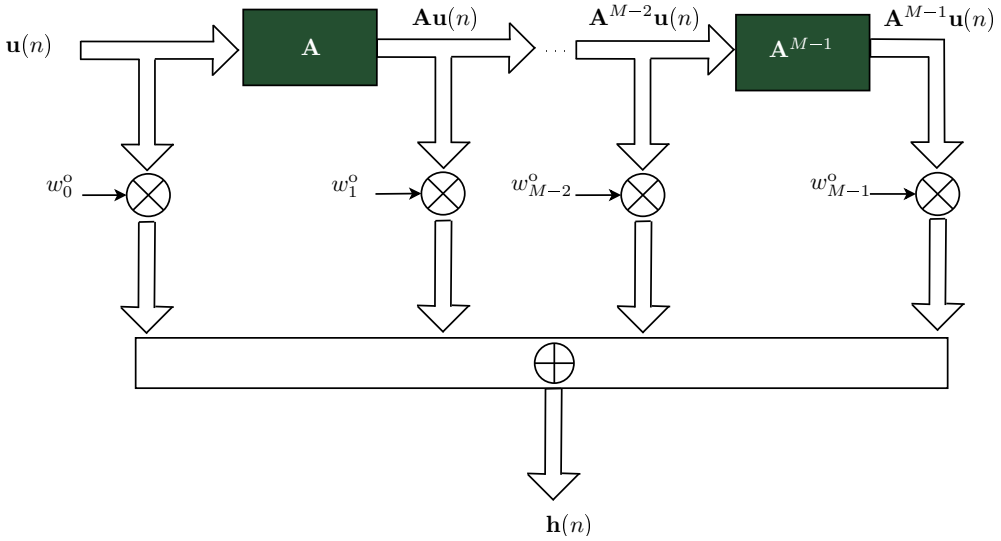

Figure 11: Schematic representation of a linear shift-invariant graph filter whose output is given by (49). The output of each block with the matrix $\mathbf{A}$, having the vector $\mathbf{u}(n)$ as its input, is given by the left-multiplication of $\mathbf{u}(n)$ by $\mathbf{A}$, i.e., by the vector $\mathbf{A}\mathbf{u}(n)$.

We should notice that there is a clear analogy to the tapped delay line commonly found in discrete-time filters [74]. This motivated the proposal of adaptive filters for GSP [28, 29] based on this concept, which will be exposed in Sec. 7.2

7.2. Diffusion Algorithms for Graph Signal Processing

Let us consider a graph with a predefined topology and V nodes labeled $1, \cdots, k, \cdots, V$. Each node k has access at each time instant n to an input signal $u_k(n)$ and to a desired signal $d_k(n)$, modeled as [26, 28, 29, 234, 235]

$$d_k(n) = \mathbf{x}_k^{\mathrm{T}}(n)\mathbf{w}^{\mathrm{o}} + v_k(n), \tag{50}$$

where $v_k(n)$ is the measurement noise at node k, $\mathbf{w}^{\mathrm{o}}$ is the optimal system, and the input $\mathbf{x}_k(n)$ is given by [26, 28, 29, 234, 235]

$$\mathbf{x}_k(n) = \Big[[\mathbf{u}(n)]_k \, [\mathbf{A}\mathbf{u}(n-1)]_k \, \cdots \, [\mathbf{A}^{M-1}\mathbf{u}(n-M+1)]_k \Big]^{\mathrm{T}}. \tag{51}$$

Eq. (51) can be interpreted as follows. The vector $\mathbf{u}(n)$ represents the "raw" information available at each node of the network at the iteration n, whereas $\mathbf{x}_k(n)$ models the spreading of that information throughout the graph, which is the result of both a temporal and spatial shift, or "delay".

In light of (51), it may be interesting to compare (50) with (1). In the model of Eq. (1), only the input signal $u_k(n)$ influences $d_k(n)$, whereas in the model of Eq. (50), $d_k(n)$ is influenced by $u_k(n)$, by $u_i(n)$ for

$i \in \mathcal{N}_k$, by the input signal at the two-hop neighbors of node k, and so on. Hence, unlike (1), the model of (50) can be used to represent situations in which the desired signal at a node is affected by what happens around it, and, therefore, on the measurements collected by its neighbors. In other words, the defining difference between the "classical" model and the GSP framework lies in the role of the spatial aspect of the problem. In the former, the topology of the network does not influence the dynamics of the desired signal. We see from (1) that $d_k(n)$ depends only on the signal $u_k(n)$ and on the measurement noise $v_k(n)$, and is independent of $u_i(n)$ for all $i = 1, \cdots, V$, $i \neq k$. In this case, the information does not propagate from one node to another. In the context of GSP, we see from (50) and (51) that if the nodes i and k are immediate neighbors, $d_k(n)$ depends on $u_i(n-1)$, since the information from one node spreads to its neighbors over time. Moreover, if nodes j and k are two-hop neighbors (i.e., it is possible to travel from node j to node k in two hops), $d_k(n)$ also depends on $u_j(n-2)$, and so on. Hence, the topology of the network plays a major role in how the desired signal $d_k(n)$ unfolds at each node k. In its turn, the optimal system $\mathbf{w}^{\mathrm{o}}$ models how exactly the graph topology and time lag affect the spreading of information in (50). This makes the graph-based framework well suited for distributed problems where both time and space must be taken into consideration, as in meteorological applications [28, 29, 127].

Furthermore, comparing (50) and (51) with (49), we can see that the model presented in this section modifies the original concept of graph filter by incorporating a time lag. In this case, rather than spreading instantaneously throughout the network as in (49), the information takes time to arrive at other nodes, with distant locations being influenced later than closer ones.

Apart from these differences between the GSP problem from the original distributed signal processing framework, the derivation of the dLMS algorithm for GSP in [28] follows an analogous path to what was done in Sec. 3. Replacing $\mathbf{u}_k(n)$ with $\mathbf{x}_k(n)$ in (3) and considering (50), the adoption of an approach of the LMS type to the minimization of (8) with an ATC configuration leads to [28, 29, 234, 235]

$$\begin{cases} \boldsymbol{\psi}_k(n) = \mathbf{w}_k(n-1) + \mu_k[d_k(n) - \mathbf{x}_k^{\mathrm{T}}(n)\mathbf{w}_k(n-1)]\mathbf{x}_k(n) & \text{(52a)} \\ \mathbf{w}_k(n) = \sum\limits_{i \in \mathcal{N}_k} c_{ik}\boldsymbol{\psi}_i(n). & \text{(52b)} \end{cases}$$

Eqs. (52a) and (52b) form the basis for the ATC dLMS algorithm for GSP. For ease of reference, in this paper we shall refer to the algorithm of (52) as the diffusion Graph LMS (dGLMS) algorithm, whose pseudo-code description is presented in Algorithm 4. Comparing (52) with (12), we see that there is a clear analogy between these solutions, with the main difference residing on the meaning of their inputs. Furthermore, it is important to notice that the neighborhood $\mathcal{N}_k$ of node k in (52b) refers to the communication network of the diffusion algorithm, which does not necessarily coincide with the graph represented by $\mathbf{A}$. For example, we may allow nodes to communicate with farther agents, whose effects on their signals are negligible. The inverse may also happen, with nodes being unable to communicate with other agents that do affect their

dynamics.

Algorithm 4 The ATC dGLMS Algorithm of Eq. (52).

% Initialization - for each node $k = 1, \cdots, V$, select a step size μ_k and combination weights c_{ik} satisfying (10) for $i = 1, \cdots, V$, and set $\boldsymbol{\psi}_k(0) \leftarrow \mathbf{0}_M$ and $\mathbf{w}_k(0) \leftarrow \mathbf{0}_M$
for $n = 1, 2, \cdots$ **do**
    % Adaptation Step
    for $k = 1, \cdots, V$ **do**
        Update $\mathbf{u}_k(n)$
        % Calculating the vector $\mathbf{x}_k$:
        $\mathbf{x}_k(n) \leftarrow \left[[\mathbf{u}(n)]_k \, [\mathbf{A}\mathbf{u}(n-1)]_k \, \cdots \, [\mathbf{A}^{M-1}\mathbf{u}(n-M+1)]_k\right]^{\mathrm{T}}$.
        % Adapting the local estimate $\boldsymbol{\psi}_k(n)$:
        $\boldsymbol{\psi}_k(n) \leftarrow \mathbf{w}_k(n-1) + \mu_k[d_k(n) - \mathbf{x}_k^{\mathrm{T}}(n)\mathbf{w}_k(n-1)]\mathbf{x}_k(n)$
    end for
    % The nodes transmit their local estimates $\boldsymbol{\psi}$ to their neighbors
    % Combination Step
    for $k = 1, \cdots, V$ **do**
        % Forming the combined estimate $\mathbf{w}_k(n)$:
        $\mathbf{w}_k(n) \leftarrow \mathbf{0}_M$
        for $i \in \mathcal{N}_k$ **do**
            $\mathbf{w}_k(n) \leftarrow \mathbf{w}_k(n) + c_{ik}\boldsymbol{\psi}_i(n)$
        end for
    end for
end for

Finally, it is worth mentioning that multitask [38, 234] and kernel-based [26, 236, 237] versions of graph diffusion algorithms have also been proposed in the literature. They apply analogous lines of reasoning to those exposed in Secs. 5 and 6 to diffusion graph adaptive filters such as (52). For instance, in [26], a Graph Diffusion KLMS filter with a pre-selected dictionary is proposed, as well as an RFF Graph diffusion KLMS algorithm. In this latter case, the algorithm is similar to the one described by (47), with the difference that the mapped vector $\mathbf{z}_k$ is calculated using $\mathbf{x}_k$ instead of $\mathbf{u}_k$ in (46), with $\mathbf{x}_k$ given by (51) [26].

Besides the dLMS of (12), other types of diffusion algorithms can also be extended to the system identification problem in GSP. For example, in [29], building from (52), an LMS-Newton type of diffusion algorithm was proposed to improve the convergence rate of the solution.

8. Application Example: Temperature Prediction

In this section, we consider a temperature dataset of daily average measurements (in °F) from 12/25/2001 to 12/21/2012 at $V = 100$ weather stations across Brazil [238]. We consider that each station corresponds to a node of a network. To determine the communication links, we applied the following procedure. Firstly, we represent the network as a directed weighted graph in which each node k is connected to the six nearest stations. Denoting this set by $\mathcal{N}_{\mathbf{A}_k}$, each element $[\mathbf{A}]_{kj}$ of the adjacency matrix $\mathbf{A}$ is given by [28]

$$[\mathbf{A}]_{kj} = \begin{cases} \dfrac{\exp(-g_{kj}^2)}{\displaystyle\sum_{\ell \in \mathcal{N}_{\mathbf{A}_k}} \exp(-g_{\ell k}^2) \sum_{i \in \mathcal{N}_{\mathbf{A}_j}} \exp(-g_{ji}^2)}, \text{if } j \in \mathcal{N}_{\mathbf{A}_k} \\ 0, \text{ otherwise} \end{cases}, \tag{53}$$

where g_{kj} is the geodesical distance between nodes k and j. Thus, we consider that nodes k and j can communicate if $k \in \mathcal{N}_{\mathbf{A}_j}$ and $j \in \mathcal{N}_{\mathbf{A}_k}$ simultaneously. The resulting network is depicted in Fig. 12(a), along with the temperature measured in each station on 06/21/2002.

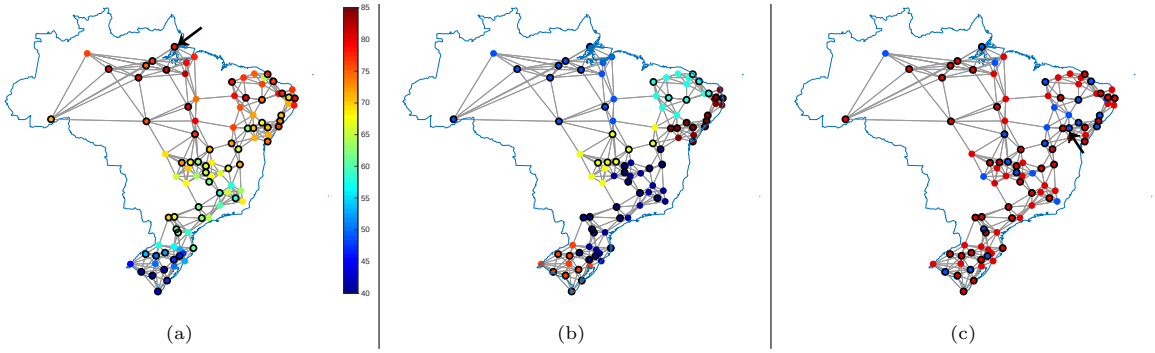

Figure 12: Network used in the simulations of Secs. 8.1 and 8.2. Edges represent communication links. The nodes that are circled in black use a normalized step size $\tilde{\mu}_k = 1$, whereas the others use $\tilde{\mu}_k = 0.1$ in (55). (a) Daily average temperature measured by 100 weather stations on 06/21/2002 (°F). The arrow points to the station whose data are used in Fig. 14. (b) Clusters adopted for the multitask algorithms in Secs. 8.1 and 8.2. (c) Scenario studied in Sec. 8.2. Blue nodes are unobserved, whereas the red ones are observed. Edges represent communication links. The arrow points to the station whose data are used in Fig. 16. This figure was created using the GSPBOX toolkit [239].

We divided our dataset into training and testing sets. The former consists of $N_{\text{tr.}} = 3650$ measurements from 12/25/2001 to 12/22/2011, which were periodically replicated to form a number of training epochs, depending on the experiment. The testing set consists of the measurements from 12/23/2011 to 12/21/2012. In both periods, we consider that $d_k(n) = u_k(n+1)$, where $u_k(n)$ denotes the temperature measurement at node k and time instant n. Hence, the simulation scenario could be interpreted as follows. At the dawn of each day, the algorithms predict the average temperature for that date at all the nodes using the last measurements available and the combined estimate calculated at the end of the previous day. At the end

of the day, after the average temperature for that certain date has been calculated, the algorithms compute their estimation errors and perform their adaptation and combination steps to update their parameters $\boldsymbol{\psi}_k$ and $\mathbf{w}_k$.

In both the training and testing periods, we conduct the adaptation of the algorithms as usual. The idea behind the division of the data in these sets is to enable us to test the algorithms using data different from the ones employed in the training, and also to examine the steady-state performance along 365 days. As a performance indicator, we adopt the squared relative reconstruction error (SSRE), given by [28]

$$\text{SRRE} = \frac{1}{V}\sum_{k=1}^{V}\frac{\left[u_k(n+1)-\mathbf{u}_k^{\mathrm{T}}(n)\mathbf{w}_k(n)\right]^2}{u_k^2(n+1)}. \tag{54}$$

We consider two types of application. In Sec. 8.1, we study the behavior of the techniques studied in Secs. 3–7 in a scenario in which the temperature is measured daily at every node. On the other hand, in Sec. 8.2, we consider that some of the nodes are not capable of measuring the temperature. Thus, the data remain unobserved at these nodes throughout the training and testing sets. This model could be used to, e.g., predict the temperature in a region where there are no weather stations. To cope with the lack of information at these nodes, we consider different diffusion techniques that utilize the GSP framework.

8.1. Temporal Prediction with Fully Observed Nodes

In this scenario, we use the dNLMS, RFF-dKNLMS, multitask dNLMS, and the diffusion Graph NLMS (dGNLMS) algorithms to predict the current temperature at every station, assuming that we have access to a few past measurements at every node. We opted to use the normalized version of each algorithm due to the fact that the input signal is not white in this scenario, which could hinder the performance otherwise [74, 75]. Hence, each node k uses

$$\mu_k(n) = \frac{\widetilde{\mu}_k}{\delta + \|\mathbf{y}_k(n)\|^2}, \tag{55}$$

where $\delta > 0$ is a small regularization factor, and $\mathbf{y}_k(n) = \mathbf{u}_k(n)$ for the single-task and multitask dNLMS algorithms, $\mathbf{y}_k(n) = \mathbf{z}_k(n)$ for RFF-dKNLMS, and $\mathbf{y}_k(n) = \mathbf{x}_k(n)$ for the dGNLMS algorithm. We adopted $\widetilde{\mu}_k = 0.1$ for half of the nodes, and $\widetilde{\mu}_k = 1$ for the other half. This is depicted in Fig. 12, in which the nodes that use $\widetilde{\mu}_k = 1$ are circled in black, whereas the ones that use $\widetilde{\mu}_k = 0.1$ are not. For every node, we set $\delta = 10^{-5}$. Furthermore, instead of directly using $u_k(n)$ as the input signal for each node k, we divided by the highest temperature measured registered during the training period, so as to ensure that $|u_k(n)| \leqslant 1$ for every k, $k = 1, \cdots, V$. Later, the outputs of the algorithms were re-scaled. In every case, we adopted the ATC configuration, and the ACW algorithm of Table 2 for the selection of the combination weights $c_{ik}(n)$, with $\nu_k = 0.2$. To avoid potential problems due to a division by zero in the ACW algorithm, we also added the regularization factor δ to $\widehat{\sigma}_{ik}^2(n)$ before calculating its reciprocal, similarly to what was done in [127]. We also set $M = 5$ for every solution. For the RFF-dKNLMS algorithm, we used the Gaussian

kernel with $D = 20$. Increasing D further did not improve the performance significantly, while raising the computational cost noticeably. Furthermore, the value of $h^2 = 0.1$ was selected because it led to the best steady-state performance. For the multitask case, we considered the approach of [7] with $\eta = 0$ and $b_{ik} = 1$ if $i = k$ and $b_{ik} = 0$ otherwise. Hence, the nodes do not exchange information during the adaptation step, only in the combination step, and nodes from different clusters do not communicate with each other. In order to determine the clusters, we calculated the Wiener solution for each node and ran the k-means algorithm. Then, a few manual adjustments were made to ensure that nodes belonging to the same clusters are indeed neighbors. The resulting clusters are depicted in Fig. 12(b). For the dGNLMS algorithm, we considered the adjacency matrix given by (53) as the graph shift operator. This matrix was normalized by its greatest absolute eigenvalue, which is a common practice in the literature [28, 29]. Finally, for comparison, we also included the non-cooperative NLMS approach in the simulations. In the training period, we consider a total of 40 epochs.

In Fig. 13 we show the SRRE curves yielded by the solutions in the training period. For the sake of visualization, these curves were filtered by a moving-average filter with 2048 coefficients. In Table 4, we present the average SRRE obtained with each solution during the testing period, as well as the approximate number of multiplications required per iteration by each solution. From Fig. 13 and Table 4, we can see that the non-cooperative NLMS achieves a higher level of SRRE in steady state in comparison with all of the other solutions. The dNLMS, dGNLMS, and multitask dNLMS algorithms achieved similar performances, whereas the RFF-dKNLMS algorithm slightly outperforms them in steady-state and in the testing period. Nonetheless, it is worth noting that this improvement comes at the expense of a greater computational cost, as is evident from Table 4. Furthermore, the usage of information from neighboring nodes in the dGNLMS algorithm did not improve the performance in this particular scenario. Analyzing the behavior of the single-task and multitask dNLMS algorithms, we can see that, in this case, the communication solely within each cluster was sufficient to achieve the same performance in comparison with the case in which each node was allowed to exchange information with all of its neighbors. In other words, the multitask dNLMS algorithm achieved a similar performance in comparison with the single-task dNLMS while presenting lower communication and computational burdens, as shown in Table 4.

Lastly, in order to enable a direct comparison in terms of degrees Fahrenheit, in Fig. 14, we illustrate the behavior of the solutions by showing the estimates provided by each of them for the temperature at a single node, which is indicated by an arrow in Fig. 12(a), in the testing set. The actual temperature, i.e. the desired signal, is also shown. We can see that the non-cooperative approach produces a noisier estimate in comparison with the other solutions, which match the desired signal reasonably well.

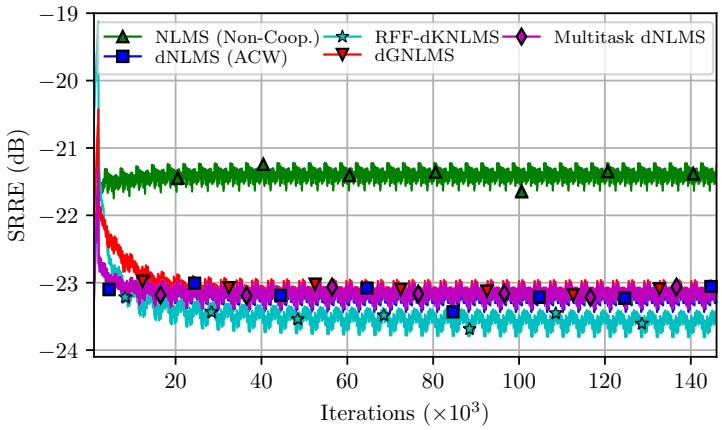

Figure 13: SRRE curves along the iterations for the training dataset obtained with the dNLMS, RFF-dKNLMS, dGNLMS, multitask dNLMS, and non-cooperative NLMS algorithms.

Table 4: Comparison between the dNLMS, RFF-dKNLMS, dGNLMS, multitask dNLMS, and non-cooperative NLMS algorithms in terms of the performance on the testing set and computational cost.

Solutions	Average SRRE in the testing set (dB)	Approximate number of multiplications per iteration ($\times 10^3$)
NLMS (Non-cooperative)	-20.6910	3.99
dNLMS (12)	-22.8933	7.90
Multitask dNLMS (32)	-22.8338	6.48
RFF-dKNLMS (47)	-23.5502	39.54
dGNLMS (52)	-22.8725	14.10

8.2. Prediction at Unobserved Nodes

In this subsection, we consider that there are 75 observed nodes, which can measure their data daily, and 25 unobserved nodes, which are not capable of performing that measurement at all. This is depicted in Fig. 12(c), in which observed nodes are shown in red, and unobserved ones are depicted in blue. This situation could be used to model a scenario in which we would like to estimate the temperature in a region in which there is no weather station, based on the measurements from the surrounding region.

In order to predict the temperature at unobserved nodes, we must rely on the information at their neighbors. To this end, we employ the dGNLMS algorithm, similarly to what was done in Sec. 8.1, as well as GSP-based versions of the multitask and kernel solutions of Secs. 5 and 6, respectively.

To cope with the unavailability of the information at the unobserved nodes, the vector $\mathbf{x}_k(n)$ of (51) is slightly modified to [234]

$$\mathbf{x}'_k(n) = \Big[[\mathbf{A}\mathbf{S}\mathbf{u}(n)]_k \; \cdots \; [\mathbf{A}^M\mathbf{S}\mathbf{u}(n-M+1)]_k \Big]^{\mathrm{T}}, \tag{56}$$

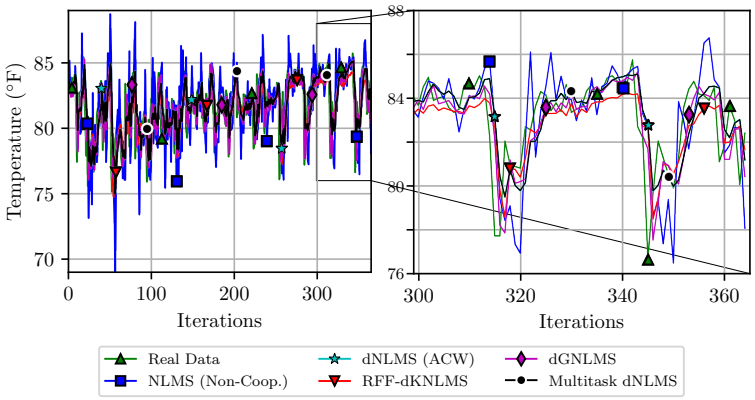

Figure 14: Comparison between the temperature measured at the station indicated by an arrow in Fig. 12 and the estimates provided by the dNLMS, RFF-dKNLMS, dGNLMS, multitask dNLMS, and non-cooperative NLMS algorithms.

where $\mathbf{S}$ is a diagonal matrix such that $[\mathbf{S}]_{kk} = 1$ if node k is observed and $[\mathbf{S}]_{kk} = 0$ otherwise.

We employ a normalized version of the RFF kernel solution of [236], resulting in the RFF-dGKNLMS algorithm. For the multitask approach, we consider the solution of (32) with the same clusters that were used in the simulations of Sec. 8.1. In comparison with the original solutions of (32) and (47), the main differences between the algorithms employed in this section and the ones studied previously lie in the usage of the vector given by (56) instead of the regressor vector $\mathbf{u}_k(n)$ in the adaptation step, and in the adoption of a normalized step size as in (55). It is worth noting that an unsupervised clustering algorithm for multitask diffusion solutions was proposed in [234]. Based on this method, several multitask algorithms were proposed in the aforementioned paper specifically for diffusion adaptive graph filtering. Due to space limitations, and since in this case we can group the nodes in clusters based on the knowledge of the Wiener solution, we have opted to restrict our analysis to the modified version of (32). Nonetheless, we encourage the reader interested in distributed GSP to consult [234] for the multitask solutions with unsupervised clustering.

Similarly to what was done in Sec. 8.1, we adopt a length of $M = 5$ for $\mathbf{u}_k(n)$. We also use normalized step size $\tilde{\mu}_k = 1$ for half of the nodes, whereas the other half employs $\tilde{\mu}_k = 0.1$, as shown in Fig. 12(b). Furthermore, we consider an ATC configuration and seek to obtain the best performance possible with each solution. To this end, we have adopted ACW to select the weights of the dGNLMS and multitask dGNLMS algorithms. For the latter, we adopted $\eta = 0$ and $b_{ik} = c_{ki}$ for every k, $k = 1, \cdots, 100$ and i, $i = 1, \cdots, 100$. Adopting $\eta \neq 0$ with weights ρ_{ik} did not affect the performance significantly. For the RFF-dGKNLMS algorithm, we adopted $h^2 = 10$ and $D = 20$. Increasing D further did not improve the results noticeably. Finally, specifically for the RFF-dGKNLMS algorithm we adopted Metropolis weights, since the adoption of ACW did not improve the performance significantly, while increasing a computational cost that was already comparatively high. We also tested multitask versions of RFF-dGKNLMS following different configurations for the weights b_{ik}, c_{ik}, ρ_{ik} and different values for η, but did not observe any significant improvement in

comparison with the single-task RFF-dGKNLMS algorithm. For this reason, these results are not depicted in the following figures and tables.

We consider 20 epochs in the training set, and evaluate the performance of each solution by applying the SRRE given by (54) to the unobserved nodes only. The resulting SRRE curves are presented in Fig. 15. For the sake of visualization, these curves were filtered by a moving-average filter with 1024 coefficients. In Table 5, we present the average SRRE measured at the unobserved nodes in the testing period, as well as the number of multiplications required by each solution. Analyzing Fig. 15, we can see that the multitask dGNLMS outperforms the single-task dGNLMS algorithm, but both are outperformed by the RFF-GKNLMS solution. The same holds in the testing set, as evidenced in Table 5. On the other hand, we can see from the same table that the computational cost of RFF-dGKNLMS is considerably greater than that of the other solutions. It is interesting to notice that the multitask dGNLMS outperforms the single-task version while presenting only a slightly higher computational burden.

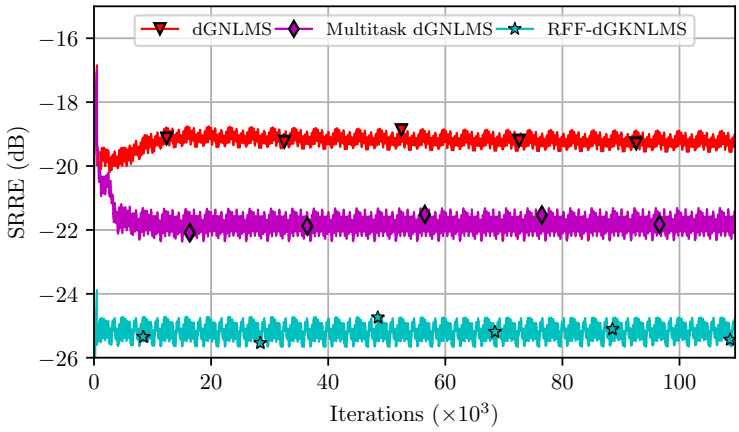

Figure 15: SRRE curves for the unobserved nodes along the iterations for the training dataset obtained with the dGNLMS, multitask dGNLMS, and RFF-dGKNLMS algorithms.

Table 5: Comparison between the dGNLMS, multitask dGNLMS, and RFF-dGKNLMS algorithms in terms of the performance the testing set and computational cost.

Solutions	Average SRRE in the testing set (dB)	Approximate number of multiplications per iteration ($\times 10^3$)
dGNLMS	-20.2872	17.17
Multitask dGNLMS (32)	-23.8701	20.14
RFF-dGKNLMS (47)	-26.2089	299.47

Finally, similarly to what was done in Fig. 14, in Fig. 16, we show the estimates provided by the dGNLMS, multitask dGNLMS, and RFF-dGKNLMS algorithms for the temperature at the unobserved node indicated

by an arrow in Fig. 12(b) in the testing set. We also show the actual temperature measured at that station. We can see that the estimates match the actual temperature signal reasonably accurately. This is especially true for the RFF-dGKNLMS algorithm, which supports the results of Fig. 15 and Table 5.

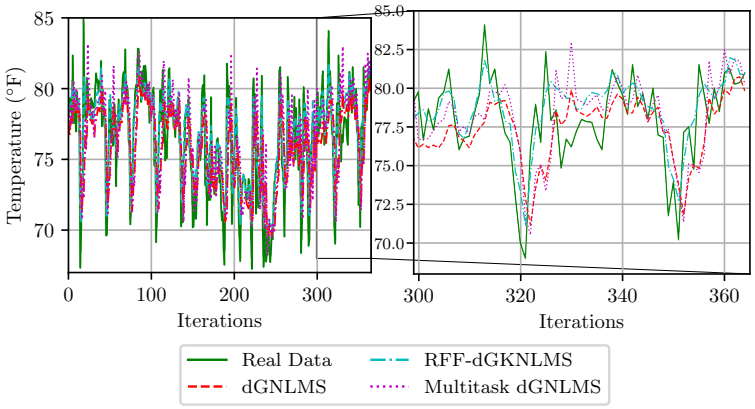

Figure 16: Comparison between the temperature measured at the station indicated by an arrow in Fig. 12(b) and the estimates provided by the dGNLMS, multitask dGNLMS, and RFF-dGKNLMS algorithms.

9. Conclusions and Open Topics for Future Work

Starting with the technological developments that motivated the emergence of adaptive diffusion networks, we have presented the progress that has occurred in the area throughout the past fifteen years or so. Some theoretical results were presented in order to provide some insights into their behavior. On the other hand, a few considerations were presented regarding the feasibility of these solutions in practical applications. In particular, significant attention was devoted to the necessity of restricting the amount of communication between nodes, even if at the expense of some deterioration in performance, due to energy constraints. We have also looked at advances that have been proposed in the literature and developed into mature research topics in their own right, such as the multitask and kernel-based diffusion networks. Furthermore, diffusion-based adaptive algorithms for Graph Signal Processing were also addressed, and the similarities and differences in comparison with the more conventional diffusion solutions were highlighted. In each case, we have highlighted what type of problems each of these solutions is intended for. A brief summary is presented in Table 6 for ease of reference. Finally, simulations carried out with real-world data exemplify the opportunities and challenges that can arise from the usage of adaptive diffusion networks in practice.

Despite the maturity of adaptive diffusion networks as a research field, there are still many open questions and challenges in the area, some of which are listed below.

1. **Energy-efficient algorithms**: as mentioned in Sec. 4, the search for restrictive communication poli-

Table 6: Summary of some of the techniques reviewed in the paper.

Type	Techniques	Comments and References
Distributed linear filtering with a common objective	Non-cooperative (Sec. 3) Diffusion (Sec. 3)	Distributed linear filtering algorithms [1–3].
Distributed linear filtering with a common objective and strict energy constraints	Diffusion strategies with restrictive communication policies (Sec. 4)	Important when the energy consumption associated with the communication processes within the network pose a severe limitation [100–130].
Distributed linear filtering with possibly many distinct goals	Multitask solutions (Sec. 5)	Aimed at problems of distributed nature in which the objective can change from one node to the other, or from a cluster of nodes to the other. [7–18].
Distributed non-linear signal processing	Diffusion kernel algorithms (Sec. 6)	Aimed at problems that cannot be satisfactorily addressed by linear solutions, and which are of distributed nature [19–27].
Distributed spatial, as well as temporal, prediction	Diffusion implementations of graph signal processing algorithms (Sec. 7)	Aimed at problems in which both spatial and temporal aspects play an important role on the development of the signals of interest over time [28–31].

cies that impact network performance as little as possible continues to spark research and publications in the area, such as, e.g., [107, 108, 128–130]. The research efforts in this area will likely continue as long as energy remains a significant constraint for battery-operated WSN's;

2. **Computational cost concerns**: this aspect is applicable to adaptive diffusion networks as a whole, due to the potentially low computational power of each node in IoT applications [127, 240–242]. For kernel-based adaptive networks, this is an especially pressing issue, due to the high computational burden associated with these tools, as can be seen from Tables 4 and 5. The reduction of the computational cost of kernel methods has been a topic of intense research in general [243–245], as well as estimating the number of RFF's required for satisfactory performance [246]. In the GSP field, the idea of sampling the nodes – i.e., measuring the data only at a subset of nodes and seeking to estimate the behavior of the network as a whole from the acquired information – has also been a topic of constant research [247–249]. This is important because the computational cost of GSP solutions increases with the number of nodes, and may become prohibitively high if the network is too large [30, 31];

3. **Opportunities for improvements in performance**: there are several fronts that could be explored to enhance the performance of adaptive diffusion networks. For multitask networks, an idea has been raised in [38] to automatically determine the optimal regularization strength η in (31), but to the best of our knowledge this has not been done yet. Moreover, adaptive clusterizarion algorithms continue to be proposed for these types of networks [250, 251]. For kernel adaptive networks, multikernel solutions [228, 252] also constitute opportunities for future research.

4. **Privacy of the data**: concerns regarding the potential leakage of data in distributed learning have lead to proposals of schemes to mitigate the associated risks in adaptive diffusion networks [253–255]. As concern regarding data privacy increases in the machine learning and signal processing communities as a whole, this topic has the potential to gather widespread attention;

5. **Extensions of other tools**: as mentioned in Sec. 2.4, in the past few years, many concepts of adaptive diffusion networks have been applied to the distributed training of other machine learning solutions, such as neural networks and generative adversarial networks (GANs) [162–165]. Hence, the incorporation of core ideas of adaptive diffusion networks into other solutions may become yet another topic for intense research in the near future.

Evidently, these are only some of the many open challenges and opportunities for research in the adaptive diffusion networks. Advances in these topics, as well as in other technologies such as 5G communication networks, wearable technology, and low-power sensors, will most likely contribute to increase the interest in adaptive diffusion networks over the coming years.

Acknowledgment

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.

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. 323–453. Academic Press, 2014. [pdf] Also available as: arXiv:1205.4220 [cs.MA], May 2012.

[3] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, 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 Transactions on Signal Processing*, vol. 56, pp. 3122–3136, Jun. 2008.

[5] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, 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 Transactions on Signal Processing*, vol. 56, pp. 1865–1877, Apr. 2008.

[7] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Transactions on Signal Processing*, vol. 62, pp. 4129–4144, Jun. 2014.

[8] J. Chen, C. Richard, A. O. Hero, and A. H. Sayed, "Diffusion LMS for multitask problems with overlapping hypothesis subspaces," in *Proc. IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2014, pp. 1–6.

[9] J. Chen, C. Richard, and A. H. Sayed, "Diffusion LMS over multitask networks," *IEEE Transactions on Signal Processing*, vol. 63, pp. 2733–2748, Mar. 2015.

[10] J. Plata-Chaves, N. Bogdanović, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," *IEEE Transactions on Signal Processing*, vol. 63, pp. 3448–3460, Apr. 2015.

[11] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Proximal multitask learning over networks with sparsity-inducing coregularization," *IEEE Transactions on Signal Processing*, vol. 64, pp. 6329–6344, Aug. 2016.

[12] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks with common latent representations," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, pp. 563–579, Feb. 2017.

[13] V. C. Gogineni and M. Chakraborty, "Partial diffusion affine projection algorithm over clustered multitask networks," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019, pp. 1–5.

[14] D. Jin, J. Chen, C. Richard, and J. Chen, "Online proximal learning over jointly sparse multitask networks with $\ell_{\infty,1}$ regularization," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6319–6335, Sep. 2020.

[15] R. Nassif, S. Vlaski, C. Richard, and A. H. Sayed, "Learning over multitask graphs – part I: Stability analysis," *IEEE Open Journal of Signal Processing*, vol. 1, pp. 28–45, Apr. 2020.

[16] R. Nassif, S. Vlaski, C. Richard, and A. H. Sayed, "Learning over multitask graphs – part II: Performance analysis," *IEEE Open Journal of Signal Processing*, vol. 1, pp. 46–63, Apr. 2020.

[17] V. C. Gogineni, S. P. Talebi, and S. Werner, "Performance of clustered multitask diffusion LMS suffering from inter-node communication delays," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, pp. 2695–2699, Jan. 2021.

[18] S. Marano and A. H. Sayed, "Decision learning and adaptation over multi-task networks," *IEEE Transactions on Signal Processing*, vol. 69, pp. 2873–2887, May 2021.

[19] W. Gao, J. Chen, C. Richard, and J. Huang, "Diffusion adaptation over networks with kernel least-mean-square," in *Proc. IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2015, pp. 217–220.

[20] B.-S. Shin, H. Paul, and A. Dekorsy, "Distributed kernel least squares for nonlinear regression applied to sensor networks," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1588–1592.

[21] S. Chouvardas and M. Draief, "A diffusion kernel LMS algorithm for nonlinear adaptive networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (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 Transactions on Signal Processing*, vol. 66, pp. 5505–5519, Aug. 2018.

[23] W. Gao, J. Chen, and L. Zhang, "Diffusion approximated kernel least mean P-power algorithm," in *Proc. IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2019, pp. 1–6.

[24] P. Bouboulis, S. Chouvardas, and S. Theodoridis, "Online distributed learning over networks in RKH spaces using random Fourier features," *IEEE Transactions on Signal Processing*, vol. 66, pp. 1920–1932, Apr. 2018.

[25] 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*, D. Comminiello and J. C. Príncipe, Eds., chapter 7, pp. 149–172. Elsevier, 2018.

[26] V. R. M. Elias, V. C. Gogineni, W. A. Martins, and S. Werner, "Adaptive graph filters in reproducing kernel Hilbert spaces: Design and performance analysis," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 62–74, Dec. 2020.

[27] R. Mitra and G. Kaddoum, "Random Fourier feature-based deep learning for wireless communications," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, pp. 468–479, Apr. 2022.

[28] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "Distributed diffusion adaptation over graph signals," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4129–4133.

[29] F. Hua, R. Nassif, C. Richard, H. Wang, and A. H. Sayed, "A preconditioned graph diffusion LMS for adaptive graph signal processing," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2018, pp. 111–115.

[30] P. Di Lorenzo, P. Banelli, S. Barbarossa, and S. Sardellitti, "Distributed adaptive learning of graph signals," *IEEE Transactions on Signal Processing*, vol. 65, pp. 4193–4208, May 2017.

[31] 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. European Signal Processing Conference (EUSIPCO)*, 2017, pp. 2289–2293.

[32] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning – part I: Algorithm development," *IEEE Transactions on Signal Processing*, vol. 67, pp. 708–723, Oct. 2018.

[33] Kun Yuan, Bicheng Ying, Xiaochuan Zhao, and Ali H Sayed, "Exact diffusion for distributed optimization and learning—part II: Convergence analysis," *IEEE Transactions on Signal Processing*, vol. 67, pp. 724–739, Oct. 2018.

[34] V. Bordignon, V. Matta, and A. H. Sayed, "Adaptive social learning," *IEEE Transactions on Information Theory*, vol. 67, pp. 6053–6081, Jul. 2021.

[35] Y. Inan, M. Kayaalp, E. Telatar, and A. H. Sayed, "Social learning under randomized collaborations," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 115–120.

[36] P. Hu, V. Bordignon, S. Vlaski, and A. H. Sayed, "Optimal combination policies for adaptive social learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 5842–5846.

[37] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks," vol. 30, pp. 155–171, May 2013.

[38] R. Nassif, S. Vlaski, C. Richard, J. Chen, and A. H. Sayed, "Multitask learning over graphs: An approach for distributed, streaming machine learning," *IEEE Signal Processing Magazine*, vol. 37, pp. 14–25, May 2020.

[39] S. Modalavalasa, U. K. Sahoo, A. K. Sahoo, and S. Baraha, "A review of robust distributed estimation strategies over wireless sensor networks," *Signal Processing*, vol. 188, pp. 108150, Nov. 2021.

[40] S. Vlaski, S. Kar, A. H. Sayed, and J. M. F. Moura, "Networked signal and information processing: Learning by multiagent systems," *IEEE Signal Processing Magazine*, vol. 40, pp. 92–105, Jul. 2023.

[41] G. J. Pottie, "Wireless sensor networks," in *Proc. IEEE Information Theory Workshop (Cat. No. 98EX131)*, 1998, pp. 139–140.

[42] J. C. Haartsen, "The Bluetooth radio system," *IEEE Personal Communications*, vol. 7, pp. 28–36, Feb. 2000.

[43] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 wireless local area networks," *IEEE Communications Magazine*, vol. 35, pp. 116–126, Sep. 1997.

[44] J. Lansford, A. Stephens, and R. Nevo, "Wi-Fi (802.11b) and Bluetooth: enabling coexistence," *IEEE Network*, vol. 15, pp. 20–27, Sept.–Oct. 2001.

[45] "First 3G mobiles launched in japan." BBC News. `http://news.bbc.co.uk/2/hi/business/1572372.stm` (accessed Mar. 12, 2023).

[46] K. Bult, A. Burstein, D. Chang, M. Dong, M. Fielding, E. Kruglick, J. Ho, F. Lin, T. H. Lin, W. J. Kaiser, et al., "Low power systems for wireless microsensors," in *Proc. IEEE International Symposium on Low Power Electronics and Design*, 1996, pp. 17–21.

[47] M. J. Dong, K. G. Yung, and W. J. Kaiser, "Low power signal processing architectures for network microsensors," in *Proc. IEEE International Symposium on Low Power Electronics and Design*, 1997, pp. 173–177.

[48] T.-H. Lin, H. Sanchez, R. Rofougaran, and W. J. Kaiser, "CMOS front end components for micropower RF wireless systems," in *Proc. IEEE International Symposium on Low Power Electronics and Design*, 1998, pp. 11–15.

[49] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, pp. 51–58, May 2000.

[50] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, pp. 102–114, Aug. 2002.

[51] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, 2001, vol. 4, pp. 2033–2036.

[52] J. M. Rabaey, M. J. Ammer, J. L. Da Silva, D. Patel, and S. Roundy, "Picoradio supports ad hoc ultra-low power wireless networking," *Computer*, vol. 33, pp. 42–48, Jul. 2000.

[53] S. C. Ergen, "Zigbee/IEEE 802.15. 4 summary," UC Berkeley, Berkeley, CA, USA, vol. 10, pp. 11, Sep. 2004.

[54] D. Carlson, M. Shamsi, T. Schnaare, D. Daugherty, J. Potter, M. Nixon, et al., "IEC 62591 WirelessHART® system engineering guide," *Revision 3.0 ed.: Emerson Process Management*, 2012.

[55] D. Sexton, "SP100. 11a overview," *DOE Award DE-FC36-02GO14001, GE Global Research, Research Triangle Park, NC*, 2007.

[56] D. Culler, and S. Chakrabarti, "6LoWPAN: Incorporating IEEE 802.15. 4 into the IP architecture." Paris, France, IPSO Alliance, White Paper, 2009.

[57] J. B. Predd, S. B. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, pp. 56–69, Jul. 2006.

[58] C. Lopes and A. H. Sayed, "Distributed processing over adaptive networks," in *Proc. Adaptive Sensor Array Processing Workshop*, 2006, pp. 1-5.

[59] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2006, vol. 3, pp. 584–587.

[60] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences*, vol. E90-A, pp. 1504–1510, Aug. 2007.

[61] J. K. Tugnait, H. Liu, G. Gong, and T. Li, "Editorial," *EURASIP Journal on Wireless Communications and Networking*, vol. 2004, Dec. 2004.

[62] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, pp. 268–276, Mar. 2001.

[63] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, "Graph structure in the web," *Computer Networks*, vol. 33, pp. 309–320, Jun. 2000.

[64] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *ACM SIGCOMM Computer Communication Review*, vol. 29, pp. 251–262, Oct. 1999.

[65] "Dimensions publications analytical views." Dimensions. `https://app.dimensions.ai/analytics/publication/overview/timeline?search_mode=content&search_text=wireless%20sensor%20networks&search_type=kws&search_field=text_search&year_from=1990&year_to=2022` (accessed Mar. 22, 2024)

[66] British Broadcasting Company. "First 3G mobiles launched in Japan,". BBC News. `http://news.bbc.co.uk/2/hi/business/1572372.stm` (accessed Mar. 22, 2024).

[67] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no.1, pp. 65–78, Sep. 2004.

[68] S. Boyd L. Xiao and S. Lall, "A space-time diffusion scheme for peer-to-peer least-squares estimation," in *Proc.*

International Conference on Information Processing in Sensor Networks, 2006, pp. 168–176.

[69] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proc. IEEE Conference on Decision and Control*, 2005, pp. 6698–6703.

[70] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSN's with noisy links – part I: Distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, pp. 350–364, Jan. 2008.

[71] I. D. Schizas, G. B. Giannakis, S. I. Roumeliotis, and A. Ribeiro, "Consensus in ad hoc WSN's with noisy links—part II: Distributed estimation and smoothing of random signals," *IEEE Transactions on Signal Processing*, vol. 56, pp. 1650–1666, Apr. 2008.

[72] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2006, vol. III, pp. 584–587.

[73] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no.8, pp. 4064–4077, Aug. 2007.

[74] A. H. Sayed, *Adaptive Filters*, John Wiley & Sons, NJ, 2008.

[75] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Upper Saddle River, 4th edition, 2002.

[76] P. Di Lorenzo, S. Barbarossa and A. H. Sayed, "Decentralized resource assignment in cognitive networks based on swarming mechanisms over random graphs," *IEEE Transactions on Signal Processing*, vol. 60, pp. 3755–3769, Jul. 2012.

[77] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 2–16, Feb. 2003.

[78] M. Alanyali, S. Venkatesh, O. Savas, and S. Aeron, "Distributed Bayesian hypothesis testing in sensor networks," in *Proc. American control conference*, 2004, vol. 6, pp. 5369–5374.

[79] V. Delouille, R. Neelamani, and R. Baraniuk, "Robust distributed estimation in sensor networks using the embedded polygons algorithm," in *Proc. International Symposium on Information Processing in Sensor Networks*, 2004, pp. 405–413.

[80] Z.-Q. Luo, "An isotropic universal decentralized estimation scheme for a bandwidth constrained ad hoc sensor network," *IEEE Journal on selected areas in communications*, vol. 23, no. 4, pp. 735–744, Apr. 2005.

[81] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, pp. 1520–1533, Sep. 2004.

[82] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. International Symposium on Information Processing in Sensor Networks (ISPSN)*, 2005, pp. 63–70.

[83] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Dynamic consensus on mobile networks," in *IFAC World Congress*, 2005, pp. 1–6.

[84] S. Barbarossa and G. Scutari, "Bio-inspired sensor network design," *IEEE Signal Processing Magazine*, vol. 24, pp. 26–35, May 2007.

[85] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 798–808, Apr. 2005.

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

[87] R. L. Berger, "A necessary and sufficient condition for reaching a consensus using DeGroot's method," *Journal of the American Statistical Association*, vol. 76, no. 374, pp. 415–418, Jun. 1981.

[88] S. Arora and B. Barak, *Computational complexity: a modern approach*, Cambridge University Press, 2009.

[89] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. IEEE Conference on Decision and Control*, 2008, pp. 4185–4190.

[90] I. D. Schizas, G. Mateos, and G. B. Giannakis, "Distributed LMS for consensus-based in-network adaptive processing,"

IEEE Transactions on Signal Processing, vol. 57, pp. 2365–2382, Jun. 2009.

[91] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, pp. 1847–1864, Nov. 2010.

[92] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007, vol. 3, pp. 917–920.

[93] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, pp. 6217–6234, Dec. 2012.

[94] R. Abdolee and B. Champagne, "Distributed blind adaptive algorithms based on constant modulus for wireless sensor networks," in *Proc. International Conference on Wireless and Mobile Communications*, 2010, pp. 303–308.

[95] N. Bogdanović, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific adaptive parameter estimation," *IEEE Transactions on Signal Processing*, vol. 62, pp. 5382–5397, Oct. 2014.

[96] L. Lu and H. Zhao, "Diffusion leaky LMS algorithm: Analysis and implementation," *Signal processing*, vol. 140, pp. 77–86, Nov. 2017.

[97] H. Yu and X. Xia, "Adaptive consensus of multi-agents in networks with jointly connected topologies," *Automatica*, vol. 48, pp. 1783–1790, Aug. 2012.

[98] F. Xiao and T. Chen, "Adaptive consensus in leader-following networks of heterogeneous linear systems," *IEEE Transactions on Control of Network Systems*, vol. 5, pp. 1169–1176, Sep. 2018.

[99] H. Zhang, X. Zhou, Z. Wang, H. Yan, and J. Sun, "Adaptive consensus-based distributed target tracking with dynamic cluster in sensor networks," *IEEE Transactions on Cybernetics*, vol. 49, pp. 1580–1591, May 2019.

[100] R. Arablouei, S. Werner, Y.-F. Huang, and K. Doğançay, "Distributed least mean-square estimation with partial diffusion," *IEEE Transactions on Signal Processing*, vol. 62, pp. 472–484, Nov. 2013.

[101] S. Xu, R. C. De Lamare, and H. V. Poor, "Dynamic topology adaptation for distributed estimation in smart grids," in *Proc. IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2013, pp. 420–423.

[102] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Trading off complexity with communication costs in distributed adaptive learning via Krylov subspaces for dimensionality reduction," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, pp. 257–273, Apr. 2013.

[103] M. O. Sayin and S. S. Kozat, "Compressive diffusion strategies over distributed networks for reduced communication load," *IEEE Transactions on Signal Processing*, vol. 62, pp. 5308–5323, Oct. 2014.

[104] S. Xu, R. C. De Lamare, and H. V. Poor, "Distributed compressed estimation based on compressive sensing," *IEEE Signal Processing Letters*, vol. 22, pp. 1311–1315, Feb. 2015.

[105] S. Gupta, A. K. Sahoo, and U. K. Sahoo, "Partial diffusion over distributed networks to reduce inter-node communication," in *Proc. IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2017, pp. 1–6.

[106] I. E. K. Harrane, R. Flamary, and C. Richard, "On reducing the communication cost of the diffusion LMS algorithm," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, pp. 100–112, Mar. 2019.

[107] M. Carpentiero, V. Matta, and A. H. Sayed, "Adaptive diffusion with compressed communication," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 5672–5676.

[108] M. Carpentiero, V. Matta, and A. H. Sayed, "Compressed distributed regression over adaptive networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[109] C. G. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 3285–3288.

[110] S. Werner, Y.-F. Huang, M. L. R. De Campos, and V. Koivunen, "Distributed parameter estimation with selective

cooperation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 2849–2852.

[111] N. Takahashi and I. Yamada, "Link probability control for probabilistic diffusion least-mean squares over resource-constrained networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 3518–3521.

[112] Ø. L. Rørtveit, J. H. Hursøy, and A. H. Sayed, "Diffusion LMS with communication constraints," in *Proc. Asilomar Conference on Signals, Systems and Computers*, 2010, pp. 1645–1649.

[113] X. Zhao and A. H. Sayed, "Single-link diffusion strategies over adaptive networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 3749–3752.

[114] S. Xu, R. C. de Lamare, and H. V. Poor, "Adaptive link selection algorithms for distributed estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, pp. 86, Oct. 2015.

[115] R. Arablouei, S. Werner, K. Doğançay, and Y.-F. Huang, "Analysis of a reduced-communication diffusion LMS algorithm," *Signal Processing*, vol. 117, pp. 355–361, Dec. 2015.

[116] F. Chen and X. Shao, "Broken-motifs diffusion LMS algorithm for reducing communication load," *Signal Processing*, vol. 133, pp. 213–218, Apr. 2017.

[117] A. Rastegarnia, "Reduced-communication diffusion RLS for distributed estimation over multi-agent networks," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, pp. 177–181, Jan. 2020.

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

[119] O. N. Gharehshiran, V. Krishnamurthy, and G. Yin, "Distributed energy-aware diffusion least mean squares: Game-theoretic learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, pp. 821–836, Oct. 2013.

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

[121] D. K. Berberidis, V. Kekatos, G. Wang, and G. B. Giannakis, "Adaptive censoring for large-scale regressions," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5475–5479.

[122] C.-K. Yu, M. Van Der Schaar, and A. H. Sayed, "Information-sharing over adaptive networks with self-interested agents," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, pp. 2–19, Mar. 2015.

[123] Z. Wang, Z. Yu, Q. Ling, D. Berberidis, and G. B. Giannakis, "Distributed recursive least-squares with data-adaptive censoring," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5860–5864.

[124] Z. Wang, Z. Yu, Q. Ling, D. Berberidis, and G. B. Giannakis, "Decentralized RLS with data-adaptive censoring for regressions over large-scale networks," *IEEE Transactions on Signal Processing*, vol. 66, pp. 1634–1648, Mar. 2018.

[125] L. Yang, H. Zhu, K. Kang, X. Luo, H. Qian, and Y. Yang, "Distributed censoring with energy constraint in wireless sensor networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6428–6432.

[126] L. Yang, H. Zhu, H. Wang, K. Kang, and H. Qian, "Data censoring with network lifetime constraint in wireless sensor networks," *Digital Signal Processing*, vol. 92, pp. 73–81, Sep. 2019.

[127] D. G. Tiglea, R. Candido, and M. T. M. Silva, "A low-cost algorithm for adaptive sampling and censoring in diffusion networks," *IEEE Transactions on Signal Processing*, vol. 69, pp. 58–72, Jan. 2021.

[128] 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," *Digital Signal Processing*, vol. 127, pp. 103587, Jul. 2022.

[129] P. Xu, Y. Wang, X. Chen, and Z. Tian, "COKE: Communication-censored decentralized kernel learning," *Journal of Machine Learning Research*, vol. 22, pp. 1–35, Jan. 2021.

[130] D. G. Tiglea, R. Candido, L. A. Azpicueta-Ruiz, and M. T. M. Silva, "Reducing the communication and computational cost of random Fourier features kernel LMS in diffusion networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[131] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part I: Modeling and stability analysis," *IEEE Transactions on Signal Processing*, vol. 63, pp. 811–826, Feb. 2015.

[132] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part II: Performance analysis," *IEEE Transactions on Signal Processing*, vol. 63, pp. 827–842, Feb. 2015.

[133] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks – part III: Comparison analysis," *IEEE Transactions on Signal Processing*, vol. 63, pp. 843–858, Feb. 2015.

[134] W. Liu, J. C. Principe, and S. Haykin, *Kernel adaptive filtering: a comprehensive introduction*, vol. 57, John Wiley & Sons, 2011.

[135] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, pp. 2275–2285, Aug. 2004.

[136] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, pp. 1058–1067, Mar. 2009.

[137] S. N. Simić and S. Sastry, "Distributed environmental monitoring using random sensor networks," in *Information Processing in Sensor Networks*. Springer, 2003, pp. 582–592.

[138] X. Li, Q. Shi, S. Xiao, S. Duan, and F. Chen, "A robust diffusion minimum kernel risk-sensitive loss algorithm over multitask sensor networks," *Sensors*, vol. 19, pp. 2339, May 2019.

[139] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Multitask diffusion adaptation over asynchronous networks," *IEEE Transactions on Signal Processing*, vol. 64, pp. 2835–2850, Jun. 2016.

[140] P. M. Djurić, "Editorial," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, pp. 1–1, 2015.

[141] S. O'Dea, "Number of smartphones sold to end users worldwide from 2007 to 2021 (in million units)," Statista. `https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/` (accessed on Mar. 22, 2024).

[142] A. Perrin, "Social media usage," *Pew research center*, vol. 125, pp. 52–68, Oct. 2015.

[143] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, A. H. Byers, et al., *Big data: The next frontier for innovation, competition, and productivity*, McKinsey Global Institute, 2011.

[144] M. Zwolenski and L. Weatherill, "The digital universe: Rich data and the increasing value of the internet of things," *Journal of Telecommunications and the Digital Economy*, vol. 2, p. 47, Sep. 2014.

[145] A. Osseiran, J. F. Monserrat, and P. Marsch, *5G mobile and wireless communications technology*, Cambridge University Press, 2016.

[146] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, "A review on internet of things (IoT), internet of everything (IoE) and internet of nano things (IoNT)," in *Proc. Internet Technologies and Applications (ITA)*, 2015, pp. 219–224.

[147] S. Kumar and Z. Raza, "Internet of things: possibilities and challenges," *International Journal of Systems and Service-Oriented Engineering (IJSSOE)*, vol.7, pp. 1–24, Jul. 2017.

[148] B. Latré, B. Braem, I. Moerman, C. Blondia, and P. Demeester, "A survey on wireless body area networks," *Wireless Networks*, vol. 17, pp. 1–18, Nov. 2010.

[149] R. Negra, I. Jemili, and A. Belghith, "Wireless body area networks: Applications and technologies," *Procedia Computer Science*, vol. 83, pp. 1274–1281, 2016.

[150] D. P. Tobón, T. H. Falk, and M. Maier, "Context awareness in WBANs: a survey on medical and non-medical applications," *IEEE Wireless Communications*, vol. 20, pp. 30–37, Aug. 2013.

[151] A. Bertrand, "Distributed signal processing for wireless eeg sensor networks," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, pp. 923–935, Nov. 2015.

[152] A. Z. Abbasi, N. Islam, Z. A. Shaikh, et al., "A review of wireless sensors and networks' applications in agriculture," *Computer Standards & Interfaces*, vol. 36, pp. 263–270, Feb. 2014.

[153] J. Plata-Chaves, A. Bertrand, M. Moonen, S. Theodoridis, and A. M. Zoubir, "Heterogeneous and multitask wireless sensor networks—algorithms, applications, and challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, pp. 450–465, Apr. 2017.

[154] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, pp. 1644–1656, Apr. 2013.

[155] P. Di Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, "Adaptive least mean squares estimation of graph signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, pp. 555–568, Dec. 2016.

[156] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, pp. 83–98, May 2013.

[157] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, pp. 808–828, May 2018.

[158] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, pp. 44–63, May 2019.

[159] G. B Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proceedings of the IEEE*, vol. 106, pp. 787–807, May 2018.

[160] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Processing Magazine*, vol. 31, pp. 80–90, Sep. 2014.

[161] J. M. F. Moura, "Chapter 8 - Graph Signal Processing," in *Cooperative and Graph Signal Processing*, Petar M Djurić and Cédric Richard, Eds., pp. 239–259. Academic Press, 2018.

[162] B. Liu, Z. Ding, and C. Lv, "Distributed training for multi-layer neural networks by consensus," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, pp. 1771–1778, May 2020.

[163] B. Liu and Z. Ding, "Distributed heuristic adaptive neural networks with variance reduction in switching graphs," *IEEE Transactions on Cybernetics*, vol. 51, pp. 3836–3844, Jul. 2021.

[164] S. Vlaski and A. H. Sayed, "Competing adaptive networks," in *Proc. IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 71–75.

[165] Z. Wang, F. R. M. Pavan, and A. H. Sayed, "Decentralized GAN training through diffusion learning," in *Proc. IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2022, pp. 1–6.

[166] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, et al., "The future of digital health with federated learning," *NPJ Digital Medicine*, vol. 3, pp. 1–7, Sep. 2020.

[167] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments—part I: Agreement at a linear rate," *IEEE Transactions on Signal Processing*, vol. 69, pp. 1242–1256, Jan. 2021.

[168] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments—part II: Polynomial escape from saddle-points," *IEEE Transactions on Signal Processing*, vol. 69, pp. 1257–1270, Jan. 2021.

[169] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, Oct. 2016.

[170] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, Oct. 2016.

[171] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, pp. 50–60, May 2020.

[172] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Communications Magazine*, vol. 58, pp. 46–51, Jan. 2020.

[173] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan, et al., "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[174] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 58, pp. 4795–4810, Sep. 2010.

[175] A. Bertrand, M. Moonen, and A. H. Sayed, "Diffusion bias-compensated RLS estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 59, pp. 5212–5224, Nov. 2011.

[176] R. Arablouei, K. Doğançay, S. Werner, and Y.-F. Huang, "Adaptive distributed estimation based on recursive least-squares and partial diffusion," *IEEE Transactions on Signal Processing*, vol. 62, pp. 3510–3522, Jul. 2014.

[177] Z. Liu, Y. Liu, and C. Li, "Distributed sparse recursive least-squares over networks," *IEEE Transactions on Signal Processing*, vol. 62, pp. 1386–1395, Mar. 2014.

[178] S. A. Baqi, A. Zerguine, and M. O. B. Saeed, "Diffusion normalized least mean squares over wireless sensor networks," in *Proc. International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013, pp. 1454–1457.

[179] S. M. Jung, J.-H. Seo, and P. G. Park, "A variable step size diffusion normalized least-mean-square algorithm with a combination method based on mean-square deviation," *Circuits, Systems, and Signal Processing*, vol. 34, pp. 3291–3304, Feb. 2015.

[180] L. Li and J. A. Chambers, "Distributed adaptive estimation based on the APA algorithm over diffusion networks with changing topology," in *Proc. IEEE Workshop on Statistical Signal Processing (SSP)*, 2009, pp. 757–760.

[181] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, pp. 4692–4707, Oct. 2011.

[182] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004, pp. 20–27.

[183] Z. Li and S. Guan, "Diffusion normalized Huber adaptive filtering algorithm," *Journal of the Franklin Institute*, vol. 355, pp. 3812–3825, May 2018.

[184] Y. Yu, H. Zhao, W. Wang, and L. Lu, "Robust diffusion Huber-based normalized least mean square algorithm with adjustable thresholds," *Circuits, Systems, and Signal Processing*, vol. 39, pp. 2065–2093, Apr. 2020.

[185] I. Markovsky and S. Van Huffel, "Overview of total least-squares methods," *Signal Processing*, vol. 87, pp. 2283–2302, Oct. 2007.

[186] R. Arablouei, S. Werner, and K. Doğançay, "Diffusion-based distributed adaptive estimation utilizing gradient-descent total least-squares," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 5308–5312.

[187] C. Li, S. Huang, Y. Liu, and Y. Liu, "Distributed TLS over multitask networks with adaptive intertask cooperation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, pp. 3036–3052, Dec. 2016.

[188] Z. Wang, L. Jia, and Z. Yang, "Multi-task total least-squares adaptation over networks," in *Proc. Chinese Control Conference (CCC)*, 2018, pp. 4300–4304.

[189] L. Li, H. Zhao, and S. Lv, "Diffusion recursive total least square algorithm over adaptive networks and performance analysis," *Signal Processing*, vol. 182, pp. 107954, May 2021.

[190] H. Zhao, Y. Chen, and S. Lv, "Robust diffusion total least mean m-estimate adaptive filtering algorithm and its performance analysis," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, pp. 654–658, Feb. 2022.

[191] V. Matta, P. Braca, S. Marano, and A. H. Sayed, "Diffusion-based adaptive distributed detection: Steady-state performance in the slow adaptation regime," *IEEE Transactions on Information Theory*, vol. 62, pp. 4710–4732, Aug. 2016.

[192] V. Matta, P. Braca, S. Marano, and A. H. Sayed, "Distributed detection over adaptive networks: Refined asymptotics and the role of connectivity," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, pp. 442–460, Dec. 2016.

[193] S. Al-Sayed, J. Plata-Chaves, M. Muma, M. Moonen, and A. M. Zoubir, "Node-specific diffusion LMS-based distributed detection over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 66, pp. 682–697, Feb. 2018.

[194] A. E. Feitosa, V. H. Nascimento, and C. G. Lopes, "Adaptive detection in distributed networks using maximum likelihood detector," *IEEE Signal Processing Letters*, vol. 25, pp. 974–978, Jul. 2018.

[195] Y. Liu, C. Li, and Z. Zhang, "Diffusion sparse least-mean squares over networks," *IEEE Transactions on Signal Processing*, vol. 60, pp. 4480–4485, Aug. 2012.

[196] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity promoting adaptive algorithm for distributed learning," *IEEE Transactions on Signal Processing*, vol. 60, pp. 5412–5425, Oct. 2012.

[197] P. Di Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 61, pp. 1419–1433, Mar. 2013.

[198] P. Di Lorenzo, "Diffusion adaptation strategies for distributed estimation over Gaussian Markov random fields," *IEEE Transactions on Signal Processing*, vol. 62, pp. 5748–5760, Nov. 2014.

[199] B. Ying, K. Yuan, and A. H. Sayed, "Supervised Learning Under Distributed Features," *IEEE Transactions on Signal Processing*, vol. 67, pp. 977–992, Feb. 2019.

[200] Y. Liu, X. Zhang, Y. Kang, L. Li, T. Chen, M. Hong, and Q. Yang, "FedBCD: A Communication-Efficient Collaborative Learning Framework for Distributed Features," *IEEE Transactions on Signal Processing*, vol. 70, pp. 4277–4290, Aug. 2022.

[201] C. A. Musluoglu, and A. Bertrand, "A Unified Algorithmic Framework for Distributed Adaptive Signal and Feature Fusion Problems – Part I: Algorithm Derivation," *IEEE Transactions on Signal Processing*, vol. 71, pp. 1863–1878, May 2023.

[202] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. IEEE Conference on Decision and Control European Control Conference (CDC-ECC)*, 2005, pp. 2996–3000.

[203] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, pp. 1087–1092, Jun. 1953.

[204] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, Apr. 1970.

[205] X. Zhao and A. H. Sayed, "Performance limits for distributed estimation over LMS adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, pp. 5107–5124, Oct. 2012.

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

[207] X. Zhao, S.-Y. Tu, and A. H. Sayed, "Diffusion adaptation over networks under imperfect information exchange and non-stationary data," *IEEE Transactions on Signal Processing*, vol. 60, pp. 3460–3475, Jul. 2012.

[208] C.-K. Yu and A. H. Sayed, "A strategy for adjusting combination weights over adaptive networks," in *Proc. IEEE*

International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013, pp. 4579–4583.

[209] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Review*, vol. 46, pp. 667–689, Dec. 2004.

[210] J. Fernandez-Bes, J. Arenas-García, and A. H. Sayed, "Adjustment of combination weights over adaptive diffusion networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6409–6413.

[211] J. Fernandez-Bes, L. A. Azpicueta-Ruiz, J. Arenas-García, and M. T. M. Silva, "Distributed estimation in diffusion networks using affine least-squares combiners," *Digital Signal Processing*, vol. 36, pp. 1–14, Jan. 2015.

[212] A. Nakai and K. Hayashi, "An adaptive combination rule for diffusion LMS based on consensus propagation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 3839–3843.

[213] J. Fernandez-Bes, J. Arenas-Garcia, M. T. M. Silva, and L. A. Azpicueta-Ruiz, "Adaptive diffusion schemes for heterogeneous networks," *IEEE Transactions on Signal Processing*, vol. 65, pp. 5661–5674, Nov. 2017.

[214] C. C. Moallemi and B. Van Roy, "Consensus propagation," *IEEE Transactions on Information Theory*, vol. 52, pp. 4753–4766, Nov. 2006.

[215] R. Abdolee and V. Vakilian, "An iterative scheme for computing combination weights in diffusion wireless networks," *IEEE Wireless Communications Letters*, vol. 6, pp. 510–513, Aug. 2017.

[216] C. G. Lopes, L. F. O. Chamon, and V. H. Nascimento, "Towards spatially universal adaptive diffusion networks," in *Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 803–807.

[217] J.-H. Seo, S. M. Jung, and P.G. Park, "A diffusion subband adaptive filtering algorithm for distributed estimation using variable step size and new combination method based on the MSD," *Digital Signal Processing*, vol. 48, pp. 361–369, Jan. 2016.

[218] Y. E. Erginbas, S. Vlaski, and A. H. Sayed, "Gramian-based adaptive combination policies for diffusion learning over networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5215–5219.

[219] M. Ergen and P. Varaiya, "Decomposition of energy consumption in IEEE 802.11," in *Proc. IEEE International Conference on Communications*, 2007, pp. 403–408.

[220] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2001, vol. 3, pp. 1548–1557.

[221] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," *Proc. Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[222] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. Advances in Neural Information Processing Systems*, 2018, pp.4448–4459.

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

[224] S. Scardapane, J. Chen, and C. Richard, "Adaptation and learning over networks for nonlinear system modeling," in *Adaptive Learning Methods for Nonlinear System Modeling*, D. Comminiello and J. C. Príncipe, Eds., chapter 10, pp. 223–242. Butterworth-Heineman, 2018.

[225] B. Scholkopf and A. J. Smola, *Learning with Kernels: support vector machines, regularization, optimization, and beyond*, MIT Press, Cambridge, 2002.

[226] I. Steinwart and A. Christmann, *Support vector machines*, Springer Science & Business Media, 2008.

[227] B.-S. Shin, H. Paul, M. Yukawa, and A. Dekorsy, "Distributed nonlinear regression using in-network processing with multiple Gaussian kernels," in *Proc. IEEE International Workshop on Signal Processing Advances in Wireless Commu-*

nications (SPAWC), 2017, pp. 1–5.

[228] S. Hong and J. Chae, "Distributed online learning with multiple kernels," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, pp. 1263-1277, Aug. 2021.

[229] B.-S. Shin, M. Yukawa, R. L. G. Cavalcante, and A. Dekorsy, "A hybrid dictionary approach for distributed kernel adaptive filtering in diffusion networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 3414–3418.

[230] P. Bouboulis, S. Pougkakiotis, and S. Theodoridis, "Efficient KLMS and KRLS algorithms: A random Fourier feature perspective," in *Proc. IEEE Statistical Signal Processing Workshop (SSP)*, 2016, pp. 1–5.

[231] X. Dong, D. Thanou, L. Toni, M. Bronstein, and P. Frossard, "Graph signal processing for machine learning: A review and new perspectives," *IEEE Signal Processing Magazine*, vol. 37, pp. 117–127, Oct. 2020.

[232] P. Latouche and F. Rossi, "Graphs in machine learning: an introduction," in *Proc. European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2015, pp. 207–218.

[233] J. A. Bondy and U. S. R. Murty, *Graph Theory With Applications*, Macmillan Press Ltd, 1976.

[234] F. Hua, R. Nassif, C. Richard, H. Wang, and A. H. Sayed, "Online distributed learning over graphs with multitask graph-filter models," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 63–77, 2020.

[235] A. Alinaghi, S. Weiss, V. Stankovic, and I. Proudler, "Graph filter design for distributed network processing: a comparison between adaptive algorithms," in *Proc. Sensor Signal Processing for Defence Conference (SSPD)*, 2021, pp. 1–5.

[236] V. C. Gogineni, V. R. M. Elias, W. A. Martins, and S. Werner, "Graph diffusion kernel LMS using random Fourier features," in *Proc. Asilomar Conference on Signals, Systems, and Computers*, 2020, pp. 1528–1532.

[237] V. R. M. Elias, V. C. Gogineni, W. A. Martins, and S. Werner, "Kernel regression over graphs using random Fourier features," *IEEE Transactions on Signal Processing*, vol. 70, pp. 936–949, Feb. 2022.

[238] "Historical temperature dataset," Instituto Nacional de Meteorologia (INMET). `https://portal.inmet.gov.br/` (accessed Mar. 22 2024). Data used in the simulations also available at: `https://github.com/dgtiglea/Daily-Average-Temperature-Brazilian-Stations`, Sep. 2020.

[239] N. Perraudin, J. Paratte, D. I. Shuman, V. Kalofolias, P. Vandergheynst, and D.K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *arXiv*, vol. preprint:1408.5781, Aug. 2014.

[240] A. E. Feitosa, V. H. Nascimento, and C. G. Lopes, "Low complexity distributed estimation for IoT sensor networks," in *Proc. IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 136–140.

[241] R. M. Coelho, C. G. Lopes, and H. F. Ferro, "Adaptive IIR diffusion networks for IoT applications," in *Proc. IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 141–145.

[242] C. G. Lopes, V. H. Nascimento, and L. F. O. Chamon, "Distributed universal adaptive networks," *IEEE Transactions on Signal Processing*, vol. 71, pp. 1817 – 1832, May 2023.

[243] X. Hou, H. Zhao, and X. Long, "Graph diffusion kernel maximum correntropy criterion over sensor network and its performance analysis," *IEEE Sensors Journal*, vol. 23, pp. 14583 – 14591, May 2023.

[244] K. Xiong and S. Wang, "The online random Fourier features conjugate gradient algorithm," *IEEE Signal Processing Letters*, vol. 26, pp. 740–744, Mar. 2019.

[245] A. A. Bueno and M. T. M. Silva, "Gram-Schmidt-based sparsification for kernel dictionary," *IEEE Signal Processing Letters*, vol. 27, pp. 1130–1134, Jun. 2020.

[246] Z. Li, J.-F. Ton, D. Oglic, and D. Sejdinovic, "Towards a unified analysis of random Fourier features," in *Proc. International Conference on Machine Learning*, 2019, pp. 3905–3914.

[247] D. Bacciu, F. Errica, A. Micheli, and M. Podda, "A gentle introduction to deep learning for graphs," *Neural Networks*, vol. 129, pp. 203–221, Sep. 2020.

[248] X. Wei, R. Yu, and J. Sun, "View-GCN: View-based graph convolutional network for 3D shape analysis," in *Proc.*

IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1850–1859.

[249] Y. Xie, Z. Xu, J. Zhang, Z. Wang, and S. Ji, "Self-supervised learning of graph neural networks: A unified review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, pp. 2412–2429, Apr. 2022.

[250] Q. Shi, F. Chen, X. Li, and S. Duan, "Distributed adaptive clustering learning over time-varying multitask networks," *Information Sciences*, vol. 567, pp. 278–297, Aug. 2021.

[251] J. Li, W. Wang, W. Abbas, and X. Koutsoukos, "Distributed clustering for cooperative multi-task learning networks," *IEEE Transactions on Network Science and Engineering*, vol 10, pp. 3933 – 3942, May 2023.

[252] R. A. Abbasabad and M. Azghani, "Distributed sparsity-based non-linear regression with multiple kernels in wireless sensor networks," *Ad Hoc Networks*, vol. 125, pp. 102719, Feb. 2022.

[253] I. E. K. Harrane, R. Flamary, and C. Richard, "Toward privacy-preserving diffusion strategies for adaptation and learning over networks," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1513–1517.

[254] V. C. Gogineni, A. Moradi, N. K. D. Venkategowda, and S. Werner, "Communication-efficient and privacy-aware distributed learning," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 9, pp. 705 – 720, Oct. 2023.

[255] E. Rizk, S. Vlaskiy, and A. H. Sayed, "Enforcing privacy in distributed learning with performance guarantees," *IEEE Transactions on Signal Processing*, vol. 71, pp. 3385 – 3398, Sep. 2023.