Distributed Adaptive Learning of Signals Defined over Graphs

Paolo Di Lorenzo[1], Paolo Banelli[1], Sergio Barbarossa[2], and Stefania Sardellitti[2]

[1] Department of Engineering, University of Perugia, Via G. Duranti 93, 06125, Perugia, Italy

[2] Department of Information Engineering, Electronics, and Telecommunications, Sapienza University of Rome,
Via Eudossiana 18, 00184, Rome, Italy

Email: paolo.dilorenzo@unipg.it, paolo.banelli@unipg.it,
sergio.barbarossa@uniroma1.it, stefania.sardellitti@uniroma1.it

***Abstract*—The goal of this paper is to propose adaptive strategies for distributed learning of signals defined over graphs. Assuming the graph signal to be band-limited, the method enables distributed adaptive reconstruction from a limited number of sampled observations taken from a subset of vertices. A detailed mean square analysis is carried out and illustrates the role played by the sampling strategy on the performance of the proposed method. Finally, a distributed selection strategy for the sampling set is provided. Several numerical results validate our methodology, and illustrate the performance of the proposed algorithm for distributed adaptive learning of graph signals.**

***Index Terms*—Graph signal processing, sampling on graphs, adaptation and learning over networks, distributed estimation.**

I. Introduction

Over the last few years, there was a surge of interest in the development of processing tools for the analysis of signals defined over a graph, or graph signals for short [1], [2]. Graph signal processing (GSP) considers signals defined over a discrete domain having a very general structure, represented by a graph, and subsumes classical discrete-time signal processing as a very simple case. Several processing methods for signals defined over a graph were proposed in [2], [3], [4], and one of the most interesting aspects is that these analysis tools come to depend on the graph topology. A fundamental role in GSP is played by spectral analysis, which passes through the definition of the Graph Fourier Transform (GFT), see, e.g., [1], [2], and paves the way for the development of a *sampling theory* for signals defined over graphs, whose aim is to recover a band-limited (or approximately band-limited) graph signal from a subset of its samples, see, e.g., [5]–[7]. Several reconstruction methods have been proposed, either iterative as in [8], [9], or single shot, as in [5], [6], [10]. Furthermore, as shown in [5], [6], the selection of the sampling set plays a fundamental role in the reconstruction task.

Almost all previous art considers centralized processing methods for graph signals. In many practical systems, data are collected in a distributed network, and sharing local information with a central processor is either unfeasible or not efficient, owing to the large size of the network and volume of data, time-varying network topology, bandwidth/energy constraints, and/or privacy issues. In addition, a centralized solution may limit the ability of the nodes to adapt in real-time to time-varying scenarios. Motivated by these observations, in this paper we focus on distributed techniques for graph signal processing. Some distributed methods were recently proposed in the literature, see, e.g. [11]–[13]. In this paper, we propose *distributed* strategies for *adaptive learning* of signals defined on graphs. The work merges, for the first time in the literature, the well established field of adaptation and learning over networks, see, e.g., [14], with the emerging area of graph signal processing. The proposed method exploits the graph structure that describes the observed signal and, under a band-limited assumption, enables adaptive reconstruction and tracking from a limited number of observations taken over a subset of vertices in a totally distributed fashion. A detailed mean square analysis illustrates the role of the sampling strategy on the reconstruction capability, stability, and performance of the proposed algorithm. Thus, based on these results, we also propose a distributed method to select the set of sampling nodes in an efficient manner. An interesting feature of our proposed strategy is that this subset is allowed to vary over time, provided that the *expected* sampling set satisfies specific conditions enabling signal reconstruction.

II. Graph Signal Processing Tools

In this section, we introduce some useful concepts from GSP that will be exploited along the paper. Let us consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composed of N nodes $\mathcal{V} = \{1, 2, ..., N\}$, along with a set of weighted edges $\mathcal{E} = \{a_{ij}\}_{i,j\in\mathcal{V}}$, such that $a_{ij} > 0$, if there is a link from node j to node i, or $a_{ij} = 0$, otherwise. The adjacency matrix $\mathbf{A} = \{a_{ij}\}_{i,j=1}^{N} \in \mathbb{R}^{N\times N}$ is the collection of all the weights $a_{ij}, i, j = 1, \ldots, N$. The degree of node i is $k_i := \sum_{j=1}^{N} a_{ij}$, and the degree matrix $\mathbf{K}$ is a diagonal matrix having the node degrees on its diagonal. The Laplacian matrix is defined as: $\mathbf{L} = \mathbf{K} - \mathbf{A}$. If the graph is *undirected*, the Laplacian matrix is symmetric and positive semi-definite, and admits the eigendecomposition $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H$, where $\mathbf{U}$ collects all the eigenvectors of $\mathbf{L}$ in its columns, whereas $\mathbf{\Lambda}$ contains the eigenvalues of $\mathbf{L}$. A signal $\boldsymbol{x}$ over a graph $\mathcal{G}$ is defined as a mapping from the vertex

The work of Paolo Di Lorenzo was supported by the “Fondazione Cassa di Risparmio di Perugia”.

978-1-5386-3954-2/16/$31.00 ©2016 IEEE

set to the set of complex numbers, i.e. $\boldsymbol{x} : \mathcal{V} \to \mathbb{C}$. In many applications, the signal $\boldsymbol{x}$ admits a compact representation, i.e., it can be expressed as:

$$\boldsymbol{x} = \mathbf{U}\boldsymbol{s} \tag{1}$$

where $\boldsymbol{s}$ is exactly (or approximately) sparse. As an example, in all cases where the graph signal exhibits clustering features, i.e. it is a smooth function within each cluster, but it is allowed to vary arbitrarily from one cluster to the other, the representation in (1) is compact, i.e. $\boldsymbol{s}$ is sparse. The GFT $\boldsymbol{s}$ of a signal $\boldsymbol{x}$ is defined as the projection onto the orthogonal set of eigenvectors [1], i.e.

$$\textbf{GFT:} \qquad \boldsymbol{s} = \mathbf{U}^H \boldsymbol{x}. \tag{2}$$

The GFT has been defined in alternative ways, see, e.g., [1], [2], [5]. In this paper, we follow the approach based on the Laplacian matrix, assuming an undirected graph structure, but the theory could be extended to handle directed graphs considering, e.g., a GFT as proposed in [2]. Also, we denote the support of $\boldsymbol{s}$ in (1) as $\mathcal{F} = \{i \in \{1, \ldots, N\} : s_i \neq 0\}$, and the *bandwidth* of the graph signal $\boldsymbol{x}$ is defined as the cardinality of $\mathcal{F}$, i.e. $|\mathcal{F}|$. Finally, given a subset of vertices $\mathcal{S} \subseteq \mathcal{V}$, we define a vertex-limiting operator as

$$\mathbf{D}_{\mathcal{S}} = \mathrm{diag}\{\mathbf{1}_{\mathcal{S}}\}, \tag{3}$$

where $\mathbf{1}_{\mathcal{S}}$ is the set indicator vector, whose i-th entry is equal to one, if $i \in \mathcal{S}$, or zero otherwise.

III. Distributed Learning of Graph Signals

We consider the problem of learning a (possibly time-varying) graph signal from observations taken from a subset of vertices of the graph. Let us consider a signal $\boldsymbol{x}^o = \{x_i^o\}_{i=1}^N \in \mathbb{C}^N$ defined over the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The signal is assumed to be perfectly band-limited, i.e. its spectral content is different from zero only on a limited set of indices $\mathcal{F}$. If the signal support is fixed and known beforehand, from (1), the graph signal $\boldsymbol{x}^o$ can be modeled in compact form as:

$$\boldsymbol{x}^o = \mathbf{U}_{\mathcal{F}} \boldsymbol{s}^o, \tag{4}$$

where $\mathbf{U}_{\mathcal{F}} \in \mathbb{C}^{N \times |\mathcal{F}|}$ collects the subset of columns of matrix $\mathbf{U}$ in (1) associated to the frequency indices $\mathcal{F}$, and $\boldsymbol{s}^o \in \mathbb{C}^{|\mathcal{F}| \times 1}$ is the vector of GFT coefficients of the frequency support of the graph signal $\boldsymbol{x}^o$. Let us assume that streaming and noisy observations of the graph signal are sampled over a (possibly time-varying) subset of vertices. In such a case, the observation taken by node i at time n can be expressed as:

$$y_i[n] \,=\, d_i[n]\,(x_i^o + v_i[n]) = d_i[n]\left(\boldsymbol{c}_i^H \boldsymbol{s}^o + v_i[n]\right), \tag{5}$$

$i = 1, \ldots, N$, where ^{H} denotes complex conjugate-transposition; $d_i[n] = \{0, 1\}$ is a random sampling binary coefficient, which is equal to 1 if node i is taking the observation at time n, and 0 otherwise; $v_i[n]$ is a zero-mean, spatially and temporally independent observation noise, with variance σ_i^2; also, in (5) we have used (4), where $\boldsymbol{c}_i^H \in \mathbb{C}^{1 \times |\mathcal{F}|}$ denotes the i-th row of matrix $\mathbf{U}_{\mathcal{F}}$. In the sequel, we assume that each node i has local knowledge of its corresponding regression vector $\boldsymbol{c}_i$ in (5). This is a reasonable assumption even in the distributed scenario considered in this paper. Indeed, there exist many techniques that enable the distributed computation of eigenparameters of matrices describing sparse topologies such as the Laplacian or the adjacency, see, e.g., [15], [16]. The distributed learning task consists in recovering the band-limited graph signal $\boldsymbol{x}^o$ from the noisy, streaming, and partial observations $y_i[n]$ in (5) by means of in-network processing and local exchange of information among nodes in the graph. Following a least mean squares approach [14], the reconstruction task can be formulated as the cooperative solution of the following optimization problem:

$$\min_{\boldsymbol{s}} \; \sum_{i=1}^{N} \mathbb{E}_{\,\mathrm{d,v}} \left| d_i[n] \left(y_i[n] - \boldsymbol{c}_i^H \boldsymbol{s} \right) \right|^2, \tag{6}$$

where $\mathbb{E}_{\,\mathrm{d,v}}(\cdot)$ denotes the expectation operator evaluated over the random variables $\{d_i[n]\}_{i=1}^N$ and $\{v_i[n]\}_{i=1}^N$, and we have exploited $d_i[n]^2 = d_i[n]$ for all i, n. In the rest of the paper, to avoid overcrowded symbols, we will drop the subscripts in the expectation symbol referring to the random variables. In the sequel, we first analyze the conditions that enable signal recovery from a subset of samples. Then, we introduce adaptive strategies specifically tailored for the distributed reconstruction of graph signals from a limited number of samples.

A. Conditions for Signal Reconstruction

Assuming the random sampling and observations processes $\boldsymbol{d}[n] = \{d_i[n]\}_{i=1}^N$ and $\boldsymbol{y}[n] = \{y_i[n]\}_{i=1}^N$ to be stationary, the solution of problem (6) is given by the vector $\boldsymbol{s}^o$ that satisfies the normal equations:

$$\left(\sum_{i=1}^{N} \mathbb{E}\{d_i[n]\} \boldsymbol{c}_i \boldsymbol{c}_i^H \right) \boldsymbol{s}^o = \sum_{i=1}^{N} \mathbb{E}\{d_i[n] y_i[n]\}\, \boldsymbol{c}_i. \tag{7}$$

Letting $p_i = \mathbb{E}\{d_i[n]\}$, $i = 1, \ldots, N$, be the probability that node i takes an observation at time n, from (7), it is clear that reconstruction of $\boldsymbol{s}^o$ is possible only if the matrix

$$\sum_{i=1}^{N} p_i \boldsymbol{c}_i \boldsymbol{c}_i^H = \mathbf{U}_{\mathcal{F}}^H \mathbf{P} \mathbf{U}_{\mathcal{F}} \tag{8}$$

is invertible, with $\mathbf{P} = \mathrm{diag}(p_1, \ldots, p_N)$ denoting a vertex sampling operator as (3), but weighted by the sampling probabilities $\{p_i\}_{i=1}^N$. Let us denote the expected sampling set by $\overline{\mathcal{S}} = \{i = 1, \ldots, N \,|\, p_i > 0\}$: $\overline{\mathcal{S}}$ represents the set of nodes of the graph that collect data with a probability different from zero. From (7) and (8), a necessary condition enabling reconstruction is $|\overline{\mathcal{S}}| \geq |\mathcal{F}|$, i.e., the number of nodes in the expected sampling set must be greater than equal to the signal bandwidth. However, this condition is not sufficient, because matrix $\mathbf{U}_{\mathcal{F}}^H \mathbf{P} \mathbf{U}_{\mathcal{F}}$ in (8) may loose rank, or easily become ill-conditioned, depending on the graph topology and sampling strategy (defined by $\overline{\mathcal{S}}$ and $\mathbf{P}$). To provide a condition for signal reconstruction, we proceed similarly to [6], [9]. Since $p_i > 0$ for all $i \in \overline{\mathcal{S}}$, matrix (8) is invertible if matrix $\sum_{i\in\overline{\mathcal{S}}}\boldsymbol{c}_i\boldsymbol{c}_i^H=\mathbf{U}_{\mathcal{F}}^H\mathbf{D}_{\overline{\mathcal{S}}}\mathbf{U}_{\mathcal{F}}$ has full rank, where $\mathbf{D}_{\overline{\mathcal{S}}}$ is the vertex-limiting operator that projects onto the expected sampling set $\overline{\mathcal{S}}$. Let us now introduce the operator $\mathbf{D}_{\overline{\mathcal{S}}_c}=\mathbf{I}-\mathbf{D}_{\overline{\mathcal{S}}}$, which projects onto the complement of the expected sampling set, i.e., $\overline{\mathcal{S}}_c=\{i=1,\ldots,N\,|\,p_i=0\}$. Then, exploiting $\mathbf{D}_{\overline{\mathcal{S}}_c}$ in $\mathbf{U}_{\mathcal{F}}^H\mathbf{D}_{\overline{\mathcal{S}}}\mathbf{U}_{\mathcal{F}}$, signal reconstruction is possible if $\mathbf{I}-\mathbf{U}_{\mathcal{F}}^H\mathbf{D}_{\overline{\mathcal{S}}_c}\mathbf{U}_{\mathcal{F}}$ is invertible, i.e., if condition

$$\left\|\mathbf{D}_{\overline{\mathcal{S}}_c}\mathbf{U}_{\mathcal{F}}\right\|_2<1 \tag{9}$$

is satisfied. As shown in [6], condition (9) is related to the localization properties of graph signals: It implies that there are no $\mathcal{F}$-bandlimited signals that are perfectly localized over the set $\overline{\mathcal{S}}_c$. Proceeding as in [6], it is easy to show that condition (9) is necessary and sufficient for signal reconstruction. We remark that, differently from previous works on graph signal sampling, condition (9) depends on the *expected* sampling set.

B. Adaptive Distributed Strategies

In this paper, our emphasis is on distributed, adaptive solutions, where the nodes perform the graph signal reconstruction task via online in-network processing only exchanging data between neighbors. To this aim, we employ diffusion adaptation techniques, which were largely studied in literature, see, e.g., [14]. The resulting algorithm applied to solve problem (6) is reported in Table 1, and will be termed as the Adapt-Then-Combine (ATC) diffusion strategy. The first step in (10) is an adaptation step, where the intermediate estimate $\boldsymbol{\psi}_i[n]$ is updated adopting the current observation taken by node i, i.e., $y_i[n]$, if $d_i[n]=1$ at time n. The second step is a diffusion step where the estimates $\boldsymbol{\psi}_j[n]$, from the spatial neighbors $j\in\mathcal{N}_i$, are combined through the real, non-negative, weights $\{w_{ij}\}$, which match the graph $\mathcal{G}$ and satisfy:

Table 1: ATC diffusion for graph signal learning

Data: $\boldsymbol{s}_i[0]$ chosen at random for all i; $\{w_{ij}\}_{i,j}$ satisfying (11); (sufficiently small) step-sizes $\mu_i>0$. Then, for each time $n\geq 0$ and for each node i, repeat:

$$\boldsymbol{\psi}_i[n]=\boldsymbol{s}_i[n]+\mu_i d_i[n]\boldsymbol{c}_i(y_i[n]-\boldsymbol{c}_i^H\boldsymbol{s}_i[n])$$

(adaptation step) (10)

$$\boldsymbol{s}_i[n+1]=\sum_{j\in\mathcal{N}_i}w_{ij}\boldsymbol{\psi}_j[n] \qquad \text{(diffusion step)}$$

$$x_i[n+1]=\boldsymbol{c}_i^H\boldsymbol{s}_i[n+1] \qquad \text{(reconstruction step)}$$

$$w_{ij}=0\ \text{ for } j\notin\mathcal{N}_i,\ \text{ and }\ \mathbf{W}\mathbf{1}=\mathbf{1}, \tag{11}$$

where $\mathbf{W}\in\mathbb{R}^{N\times N}$ is the matrix with individual entries $\{w_{ij}\}$, and $\mathcal{N}_i=\{j=1,\ldots,N\,|\,a_{ij}>0\}\bigcup\{i\}$ is the neighborhood of node i. Finally, given $\boldsymbol{s}_i[n+1]$, the last step produces the estimate $x_i[n+1]$ of the graph signal value at node i [cf. (5)].

IV. Mean-Square Analysis

In this section, we analyze the performance of the ATC strategy in (10) in terms of its mean-square behavior. To this aim, we introduce the error quantities $\boldsymbol{e}_i[n]=\boldsymbol{s}_i[n]-\boldsymbol{s}^o$, $i=1,\ldots,N$, and the network vector

$$\boldsymbol{e}[n]=\text{col}\{\boldsymbol{e}_1[n],\ldots,\boldsymbol{e}_N[n]\}. \tag{12}$$

We also introduce the matrices

$$\mathbf{M}=\text{diag}\{\mu_1\mathbf{I}_{|\mathcal{F}|},\ldots,\mu_N\mathbf{I}_{|\mathcal{F}|}\}, \tag{13}$$

$$\widehat{\mathbf{W}}=\mathbf{W}\otimes\mathbf{I}_{|\mathcal{F}|}, \tag{14}$$

where $\otimes$ denotes the Kronecker product operation, and the extended sampling operator

$$\widehat{\mathbf{D}}[n]=\text{diag}\left\{d_1[n]\mathbf{I}_{|\mathcal{F}|},\ldots,d_N[n]\mathbf{I}_{|\mathcal{F}|}\right\}. \tag{15}$$

We further introduce the block quantities:

$$\mathbf{Q}=\text{diag}\left\{\boldsymbol{c}_1\boldsymbol{c}_1^H,\ldots,\boldsymbol{c}_N\boldsymbol{c}_N^H\right\}, \tag{16}$$

$$\boldsymbol{g}[n]=\text{col}\{\boldsymbol{c}_1v_1[n],\ldots,\boldsymbol{c}_Nv_N[n]\}. \tag{17}$$

Then, exploiting (12)-(17), we conclude from (10) that the following relation holds for the error vector:

$$\boldsymbol{e}[n+1]=\widehat{\mathbf{W}}\left(\mathbf{I}-\mathbf{M}\widehat{\mathbf{D}}[n]\mathbf{Q}\right)\boldsymbol{e}[n]+\widehat{\mathbf{W}}\mathbf{M}\widehat{\mathbf{D}}[n]\boldsymbol{g}[n]. \tag{18}$$

This relation tells us how the network error vector evolves over time. Before moving forward, we introduce two assumptions.

Assumption 2 (Independent sampling): The sampling process $\{d_i[t]\}$ is temporally and spatially independent, for all $i=1,\ldots,N$ and $t\leq n$. ■

Assumption 3 (Small step-size): The step-sizes $\{\mu_i\}$ are sufficiently small so that terms that depend on higher-order powers of $\{\mu_i\}$ can be ignored. ■

We now proceed by illustrating the stability and steady-state performance of the proposed algorithm in (10).

A. Mean-Square Stability

The following theorem guarantees the asymptotic mean-square stability (convergence in mean and mean-square sense) of the ATC diffusion strategy (10).

Theorem 1 (mean-square stability): Assume data model (5), Assumptions 1, 2, and 3 hold. Then, for any initial condition and any choice of $\mathbf{W}$ satisfying (11) and $\mathbf{1}^T\mathbf{W}=\mathbf{1}^T$, the algorithm (10) will be mean-square stable if the sampling strategy satisfies condition (9).

Proof. See [17]. ■

Essentially, to guarantee the mean-square stability of the distributed procedure, two important conditions are necessary: (a) the network must collect samples from a sufficiently large number of nodes on average, i.e. condition (9) must hold; (b) the step-sizes μ_i must be chosen sufficiently small.

B. Steady-State Performance

After some calculations [17], assuming that the convergence conditions are satisfied, we obtain

$$\lim_{n\to\infty}\mathbb{E}\|\boldsymbol{e}[n]\|^2_{(\mathbf{I}-\mathbf{H})\boldsymbol{\sigma}}=\text{vec}\left(\widehat{\mathbf{W}}\mathbf{M}\widehat{\mathbf{P}}\mathbf{G}\mathbf{M}\widehat{\mathbf{W}}^T\right)^T\boldsymbol{\sigma}, \tag{19}$$

where we used interchangeably the notation $\|\boldsymbol{e}\|^2_{\boldsymbol{\sigma}}$ and $\|\boldsymbol{e}\|^2_{\boldsymbol{\Sigma}}$ to denote the same quantity $\boldsymbol{e}^H\boldsymbol{\Sigma}\boldsymbol{e}$, $\mathbf{G} = \mathbb{E}\left[\boldsymbol{g}[n]\boldsymbol{g}[n]^H\right]$, and

$$\mathbf{H} = \mathbb{E}\left\{\left(\mathbf{I} - \mathbf{Q}^T\widehat{\mathbf{D}}[n]\mathbf{M}\right)\widehat{\mathbf{W}}^T \otimes \left(\mathbf{I} - \mathbf{Q}\widehat{\mathbf{D}}[n]\mathbf{M}\right)\widehat{\mathbf{W}}^T\right\}.$$

From (19), letting $\widetilde{\boldsymbol{x}}[n] = \{\widetilde{x}_i[n]\}_{i=1}^N$, the mean-square deviation (MSD) is given by:

$$\begin{aligned}\text{MSD} &= \lim_{n\to\infty}\mathbb{E}\|\widetilde{\boldsymbol{x}}[n]\|^2 = \lim_{n\to\infty}\mathbb{E}\|\widetilde{\boldsymbol{s}}[n]\|^2_{\text{vec}(\mathbf{Q})} \\ &= \text{vec}\left(\widehat{\mathbf{W}}\mathbf{M}\widehat{\mathbf{P}}\mathbf{G}\mathbf{M}\widehat{\mathbf{W}}^T\right)^T(\mathbf{I}-\mathbf{H})^{-1}\boldsymbol{q}, \qquad (20)\end{aligned}$$

where $\boldsymbol{q} = \text{vec}\left(\sum_{i=1}^N \mathbf{R}_i \otimes \boldsymbol{c}_i\boldsymbol{c}_i^H\right) = \text{vec}\left(\mathbf{Q}\right)$ [cf. (16)]. In the sequel, we will confirm the validity of these theoretical expressions by comparing them with numerical results.

V. Distributed Graph Sampling Strategies

The properties of the proposed distributed algorithm in (10) for graph signal reconstruction strongly depend on the expected sampling set $\overline{\mathcal{S}}$. Thus, in this section we propose a distributed method that iteratively selects vertices from the graph in order to build an expected sampling set $\overline{\mathcal{S}}$ that enables reconstruction with a limited number of nodes, while guaranteeing good learning performance. In the sequel, we assume that the probabilities $\{p_i\}_{i=1}^N$ are known or can be locally estimated at each node. Then, to allow for distributed implementations, we consider the general selection problem:

$$\begin{aligned}\mathcal{S}^* = \arg\max_{\overline{\mathcal{S}}}\; h\left(\overline{\mathcal{S}}\right) = f\left(\sum_{i\in\overline{\mathcal{S}}}\frac{p_i}{1+\sigma_i^2}\,\boldsymbol{c}_i\boldsymbol{c}_i^H\right) \qquad (21)\\ \text{subject to} \quad |\overline{\mathcal{S}}| = M\end{aligned}$$

where $\overline{\mathcal{S}}$ is the expected sampling set; M is the given number of vertices samples to be selected; the weighting terms $p_i/(1+\sigma_i^2)$ take into account (possibly) heterogeneous sampling and noise conditions at each node; and $f(\cdot) : \mathbb{C}^{|\mathcal{F}|\times|\mathcal{F}|} \to \mathbb{R}$ is a function that measures the degree of invertibility of the matrix in its argument, e.g., the (logarithm of) pseudo-determinant [6], [9], or the minimum eigenvalue [5]. However, since the formulation in (21) translates inevitably into a selection problem, whose solution in general requires an exhaustive search over all the possible combinations, the complexity of such procedure becomes intractable also for graph signals of moderate dimensions. To cope with these issues, in table 2, we provide an efficient, albeit sub-optimal, greedy strategy that tackles the problem of selecting the (expected) sampling set in a distributed fashion. The idea underlying the proposed approach is to iteratively add to the (expected) sampling set the vertices of the graph that lead to the largest increment of the performance metric $h\left(\overline{\mathcal{S}}\right)$ in (21), in a totally distributed manner. Given the current instance of the set $\overline{\mathcal{S}}$, at Step 1, each node $j \notin \overline{\mathcal{S}}$ evaluates locally the value of the objective function $h\left(\overline{\mathcal{S}} \cup j\right)$ that the network would achieve if node j was added to $\overline{\mathcal{S}}$. Then, in step 2, the network finds the maximum among the local values computed at the previous step. This task can be easily obtained with a distributed iterative procedure as, e.g., a maximum consensus algorithm [18]. The node s^*, which has achieved the maximum value at step 2, is then added to the expected sampling set. Finally, the weighted regression vector associated to the selected node, i.e. $\sqrt{p_{s^*}/(1+\sigma_{s^*}^2)}\boldsymbol{c}_{s^*}$, is diffused over the network through a flooding process. This allows each node not belonging to the sampling set to evaluate step 1 of the algorithm at the next round. This procedure continues until the network has selected M samples. From a communication point of view, in the worst case, the procedure in Table 2 requires that each node exchanges $M\mathcal{D}(1+2|\mathcal{F}|)$ scalar values to accomplish the distributed task of sampling set selection, where $\mathcal{D}$ is the diameter of the network.

Table 2: Distributed Graph Sampling Strategy

```
Input Data : $M$, the number of samples. $\overline{\mathcal{S}} \equiv \emptyset$.
Output Data : $\overline{\mathcal{S}}$, the expected sampling set.
Function :
  while $|\overline{\mathcal{S}}| < M$
    1) Each node $j$ computes locally $h\left(\overline{\mathcal{S}} \cup j\right)$, for all $j \notin \overline{\mathcal{S}}$;
    2) Distributed selection of the maximum: find
         $s^* = \arg\max_{j\notin\overline{\mathcal{S}}}\; h\left(\overline{\mathcal{S}} \cup j\right)$
    3) $\overline{\mathcal{S}} \leftarrow \overline{\mathcal{S}} \cup \{s^*\}$;
    4) Diffusion of $\sqrt{\frac{p_{s^*}}{1+\sigma_{s^*}^2}}\,\boldsymbol{c}_{s^*}$ over the network;
  end
```

VI. Numerical Results

In this section, we illustrate some numerical simulations aimed at assessing the performance of the proposed strategy for distributed learning of signals defined over graphs. Let us consider a network composed of $N = 20$ nodes, deployed over a unitary area, and having a sparse connectivity. We generate a graph signal from (1) having a spectral content limited to the first five eigenvectors of the Laplacian matrix of the graph. The observation noise in (5) is chosen to be zero-mean, Gaussian, with variance chosen uniformly random between 0 and 0.1 for all i. As a first example, in Fig. 1, we report the transient behavior of the MSD obtained by proposed method, for different number of nodes belonging to the expected sampling set. The expected sampling set is chosen according to the distributed strategy proposed in Table 2, where the function $f(\cdot)$ is chosen to be the logarithm of the pseudo-determinant function, and the sampling probabilities are set equal to $p_i = 0.5$ for all $i \in \overline{\mathcal{S}}$. The step-sizes μ_i in (10) are chosen equal to 0.5 for all i; the combination weights $\{w_{ij}\}$ are selected using the Metropolis rule. The curves are averaged over 200 independent simulations, and the corresponding theoretical steady-state values in (20) are reported for the sake of comparison. As we can see from Fig. 1, the theoretical predictions match well the simulation results.

As a further example, in Fig. 2, we illustrate the steady-state MSD of the algorithm in (10) comparing the performance

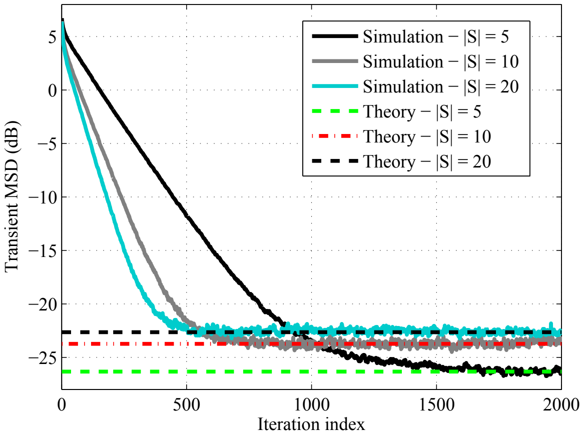

Fig. 1: Mean-Square performance: Transient MSD, and theoretical steady-state MSD in (20), for different values of $|\overline{\mathcal{S}}|$.

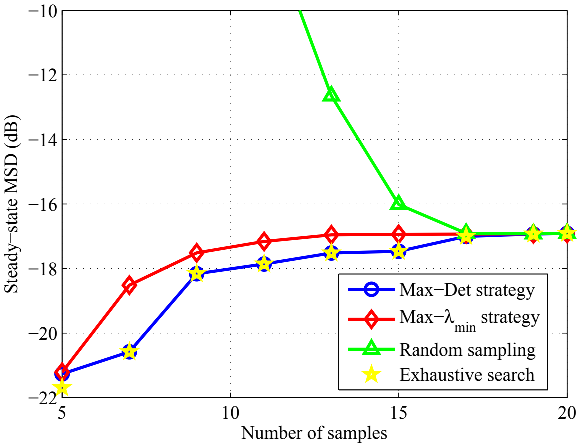

Fig. 2: Effect of sampling: Steady-state MSD versus number of samples, for different sampling strategies. $|\mathcal{F}| = 5$.

obtained by four different sampling strategies, namely: (a) the Max-Det strategy (obtained setting $f(\mathbf{X})$ as the logarithm of the pseudo-determinant of $\mathbf{X}$ in Table 2); (b) the Max-$\lambda_{\min}$ strategy (obtained setting $f(\mathbf{X}) = \lambda_{\min}(\mathbf{X})$ in Table 2); (c) the random sampling strategy, which simply picks at random $|\overline{\mathcal{S}}|$ nodes; and (d) the exhaustive search procedure aimed at minimizing the MSD in (20) over all the possible sampling combinations. In general, the latter strategy cannot be performed for large graphs and/or in a distributed fashion, and is reported only as a benchmark. Comparing the sampling strategies, we notice from Fig. 2 that the Max-Det strategy outperforms all the others, giving good performance also at low number of samples ($|\overline{\mathcal{S}}| = 5$ is the minimum number of samples that allows signal reconstruction). Interestingly, even if the proposed Max-Det strategy is a greedy approach, it shows performance that are comparable to the exhaustive search procedure, which represents the best possible performance achievable by a sampling strategy in terms of MSD.

VII. Conclusions

In this paper, we have proposed distributed strategies for adaptive learning of graph signals. The method enables distributed adaptive reconstruction and tracking from a limited number of observations taken over a subset of vertices. An interesting feature of our proposed method is that the sampling set is allowed to vary over time, and the convergence properties depend only on the expected set of sampling nodes. A detailed mean square analysis is also provided, illustrating the role of the sampling strategy on the reconstruction capability and mean-square performance of the proposed algorithm. Based on this analysis, some useful strategies for the distributed selection of the (expected) sampling set are also provided.

References

[1] 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 Proc. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[2] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs,” *IEEE Trans. on Sig. Proc.*, vol. 61, no. 7, pp. 1644–1656, 2013.

[3] S. K. Narang and A. Ortega, “Perfect reconstruction two-channel wavelet filter banks for graph structured data,” *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2786–2799, 2012.

[4] ——, “Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs,” *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4673–4685, 2013.

[5] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, “Discrete signal processing on graphs: Sampling theory,” *IEEE Trans. on Signal Proc.*, vol. 63, no. 24, pp. 6510–6523, Dec. 2015.

[6] M. Tsitsvero, S. Barbarossa, and P. Di Lorenzo, “Signals on graphs: Uncertainty principle and sampling,” *IEEE Transactions on Signal Processing*, vol. 64, no. 18, pp. 4845–4860, 2016.

[7] A. G. Marquez, S. Segarra, G. Leus, and A. Ribeiro, “Sampling of graph signals with successive local aggregations,” *IEEE Transaction on Signal Processing*, vol. 65, no. 7, pp. 1832–1843, Apr. 2016.

[8] X. Wang, P. Liu, and Y. Gu, “Local-set-based graph signal reconstruction,” *IEEE Trans. on Signal Proc.*, vol. 63, no. 9, pp. 2432–2444, 2015.

[9] 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*, Dec. 2016.

[10] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, “Reconstruction of graph signals through percolation from seeding nodes,” *IEEE Trans. Signal Processing*, vol. 64, no. 16, pp. 4363–4378, 2016.

[11] S. Chen, A. Sandryhaila, and J. Kovacevic, “Distributed algorithm for graph signal inpainting,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Brisbane, March 2015, pp. 3731–3735.

[12] D. Thanou and P. Frossard, “Distributed signal processing with graph spectral dictionaries,” in *Proceedings of Allerton Conference on Communication, Control, and Computing*, 2015.

[13] X. Wang, M. Wang, and Y. Gu, “A distributed tracking algorithm for reconstruction of graph signals,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 728–740, 2015.

[14] F. S. Cattivelli and A. H. Sayed, “Diffusion LMS strategies for distributed estimation,” *IEEE Trans. on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, March 2010.

[15] A. Bertrand and M. Moonen, “Seeing the bigger picture: How nodes can learn their place within a complex ad hoc network topology,” *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 71–82, 2013.

[16] P. Di Lorenzo and S. Barbarossa, “Distributed estimation and control of algebraic connectivity over random graphs,” *IEEE Transactions on Signal Processing*, vol. 62, no. 21, pp. 5615–5628, 2014.

[17] P. Di Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, “Distributed adaptive learning of graph signals,” *submitted to IEEE Transactions on Signal Processing, Online: https://arxiv.org/abs/1609.06100*, Sept. 2016.

[18] R. Olfati-Saber, A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.