Autoregressive Moving Average Graph Filtering

Isufi, Elvin; Loukas, Andreas; Simonetto, Andrea; Leus, Geert

DOI
10.1109/TSP.2016.2614793

Publication date
2017

Document Version
Final published version

Published in
IEEE Transactions on Signal Processing

Autoregressive Moving Average Graph Filtering

Elvin Isufi, *Student Member, IEEE*, Andreas Loukas, Andrea Simonetto, *Member, IEEE*, and Geert Leus, *Fellow, IEEE*

***Abstract*—One of the cornerstones of the field of signal processing on graphs are graph filters, direct analogs of classical filters, but intended for signals defined on graphs. This paper brings forth new insights on the distributed graph filtering problem. We design a family of autoregressive moving average (ARMA) recursions, which are able to *approximate* any desired graph frequency response, and give *exact* solutions for specific graph signal denoising and interpolation problems. The philosophy to design the ARMA coefficients independently from the underlying graph renders the ARMA graph filters suitable in static and, particularly, time-varying settings. The latter occur when the graph signal and/or graph topology are changing over time. We show that in case of a time-varying graph signal, our approach extends naturally to a two-dimensional filter, operating concurrently in the graph and regular time domain. We also derive the graph filter behavior, as well as sufficient conditions for filter stability when the graph and signal are time varying. The analytical and numerical results presented in this paper illustrate that ARMA graph filters are practically appealing for static and time-varying settings, as predicted by theoretical derivations.**

***Index Terms*—Distributed graph filtering, signal processing on graphs, infinite impulse response graph filters, autoregressive moving average graph filters, time-varying graph signals, time-varying graphs.**

I. Introduction

DUE to their ability to capture the complex relationships present in many high-dimensional datasets, graphs have emerged as a favorite tool for data analysis. Indeed, in recent years we have seen significant efforts to extend classical signal processing methods to the graph setting, where, instead of regular low-dimensional signals (e.g., a temporal or spatial signals), one is interested in graph signals, i.e., signals defined over the nodes of a graph [2]. The introduction of a Fourier-like transform for graph signals brought the tool to analyze these signals not only in the node domain, but also in the graph frequency domain [2]–[4]. One of the key tools of graph signal analysis are *graph filters*. In a direct analogy to classical filters, graph filters process a graph signal by selectively amplifying its graph Fourier coefficients. This renders them ideal for a wide range of tasks, ranging from graph signal smoothing and denoising [5], [6], classification [7]–[9] and interpolation [10], segmentation [11], wavelet construction [12], and dictionary learning [13]—among others.

Distributed implementations of filters on graphs emerged as a way of increasing the scalability of computation [6], [14]–[16]. In this way, a desired graph filtering operation is performed by only local information exchange between neighbors and there is no need for a node to have access to all the data. Nevertheless, being inspired by finite impulse response (FIR) graph filters, these methods are sensitive to time variations, such as time-varying signals and/or graphs. An alternative approach, namely distributed infinite impulse response (IIR) graph filtering, was recently proposed [17], [18]. Compared to FIR graph filters, IIR filters allow for the computation of a larger family of responses, and give exact rather than approximate solutions to specific denoising [5] and interpolation [10] problems. Yet the issue of time variations has so far been unresolved.

In a different context, we introduced IIR filter design (in fact, prior to [18]) using an autoregressive process called the potential kernel [11], [19]. These graph filters were shown to facilitate information processing tasks in sensor networks, such as smoothing and event region detection, but, due to their ad-hoc design, they only accomplished a limited subset of filtering objectives. In this paper, we build upon our prior work to develop more general autoregressive moving average (ARMA) graph filters of any order, using parallel or periodic concatenations of the potential kernel. The design philosophy of these graph filters allows for the approximation of any desired graph frequency response without knowing the structure of the underlying graph. In this way, we design the filter coefficients independently of the particular graph. This allows the ARMA filters to be *universally* applicable for any graph structure, and in particular when the graph varies over time, or when the graph structure is unknown to the designer.

Though ARMA graph filters belong to the class of IIR graph filters, they have a distinct design philosophy which bestows them the ability to filter graph signals not only in the graph frequency domain, but also in the regular temporal frequency domain (in case the graph signal is time-varying). Specifically, our design extends naturally to time-varying signals leading to two-dimensional ARMA filters: a filter in the graph domain as well as a filter in the time domain.

Our contributions are twofold:

(i) Distributed graph filters (Sections III and IV-A): We propose two types of autoregressive moving average (ARMA) recursions, namely the parallel and periodic implementation, which attain a rational graph frequency response. Both methods are implemented distributedly, attain fast convergence, and have message and memory requirements that are linear in the number of graph edges and the approximation order. Using a

Manuscript received January 7, 2016; revised May 9, 2016 and July 20, 2016; accepted September 9, 2016. Date of publication October 3, 2016; date of current version November 17, 2016. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Pierre Borgnat. This work was supported in part by STW under the D2S2 Project from the ASSYS program under Project 10561. This paper presents a generalization of [1]. *(Elvin Isufi and Andreas Loukas contributed equally to this work.)*

E. Isufi and G. Leus are with the Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology, Delft 2826 CD, The Netherlands (e-mail: e.isufi-1@tudelft.nl; g.j.t.leus@tudelft.nl).

A. Loukas is with the Department of Telecommunication Systems, TU Berlin, Berlin 10623, Germany (e-mail: a.loukas@tu-berlin.de).

A. Simonetto is with ICTEAM Institute, Université Catholique de Louvain, 1348 Louvain-la-Neuve, Belgium (e-mail: andrea.simonetto@uclouvain.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2016.2614793

variant of Shanks' method, we are able to design graph filters that approximate any desired graph frequency response. In addition, we give exact closed-form solutions for tasks such as Tikhonov and Wiener-based graph signal denoising and graph signal interpolation under smoothness assumptions.

(ii) Time-varying graphs and signals (Section V): We begin by providing a complete temporal characterization of ARMA graph filters w.r.t. time-varying graph signals. Our results show that the proposed recursions naturally extend to two-dimensional filters operating simultaneously in the graph-frequency domain and in the time-frequency domain. We also discuss the ARMA recursion behavior when both the graph topology and graph signal are time-varying. Specifically, we provide sufficient conditions for filter stability, and show that a decomposition basis exists (uniquely determined by the sequence of graph realizations), over which the filters achieve the same frequency response as in the static case.

Our results are validated by simulations in Section VI, and conclusions are drawn in Section VII.

Notation and terminology: We indicate a scalar valued variable by normal letters (i.e., a or A); a bold lowercase letter $\boldsymbol{a}$ will indicate a vector variable and a bold upper case letter $\boldsymbol{A}$ a matrix variable. With a_i and A_{ij} we will indicate the eneries of $\boldsymbol{a}$ and $\boldsymbol{A}$, respectively. For clarity, if needed we will refer to these entries also as $[\boldsymbol{a}]_i$ and $[\boldsymbol{A}]_{i,j}$ and to the i-th column of $\boldsymbol{A}$ as $[\boldsymbol{A}]_i$. We indicate by $|a|$ the absolute value of a and by $\|\boldsymbol{a}\|$ and $\|\boldsymbol{A}\|$ the 2-norm and the spectral norm of the vector $\boldsymbol{a}$ and matrix $\boldsymbol{A}$, respectively. To characterize convergence, we adopt the term *linear convergence*, which asserts that a recursion converges to its stationary value exponentially with time (i.e., linearly in a logarithmic scale) [20].

II. Preliminaries

Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of N nodes and M edges, where $\mathcal{V}$ indicates the set of nodes and $\mathcal{E}$ the set of edges. Let $\boldsymbol{x}$ be the graph signal defined on the graph nodes, whose i-th component $x_i \in \mathbb{R}$ represents the value of the signal at the i-th node, denoted as $u_i \in \mathcal{V}$.

Graph Fourier transform (GFT): The GFT transforms a graph signal $\boldsymbol{x}$ into the graph frequency domain $\hat{\boldsymbol{x}}$ by projecting it into the basis spanned by the eigenvectors of the graph Laplacian $\boldsymbol{L}$, typically defined as the discrete Laplacian $\boldsymbol{L}_{\mathrm{d}}$ or the normalized Laplacian $\boldsymbol{L}_{\mathrm{n}}$. Since the Laplacian matrix of an undirected graph is symmetric, its eigenvectors $\{\boldsymbol{\phi}_n\}_{n=1}^N$ form an orthonormal basis, and the forward and inverse GFTs of $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ are $\hat{\boldsymbol{x}} = \boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{x}$ and $\boldsymbol{x} = \boldsymbol{\Phi}\hat{\boldsymbol{x}}$, respectively, where the n-th column of $\boldsymbol{\Phi}$ is indicated as $\boldsymbol{\phi}_n$.The corresponding eigenvalues are denoted as $\{\lambda_n\}_{n=1}^N$ and will indicate the graph frequencies. For an extensive review of the properties of the GFT, we refer to [2], [3]. To avoid any restrictions on the generality of our approach, in the following, we present our results for a *general representation matrix* $\boldsymbol{L}$. We only require that $\boldsymbol{L}$ is *symmetric* and *local*: for all $i \neq j$, $L_{ij} = 0$ whenever u_i and u_j are not neighbors and $L_{ij} = L_{ji}$ otherwise. We derive our results for a class of graphs with general Laplacian matrices in some restricted set $\mathcal{L}$. We assume that for every $\boldsymbol{L} \in \mathcal{L}$ the minimum eigenvalue is bounded below by λ_{min} and the maximum eigenvalue is bounded above by λ_{max}. Hence, all considered graphs have a bounded spectral norm, i.e., $\|\boldsymbol{L}\| \leq \varrho = \max\{|\lambda_{max}|, |\lambda_{min}|\}$. For instance, when $\boldsymbol{L} = \boldsymbol{L}_{\mathrm{d}}$, we can take $\lambda_{min} = 0$ and $\lambda_{max} = l$, with l related to the maximum degree of any of the graphs. When $\boldsymbol{L} = \boldsymbol{L}_{\mathrm{n}}$, we can consider $\lambda_{min} = 0$ and $\lambda_{max} = 2$.

Graph filters: A *graph filter* $\boldsymbol{H}$ is an operator that acts upon a graph signal $\boldsymbol{x}$ by amplifying or attenuating its graph Fourier coefficients as

$$\boldsymbol{H}\boldsymbol{x} = \sum_{n=1}^{N} H(\lambda_n)\, \langle \boldsymbol{x}, \boldsymbol{\phi}_n \rangle \boldsymbol{\phi}_n, \tag{1}$$

where $\langle \cdot \rangle$ denotes the usual inner product. Let λ_{min} and λ_{max} be the minimum and maximum eigenvalues of $\boldsymbol{L}$ over *all* possible graphs. The graph frequency response $H : [\lambda_{min},\, \lambda_{max}] \to \mathbb{R}$ controls how much $\boldsymbol{H}$ amplifies the signal component of each graph frequency

$$H(\lambda_n) = \langle \boldsymbol{H}\boldsymbol{x}, \boldsymbol{\phi}_n \rangle / \langle \boldsymbol{x}, \boldsymbol{\phi}_n \rangle. \tag{2}$$

We are interested in how we can filter a signal with a graph filter $\boldsymbol{H}$ having a user-provided frequency response $H^*(\lambda)$. Note that this prescribed $H^*(\lambda)$ is a continuous function in the graph frequency λ and describes the desired response for *any* graph. This approach brings benefits in those cases when the underlying graph structure is not known to the designer, or in cases the graph changes in time. The corresponding filter coefficients are thus independent of the graph and *universally* applicable. Using universal filters, we can design a single set of coefficients that instantiate the same graph frequency response $H^*(\lambda)$ over different bases. To illustrate the universality property, consider the application of a universal graph filter to two different graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ of N_1 and N_2 nodes with graph frequency sets $\{\lambda_{1,n}\}_{n=1}^{N_1}$ $\{\lambda_{2,n}\}_{n=1}^{N_2}$, and eigenvectors $\{\boldsymbol{\phi}_{1,n}\}_{n=1}^{N_1}$, $\{\boldsymbol{\phi}_{1,n}\}_{n=1}^{N_2}$. The filter will attain the same response $H^*(\lambda)$ over both graphs, but, in each case, supported over a different set of graph frequencies: For $\mathcal{G}_1$, filtering results in $\sum_{n=1}^{N_1} H^*(\lambda_{1,n})\, \langle \boldsymbol{x}, \boldsymbol{\phi}_{1,n} \rangle \boldsymbol{\phi}_{1,n}$, whereas for $\mathcal{G}_2$ the filtering operator will be $\sum_{n=1}^{N_2} H^*(\lambda_{2,n})\, \langle \boldsymbol{x}, \boldsymbol{\phi}_{2,n} \rangle \boldsymbol{\phi}_{2,n}$. Thus, the universality lies in the correctness to implement $H^*(\lambda)$ on all graphs, which renders it applicable for time-varying graph topologies.

Distributed implementation: It is well known that we can *approximate* a universal graph filter $\boldsymbol{H}$ in a distributed way using a K-th order polynomial of $\boldsymbol{L}$, for instance using Chebychev polynomials [6]. Define FIR_K as the K-th order approximation given by

$$\boldsymbol{H} = h_0 \boldsymbol{I} + \sum_{k=1}^{K} h_k \boldsymbol{L}^k, \tag{3}$$

where the coefficients h_i can be both found by Chebyshev polynomial fitting [6] or in a least-squares sense, after a (fine) gridding of the frequency range, by minimizing

$$\int_{\lambda} \left| \sum_{k=0}^{K} h_k \lambda^k - H^*(\lambda) \right|^2 \mathrm{d}\lambda. \tag{4}$$

Observe that, in contrast to traditional graph filters, the order of the considered *universal* graph filters is not necessarily limited to N. By increasing K, we can approximate any filter with square integrable frequency response arbitrarily well. On the

other hand, a larger FIR order implies a longer convergence time in a distributed setting, since each node requires information from all its K-hop neighbors to attain the desired filter response.

To perform the filter *distributedly* in the network $\mathcal{G}$, we assume that each node $u_i \in \mathcal{V}$ is imbued with memory, computation, and communication capabilities and is in charge of calculating the local filter output $[\boldsymbol{H}\boldsymbol{x}]_i$. To do so, the node has to its disposal direct access to x_i, as well as indirect access to the memory of its neighbors. For simplicity of presentation, we pose an additional restriction to the computation model: we will assume that nodes operate in synchronous rounds, each one consisting of a message exchange phase and a local computation phase. In other words, in each round u_i may compute any (polynomial-time computable) function which has as input, variables from its local memory as well as those from the memory of its neighbors in $\mathcal{G}$. Since the algorithms examined in this paper are, in effect, dynamical systems, in the following we will adopt the term *iteration* as being synonymous to rounds. Furthermore, we assume that each iteration lasts exactly one time instant. In this context, the *convergence time* of an algorithm is measured in terms of the number of iterations the network needs to perform until the output closely reaches the *steady-state*, i.e., the asymptotic output of the dynamical system.

The computation of FIR_K is easily performed distributedly. Since $\boldsymbol{L}^K\boldsymbol{x} = \boldsymbol{L}\left(\boldsymbol{L}^{K-1}\boldsymbol{x}\right)$, each node u_i can compute the K-th term from the values of the $(K-1)$-th term in its neighborhood, in an iterative manner. The algorithm performing the FIR_K graph filter terminates after K iterations, and if a more efficient recursive implementation is used [6], in total, each node u_i exchanges $K \deg u_i$ values with its neighbors, meaning that, overall, the network exchanges $NK \deg u_i$ variables, amounting to a communication cost of $O(MK)$. Further, since for this computation each node keeps track of the values of its neighbors at every iteration, the network has a memory complexity of $O(M)$.

However, FIR_K filters exhibit poor performance when the graph signal or/and graph topology are time-varying, since the intermediate steps of the recursion cannot be computed exactly. This is for two reasons: i) First, the distributed averaging is paused after K iterations, and thus the filter output is *not a steady state* of the iteration $\boldsymbol{y}_t = \boldsymbol{L}\boldsymbol{y}_{t-1}$, which for $t = K$ gives $\boldsymbol{y}_K = \boldsymbol{L}^K\boldsymbol{x}$ as above. Accumulated errors in the computation alter the trajectory of the dynamical system, rendering intermediate states and the filter output erroneous. ii) Second, the input signal is only considered during the first iteration. To track a time-varying signal $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t$, a new FIR filter should be started at each time step t having $\boldsymbol{x}_t$ as input, significantly increasing the computation, message and memory complexities. To overcome these issues and provide a more solid foundation for graph signal processing, we study ARMA graph filters.

III. ARMA Graph Filters

This section contains our main algorithmic contributions. First, Sections III-A and III-B present distributed algorithms for implementing filters with a complex rational graph frequency response

$$H(\lambda) = \frac{p_n(\lambda)}{p_d(\lambda)} = \frac{\sum_{k=0}^{K} b_k \lambda^k}{1 + \sum_{k=1}^{K} a_k \lambda^k}, \tag{5}$$

where $p_n(\lambda)$ and $p_d(\lambda)$ are the complex numerator and denominator polynomials of order K. Note that this structure resembles the frequency response of temporal ARMA filters [21], in which case $\lambda = e^{j\omega}$, with ω being the temporal frequency. Though both polynomials are presented here to be of the same order, this is not a limitation: different orders for $p_n(\lambda)$ and $p_d(\lambda)$ are achieved trivially by setting specific constants a_k or b_k to zero.

A. ARMA_1 Graph Filters

Before describing the full fledged ARMA_K filters, it helps first to consider a 1-st order graph filter. Besides being simpler in its construction, an ARMA_1 lends itself as the basic building block for creating filters with a rational frequency response of any order (cf. Section III-B). We obtain ARMA_1 filters as an extension of the potential kernel [19]. Consider the following 1-st order recursion

$$\boldsymbol{y}_{t+1} = \psi\boldsymbol{L}\boldsymbol{y}_t + \varphi\boldsymbol{x} \tag{6a}$$

$$\boldsymbol{z}_{t+1} = \boldsymbol{y}_{t+1} + c\boldsymbol{x}, \tag{6b}$$

for arbitrary $\boldsymbol{y}_0$ and where the coefficients ψ, φ and c are complex numbers (to be specified later on). For this recursion, we can prove our first result.

Theorem 1: The frequency response of the ARMA_1 is

$$H(\lambda) = c + \frac{r}{\lambda - p}, \quad \textit{subject to} \quad |p| > \varrho \tag{7}$$

with residue r and pole p given by $r = -\varphi/\psi$ and $p = 1/\psi$, respectively, and with ϱ being the spectral radius bound of $\boldsymbol{L}$. Recursion (6) converges to it linearly, irrespective of the initial condition $\boldsymbol{y}_0$ and graph Laplacian $\boldsymbol{L}$.

Proof: We can write the recursion (6a) at time t in the expanded form as

$$\boldsymbol{y}_t = (\psi\boldsymbol{L})^t\boldsymbol{y}_0 + \varphi\sum_{\tau=0}^{t-1}(\psi\boldsymbol{L})^\tau\boldsymbol{x}. \tag{8}$$

When $||\psi\varrho < 1$ and as $t \to \infty$ this recursion approaches the steady state

$$\boldsymbol{y} = \lim_{t\to\infty}\boldsymbol{y}_t = \varphi\sum_{\tau=0}^{\infty}(\psi\boldsymbol{L})^\tau\boldsymbol{x} = \varphi\left(\boldsymbol{I} - \psi\boldsymbol{L}\right)^{-1}\boldsymbol{x}, \tag{9}$$

irrespective of $\boldsymbol{y}_0$. Based on Sylvester's matrix theorem, the matrix $\boldsymbol{I} - \psi\boldsymbol{L}$ has the same eigenvectors as $\boldsymbol{L}$ and its eigenvalues are equal to $1 - \psi\lambda_n$. It is also well known that invertible matrices have the same eigenvectors as their inverse and eigenvalues that are the inverse of the eigenvalues of their inverse. Thus,

$$\boldsymbol{z} = \lim_{t\to\infty}\boldsymbol{z}_t = \boldsymbol{y} + c\boldsymbol{x} = \sum_{n=1}^{N}\left(c + \frac{\varphi}{1-\psi\lambda_n}\right)\hat{x}_n\boldsymbol{\phi}_n, \tag{10}$$

and the desired frequency response (7) follows by simple algebra. We arrive at (10) by considering a specific realization of $\boldsymbol{L}$, thus the set of eigenvalues $\lambda_n \in [\lambda_{min}, \lambda_{max}]$ is discrete. However, the same result is achieved for every other graph realization matrix $\boldsymbol{L}$ with a potentially different set of eigenvalues, still in $[\lambda_{min}, \lambda_{max}]$. Thus, we can write (7) for all $\lambda \in [\lambda_{min}, \lambda_{max}]$. ■

The stability constraint in (7) can be also understood from a dynamical system perspective. Comparing recursion (6) to a discrete-time state-space equation, it becomes apparent that, when the condition $|\psi\varrho| < 1$ holds, recursion (6) achieves frequency response (7). It is useful to observe that, since $|p| > \varrho$, an increment of the stability region can be attained, if we work with a shifted version of the Laplacian $\boldsymbol{L}$ and thereby decrease the spectral radius bound ϱ. For instance, the following shifted Laplacians can be considered: $\boldsymbol{L} = \boldsymbol{L}_{\text{d}} - l/2\boldsymbol{I}$ with $\lambda_{min} = -l/2$ and $\lambda_{max} = l/2$ or $\boldsymbol{L} = \boldsymbol{L}_{\text{n}} - \boldsymbol{I}$ with $\lambda_{min} = -1$ and $\lambda_{max} = 1$. Due to its benefits, we will use the shifted versions of the Laplacians in the filter design phase and numerical results.[1]

Recursion (6) leads to a simple distributed implementation of a graph filter with 1-st order rational frequency response: at each iteration t, each node u_i updates its value $y_{t,i}$ based on its local signal x_i and a weighted combination of the values $y_{t-1,j}$ of its neighbors u_j. Since each node must exchange its value with each of its neighbors, the message/memory complexity at each iteration is of order $O(M)$, which leads to an efficient implementation.

Remark 1: Note that there is an *equivalence* between the ARMA_1 filter and FIR filters in approximating rational frequency responses. Indeed, the result obtained in (9) from the ARMA in $t \to \infty$ can also be achieved with an FIR filter of order $K = \infty$. Further, from (8) we can see that in finite time, i.e., $t = T$ and $\boldsymbol{y}_0 = 0$ the ARMA_1 output is equivalent to an FIR_{T-1} filter with coefficients $[\varphi, \varphi\psi, \ldots, \varphi\psi^{T-1}]$.

This suggests that: (i) the same output of an FIR filter can be obtained form (6) and (ii) the ARMA_1 graph filter can be used to design the FIR coefficients to approximate frequency responses of the form (7). As we will see later on, due to their implementation form (6), the ARMA filters are more robust than FIRs in a time-varying scenario (time-varying graph and/or time-varying graph signal).

B. ARMA_K Graph Filters

Next, we use ARMA_1 as a building block to derive distributed graph filters with a more complex frequency response. We present two constructions: The first uses a *parallel* bank of K ARMA_1 filters, attaining linear convergence with a communication and memory cost per iteration of $O(KM)$. The second uses *periodic* coefficients in order to reduce the communication costs to $O(M)$, while preserving the linear convergence as the parallel ARMA_K filters.

Parallel ARMA_K filters: A larger variety of filter responses can be obtained by adding the outputs of a parallel ARMA_1 filter bank. Let's denote with superscript (k) the terms that correspond to the k-th ARMA_1 filter for $k = 1, 2, \ldots, K$. With this notation in place, the output $\boldsymbol{z}_t$ of the ARMA_K filter at time instant t is

$$\boldsymbol{y}_{t+1}^{(k)} = \psi^{(k)} \boldsymbol{L}\boldsymbol{y}_t^{(k)} + \varphi^{(k)}\boldsymbol{x} \tag{11a}$$

$$\boldsymbol{z}_{t+1} = \sum_{k=1}^{K} \boldsymbol{y}_{t+1}^{(k)} + c\boldsymbol{x}, \tag{11b}$$

where $\boldsymbol{y}_0^{(k)}$ is arbitrary.

[1]Note that from Sylvester's matrix theorem, the shifted version of the Laplacians share the same eigenvectors as the original ones.

Theorem 2: The frequency response of a parallel ARMA_K is

$$H(\lambda) = c + \sum_{k=1}^{K} \frac{r_k}{\lambda - p_k} \quad subject\ to \quad |p_k| > \varrho, \tag{12}$$

with the residues $r_k = -\varphi^{(k)}/\psi^{(k)}$ and poles $p_k = 1/\psi^{(k)}$, and ϱ the spectral radius of $\boldsymbol{L}$. Recursion (11) converges to it linearly, irrespective of the initial conditions $\boldsymbol{y}_0^{(k)}$ and graph Laplacian $\boldsymbol{L}$.

Proof: Follows straightforwardly from Theorem 1. ■

The frequency response of a parallel ARMA_K is therefore a rational function with numerator and denominator polynomials of order K (presented here in a partial fraction form). In addition, since we are simply running K ARMA_1 filters in parallel, the communication and memory complexities are K times that of the ARMA_1 graph filter. Note also that the same considerations of Remark 1 can be extended to the parallel ARMA_K filter.

Periodic ARMA_K filters: We can decrease the memory requirements of the parallel implementation by letting the filter coefficients periodically vary in time. Our periodic filters take the following form

$$\boldsymbol{y}_{t+1} = (\theta_t \mathbf{I} + \psi_t \boldsymbol{L})\boldsymbol{y}_t + \varphi_t \boldsymbol{x} \tag{13a}$$

$$\boldsymbol{z}_{t+1} = \boldsymbol{y}_{t+1} + c\boldsymbol{x}, \tag{13b}$$

where $\boldsymbol{y}_0$ is the arbitrary, the output $\boldsymbol{z}_{t+1}$ is valid every K iterations, and coefficients $\theta_t, \psi_t, \varphi_t$ are periodic with period K: $\theta_t = \theta_{t-iK}, \psi_t = \psi_{t-iK}, \varphi_t = \varphi_{t-iK}$, with i an integer in $[0, t/K]$.

Theorem 3: The frequency response of a periodic ARMA_K filter is

$$H(\lambda) = c + \frac{\sum_{k=0}^{K-1} \left(\prod_{\tau=k+1}^{K-1} (\theta_\tau + \psi_\tau \lambda) \right) \varphi_k}{1 - \prod_{k=0}^{K-1} (\theta_k + \psi_k \lambda)}, \tag{14}$$

subject to the stability constraint

$$\left| \prod_{k=0}^{K-1} (\theta_k + \psi_k \varrho) \right| < 1 \tag{15}$$

with ϱ being the spectral radius bound of $\boldsymbol{L}$. Recursion (13) converges to it linearly, irrespective of the initial condition $\boldsymbol{y}_0$ and graph Laplacian $\boldsymbol{L}$.

(The proof is deferred to the appendix.)

By some algebraic manipulation, we can see that the frequency response of periodic ARMA_K filters is also a rational function of order K. We can also observe that the stability criterion of parallel ARMA_K is more involved than that of the parallel implementation. As now we are dealing with K ARMA_1 graph filters interleaved in time, to guarantee their joint stability one does not necessarily have to examine them independently (requiring for instance that, for each k, $|\theta_k + \psi_k \varrho| < 1$). Instead, it is sufficient that the product $|\prod_{k=0}^{K-1} (\theta_k + \psi_k \varrho)|$ is smaller than one. To illustrate this, notice that if $\theta_k = 0$, the periodic ARMA_K can be stable even if some of the ARMA_1 graph filters it is composed of are unstable.

When computing a periodic ARMA_K distributedly, in each iteration each node u_i stores and exchanges $\deg(u_i)$ values with its neighbors, yielding a memory complexity of $O(M)$, rather than the $O(KM)$ of the parallel one (after each iteration, the values are overwritten). On the other hand, since the output

of the periodic ARMA$_K$ is only valid after K iterations, the communication complexity is again $O(KM)$. The low memory requirements of the periodic ARMA$_K$ render it suitable for resource constrained devices.

IV. ARMA Filter Design

In this section we focus on selecting the coefficients of our filters. We begin by showing how to approximate any given frequency response with an ARMA filter, using a variant of Shanks' method [22]. This approach gives us stable filters, ensuring the same selectivity as the universal FIR graph filters. Section IV-B then provides explicit (and exact) filter constructions for two graph signal processing problems which were up-to-now only approximated: *Tiknohov and Wiener-based signal denoising* and *interpolation under smoothness assumptions* [6], [23] and [10].

A. The Filter Design Problem

Given a graph frequency response $H^*:[\lambda_{min},\,\lambda_{max}]\rightarrow\mathbb{R}$ and filter order K, our objective is to find the complex polynomials $p_n(\lambda)$ and $p_d(\lambda)$ of order K that minimize

$$\int_\lambda \left|\frac{p_n(\lambda)}{p_d(\lambda)}-H^*(\lambda)\right|^2 \mathrm{d}\lambda = \int_\lambda \left|\frac{\sum_{k=0}^K b_k\lambda^k}{1+\sum_{k=1}^K a_k\lambda^k}-H^*(\lambda)\right|^2 \mathrm{d}\lambda \tag{16}$$

while ensuring that the chosen coefficients result in a stable system (see constraints in Theorems 2 and 3). From $p_n(\lambda)/p_d(\lambda)$ one computes the filter coefficients ($\psi^{(k)},\varphi^{(k)},c$ or θ_t,ψ_t,c) by algebraic manipulation.

Remark 2: Even if we constrain ourselves to pass-band filters and we consider only the set of $\boldsymbol{L}$ for which $\varrho=1$, it is impossible to design our coefficients based on classical design methods developed for IIR filters (e.g., Butterworth, Chebyshev). The same issue is present also using a rational fitting approach, e.g., Padé and Prony's method. This is due to the fact that, now, the filters are rational in the variable λ and the notion of frequency does not stem from $j\omega$ nor from $e^{j\omega}$. This differentiates the problem from the design of continuous and discrete time filters. Further, the stability constraint of ARMA$_K$ is different from classical filter design, where the poles of the transfer function must lie within (not outside) the unit circle.

To illustrate this remark, consider the Butterworth-like graph frequency response $h(\lambda)=\left(1+(\lambda/\lambda_c)^K\right)^{-1}$, where λ_c is the desired cut-off frequency. For $K=2$, one finds that it has two complex conjugate poles at $\pm\jmath\lambda_c$. Thus, the behavior of these filters depends on the cut-off frequency and stability is not always guaranteed. For this particular case, and for a parallel implementation, whenever $\lambda_c>\varrho$ the filters are not stable.

Design method: Similar to Shanks' method, we approximate the filter coefficients as follows:

Denominator: Determine a_k for $k=1,\ldots,K$ by finding a $\hat{K}>K$ order polynomial approximation $\hat{H}(\lambda)=\sum_{k=0}^{\hat{K}} g_k\lambda^k$ of $H^*(\lambda)$ using polynomial regression, and solving the coefficient-wise system of equations $p_d(\lambda)\hat{H}(\lambda)=p_n(\lambda)$.

Numerator: Determine b_k for $k=1,\ldots,K$ by solving the least-squares problem of minimizing $\int_\lambda |p_n(\lambda)/p_d(\lambda)-H^*(\lambda)|^2 d\mu$, w.r.t. $p_n(\lambda)$.

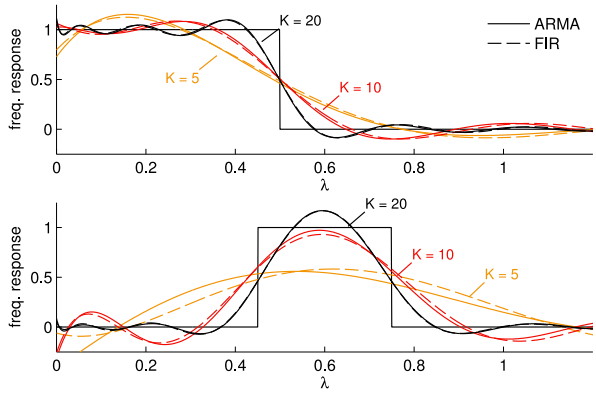

Fig. 1. The frequency response of ARMA$_K$ filters designed by Shanks' method and the FIR responses of corresponding order. Here, $H^*(\lambda)$ is a step function (top) and a window function (bottom).

Once the numerator (b_k) and denominator (a_k) coefficients of the target rational response are found:

(i) Parallel design: Perform the partial fraction expansion to find the residuals (r_k) and poles (p_k). Then, the filter coefficients $\psi^{(k)}$ and $\varphi^{(k)}$ can be found by exploiting their relation with r_k and p_k in Theorem 2.

(ii) Periodic design: Identify ψ_k by computing the roots of the (source) denominator polynomial $1-\prod_{k=0}^{K-1}(\theta_k+\psi_k\lambda)$ in (14) and equating them to the roots of the (target) denominator $1+\sum_{k=1}^K a_k\lambda^k$. It is suggested to set $\theta_1=0$ and $\theta_k=1$ for $k>0$, which has the effect of putting the two polynomials in similar form. Once coefficients ψ_k (and θ_k) have been set, we obtain φ_k by equating the numerator target and source polynomials.

The method is also suitable for numerator and denominator polynomials of different orders. We advocate however the use of equal orders, because it yields the highest approximation accuracy for a given communication/memory complexity.

The most crucial step is the approximation of the denominator coefficients. By fitting $p_d(\lambda)$ to $\hat{g}(\lambda)$ instead of $g(\lambda)$, we are able to compute coefficients a_k independently of b_k. Increasing $\hat{K}\gg K$ often leads to a (slight) increase in accuracy, but at the price of slower convergence and higher sensitivity to numerical errors (such as those caused by packet loss). Especially for sharp functions, such as the ones shown in Fig. 1, a high order polynomial approximation results in very large coefficients, which affect the numerically stability of the filters and push the poles closer to the unit circle. For this reason, in the remainder of this paper we set $\hat{K}=K+1$.

Though the proposed design method does not come with theoretical stability guarantees, it has been found to consistently produce stable filters.[2] Fig. 1 illustrates in solid lines the frequency responses of three ARMA$_K$ filters ($K=5,10,20$), designed to approximate a step function (top) and a window function (bottom). For reproducibility, Table I, which is featured in the Appendix, summarizes the filter coefficients of the parallel implementation for different K.

[2]This has been observed while working with shifted Laplacians, and especially with the shifted normalized Laplacian $\boldsymbol{L}=\boldsymbol{L}_{\mathrm{n}}-\boldsymbol{I}$.

B. Exact and Universal Graph Filter Constructions

We proceed to present exact (and in certain cases explicit) graph filter constructions for particular graph signal denoising and interpolation problems. In contrast to previous work, the proposed filters are *universal*, that is they are designed without knowledge of the graph structure. Indeed, the filter coefficients are found independently from the eigenvalues of the graph Laplacian. This makes the ARMA filters suitable for any graph, and ideal for cases when the graph structure is unknown or when the $O(N^3)$ complexity of the eigenvalue decomposition becomes prohibitive.

Tikhonov-based denoising: Given a noisy signal $\boldsymbol{t} = \boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{x}$ is the true signal and $\boldsymbol{n}$ is noise, the objective is to recover $\boldsymbol{x}$ [2], [6], [23]. When $\boldsymbol{x}$ is smooth w.r.t. the graph, denoising can be formulated as the regularized problem

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathbb{C}^N}{\arg\min} \|\boldsymbol{x} - \boldsymbol{t}\|_2^2 + w\, \boldsymbol{x}^\top \boldsymbol{L}^K \boldsymbol{x}, \tag{17}$$

where the first term asks for a denoised signal that is close to $\boldsymbol{t}$, and the second uses the quadratic form of $\boldsymbol{L}^K$ to penalize signals that are not smooth. In (17), admitted choices of $\boldsymbol{L}$ are limited to the discrete Laplacian $\boldsymbol{L}_{\text{d}}$ or the normalized Laplacian $\boldsymbol{L}_{\text{n}}$ (without shifting). The positive regularization weight w allows us to trade-off between the two terms of the objective function. Being a convex problem, the global minimum $\tilde{\boldsymbol{x}}$ is found by setting the gradient to zero, resulting in

$$\tilde{\boldsymbol{x}} = \left(\boldsymbol{I} + w\boldsymbol{L}^K\right)^{-1} \boldsymbol{t} = \sum_{n=1}^{N} \frac{1}{1 + w\lambda_n^K} \langle \boldsymbol{t}, \boldsymbol{\phi}_n \rangle \boldsymbol{\phi}_n. \tag{18}$$

It follows that (18) can be approximated with an ARMA$_K$ with frequency response

$$H(\lambda) = \frac{1}{1 + w\lambda^K} = \frac{1}{\prod_{k=1}^{K} (\lambda - p_k)} \tag{19}$$

with $p_k = -e^{\jmath\gamma_k} / \sqrt[K]{w}$ and $\gamma_k = (2k+1)\pi/K$. From the stability condition of the parallel ARMA$_K$, we have stable filters as long as that $|p_k| > \varrho$, which for the particular expression of p_k becomes $\sqrt[K]{w}\varrho < 1$.

The solution of (18) can also computed by an ARMA$_K$ implemented on the shifted Laplacians, with a notable improvement on the stability of the filters. For $\boldsymbol{L} = \boldsymbol{L}_{\text{d}} - l/2\boldsymbol{I}$ we can reformulate (19) as

$$H(\lambda) = \frac{1}{1 + w(\lambda + \frac{l}{2})^K} = \frac{1}{\prod_{k=1}^{K} (\lambda - p_k)} \tag{20}$$

where now $p_k = -l/2 + e^{\jmath\gamma_k} / \sqrt[K]{w}$ for $\gamma_k = (2k+1)\pi/K$. Again, from the stability of ARMA$_K$ $|p_k| > \varrho$, or equivalently $|p_k|^2 > \varrho^2$, we now obtain stable filters as long as

$$\left(-\frac{l}{2} + \frac{\cos(\gamma_k)}{\sqrt[K]{w}}\right)^2 + \frac{\sin^2(\gamma_k)}{\sqrt[K]{w^2}} > \varrho^2, \tag{21}$$

or equivalently

$$\left(\frac{l^2}{4} - \varrho^2\right) \sqrt[K]{w^2} - l\cos(\gamma_k) \sqrt[K]{w} + 1 > 0, \tag{22}$$

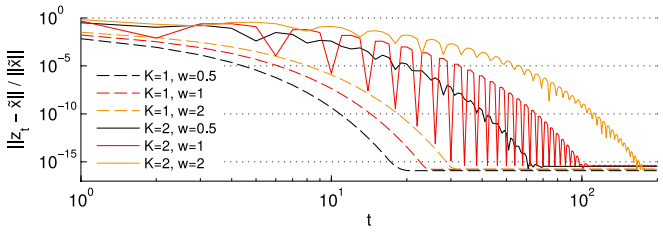

Fig. 2. Convergence of a denoising parallel ARMA$_K$ filter for $K = 1, 2$ and $w = 0.5, 1, 2$. The filtering error is $\|\boldsymbol{z}_t - \tilde{\boldsymbol{x}}\| / \|\tilde{\boldsymbol{x}}\|$, where for each parameter pair (K, w), $\tilde{\boldsymbol{x}}$ is the solution of the denoising problem, and $\boldsymbol{z}_t$ is the filter output after t rounds. The residual error is a consequence of the computer's bounded numerical precision.

are satisfied. For the shifted normalized Laplacian, $\varrho = 1$ and $l = 2$, the stability condition simplifies to

$$2\cos(\gamma_k) \sqrt[K]{w} < 1, \tag{23}$$

which is always met for the standard choices of $K = 1$ (quadratic regularization) and $K = 2$ (total variation).[3] For these values of K, and for different values of the regularized weight w, we show in Fig. 2 the normalized error between the output of the ARMA$_K$ recursion and the solution of the optimization problem (17), as a function of time.

For both (19) and (20), the denominator coefficients $\psi^{(k)}$ of the corresponding parallel ARMA$_K$ filter can be found as $\psi^{(k)} = 1/p_k$. Meanwhile, the numerator coefficients $\varphi^{(k)}$ are found in two steps: (i) express (19), (20) in the partial form as in (12) to find the residuals r_k and (ii) take $\varphi^{(k)} = -r_k\psi^{(k)}$.

Wiener-based denoising: When the statistical properties of the graph signal and noise are available, it is common to opt for a Wiener filter, i.e., the linear filter that minimizes the mean-squared error (MSE)

$$\tilde{\boldsymbol{H}} = \underset{\boldsymbol{H}\in\mathbb{R}^{N\times N}}{\arg\min}\, \mathbb{E}\left[\|\boldsymbol{H}\boldsymbol{t} - \boldsymbol{x}\|_2^2\right] \text{ and } \tilde{\boldsymbol{x}} = \tilde{\boldsymbol{H}}\boldsymbol{t}, \tag{24}$$

where as before $\boldsymbol{t} = \boldsymbol{x} + \boldsymbol{n}$ is the graph signal which has been corrupted with additive noise. It is well known that, when $\boldsymbol{x}$ and $\boldsymbol{n}$ are zero-mean with covariance $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{n}}$, respectively, the solution of (24) is

$$\tilde{\boldsymbol{x}} = \boldsymbol{\Sigma}_{\boldsymbol{x}}(\boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{n}})^{-1}\boldsymbol{t}. \tag{25}$$

given that matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{n}}$ is non-singular. The above linear system can be solved by a graph filter when matrices $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{n}}$ share the eigenvectors of the Laplacian matrix $\boldsymbol{L}$. Denote by $\sigma_{\boldsymbol{x}}(\lambda_n) = \boldsymbol{\phi}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{x}} \boldsymbol{\phi}_n$ the eigenvalue of matrix the $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ which corresponds to the n-th eigenvector of $\boldsymbol{L}$, and correspondingly $\sigma_{\boldsymbol{n}}(\lambda_n) = \boldsymbol{\phi}_n^\top \boldsymbol{\Sigma}_{\boldsymbol{n}} \boldsymbol{\phi}_n$. We then have that

$$\tilde{\boldsymbol{x}} = \sum_{n=1}^{N} \frac{\sigma_{\boldsymbol{x}}(\lambda_n)}{\sigma_{\boldsymbol{x}}(\lambda_n) + \sigma_{\boldsymbol{n}}(\lambda_n)} \langle \boldsymbol{t}, \boldsymbol{\phi}_n \rangle \boldsymbol{\phi}_n. \tag{26}$$

It follows that, when $\sigma_{\boldsymbol{x}}(\lambda)$ and $\sigma_{\boldsymbol{n}}(\lambda)$ are rational functions (of λ) of order K, the Wiener filter corresponds to an ARMA$_K$

[3] Even though w is a free parameter, for K = 1, 2 the value $\cos(\gamma_k)$ in (23) will be either 0 or -1, due to the expression of γ_k.

graph filter. Notice that the corresponding filters are still universal, as the ARMA_K coefficients depend on the rational functions $\sigma_{\boldsymbol{x}}(\lambda)$ and $\sigma_{\boldsymbol{n}}(\lambda)$, but not on the specific eigenvalues of the graph Laplacian. In a different context, similar results have been also observed in semi-supervised learning [24].

Let us illustrate the above with an example. Suppose that $\boldsymbol{x}$ is normally distributed with covariance equal to the pseudoinverse of the Laplacian $\boldsymbol{L}^\dagger$. This is a popular and well understood model for smooth signals on graphs with strong connections to Gaussian Markov random fields [25]. In addition, let the noise be white with variance w. Substituting this into (26), we find

$$\tilde{\boldsymbol{x}} = \sum_{n=1}^{N} \frac{1}{1+w\lambda_n} \langle \boldsymbol{t}, \boldsymbol{\phi}_n \rangle \boldsymbol{\phi}_n, \tag{27}$$

which is identical to the Tikhonov-based denoising for $K=1$ and corresponds to an ARMA_1 with $\varphi = 2/(2+wl)$ and $\psi = -2w/(2+wl)$, which as previously shown has always stable implementation. Note that even though the stability is ensured for the considered case, it does not necessarily hold for every covariance matrix. Indeed, the stability of the filter must be examined in a problem-specific manner.

Graph signal interpolation: Suppose that only r out of the N values of a signal $\boldsymbol{x}$ are known, and let $\boldsymbol{t}$ be the $N \times 1$ vector which contains the known values and zeros otherwise. Under the assumption of $\boldsymbol{x}$ being smooth w.r.t. $\boldsymbol{L} = \boldsymbol{L}_{\text{d}}$ or $\boldsymbol{L} = \boldsymbol{L}_{\text{n}}$, we can estimate the unknowns by the regularized problem

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x} \in \mathbb{R}^N}{\arg\min} \left\| \boldsymbol{S}\left(\boldsymbol{x} - \boldsymbol{t}\right) \right\|_2^2 + w\, \boldsymbol{x}^\top \boldsymbol{L}^K \boldsymbol{x}, \tag{28}$$

where $\boldsymbol{S}$ is the diagonal matrix with $S_{ii} = 1$ if x_i is known and $S_{ii} = 0$ otherwise. Such formulations have been used widely, both in the context of graph signal processing [10], [26] and earlier by the semi-supervised learning community [9], [27]. Similar to (17), this optimization problem is convex and its global minimum is found as

$$\tilde{\boldsymbol{x}} = \left(\boldsymbol{S} + w\boldsymbol{L}^K\right)^{-1} \boldsymbol{t}. \tag{29}$$

Most commonly, $K=1$, and $\tilde{\boldsymbol{x}}$ can be re-written as

$$\tilde{\boldsymbol{x}} = \left(\boldsymbol{I} - \hat{\boldsymbol{L}}\right)^{-1} \boldsymbol{t} = \sum_{n=1}^{N} \frac{1}{1+\hat{\lambda}_n} \langle \boldsymbol{t}, \hat{\boldsymbol{\phi}}_n \rangle \hat{\boldsymbol{\phi}}_n. \tag{30}$$

which is an ARMA_1 filter designed for the Laplacian matrix $\hat{\boldsymbol{L}} = \boldsymbol{S} - \boldsymbol{I} + w\boldsymbol{L}$ with $(\hat{\lambda}_n, \hat{\boldsymbol{\phi}}_n)$ the n-th eigenpair of $\hat{\boldsymbol{L}}$. For larger values of K, the interpolation cannot be computed distributedly using ARMA filters. That is because the corresponding basis matrix $\hat{\boldsymbol{L}} = \boldsymbol{S} + w\boldsymbol{L}^K$ cannot be appropriately factorized into a series of local matrices.

V. Time-Variations

At this point we have characterized the filtering and convergence properties of ARMA graph filters for static inputs. But do these properties hold when the graph and signal are a function of time? In the following, we characterize ARMA graph filters with respect to time-variations in the graph signal (cf. Section V-A), as well as in the graph topology (cf. Section V-B).

A. Joint Graph and Temporal Filters

To understand the impact of graph signal dynamics we broaden the analysis to a two-dimensional domain: the first dimension, as before, captures the graph (based on the graph Fourier transform), whereas the second dimension captures time (based on the Z-transform [21]). This technique allows us to provide a complete characterization of the ARMA filter subject to time-variations. First, we show that the ARMA filter output remains close to the correct time-varying solution (under sufficient conditions on the input), which implies that our algorithms exhibit a certain robustness to dynamics. Further, we realize that ARMA graph filters operate along both the graph and temporal dimensions. We find that a graph naturally dampens temporal frequencies in a manner that depends on its spectrum. Exploiting this finding, we extend the ARMA designs presented in Section III so as to also allow a measure of control over the temporal frequency response of the filters.

As previously, we start our exposition with the ARMA_1 recursion (6), but now the input graph signal $\boldsymbol{x}_t$ is time dependent (thus, indexed with the subscript t)

$$\boldsymbol{y}_{t+1} = \psi \boldsymbol{L}\boldsymbol{y}_t + \varphi \boldsymbol{x}_t \tag{31a}$$

$$\boldsymbol{z}_{t+1} = \boldsymbol{y}_{t+1} + c\boldsymbol{x}_t. \tag{31b}$$

The dimension of the above recursion can be reduced by restricting the input graph signal to lie in the subspace of an eigenvector $\boldsymbol{\phi}$ with associated eigenvalue μ, i.e., $\boldsymbol{x}_t = x_t \boldsymbol{\phi}$, where now x_t is a scalar and similarly, we take $\boldsymbol{y}_0 = y_0 \boldsymbol{\phi}$.[4] By orthogonality of the basis, the filter only alters the magnitude x_t relative to the eigenvector $\boldsymbol{\phi}$ and not the direction of $\boldsymbol{x}_t$. Therefore, (31) is equivalent to

$$y_{t+1} = \psi\lambda y_t + \varphi x_t \tag{32a}$$

$$z_{t+1} = y_{t+1} + cx_t, \tag{32b}$$

where $x_t, y_t, z_t \in \mathbb{R}$ are simply the magnitudes of the vectors $\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{z}_t \in \mathbb{C}^n$ lying in the eigenspace of $\boldsymbol{\phi}$, and we can write $\boldsymbol{y}_t = y_t \boldsymbol{\phi}$ and $\boldsymbol{z}_t = z_t \boldsymbol{\phi}$. Taking the Z-transform on both sides, we obtain the joint graph and temporal frequency transfer function

$$H(z, \lambda) = \frac{\varphi z^{-1}}{1 - \psi\lambda z^{-1}} + cz^{-1}. \tag{33}$$

It can be shown that the temporal-impulse response for each graph frequency λ is

$$\boldsymbol{h}_{t+1}(\lambda) = \left(\varphi(\psi\lambda)^t + c\delta[t]\right)\boldsymbol{\phi}, \tag{34}$$

with $\delta[t]$ the impulse function. From (34) we deduce that the region of convergence (ROC) of the filter is $\{|z| > |\psi\lambda|, \text{for all } \lambda\}$ and that the filter is causal.

The joint transfer function characterizes completely the behavior of ARMA_1 graph filters for an arbitrary yet time-invariant graph: when $z \to 1$, we return to the constant $\boldsymbol{x}$ result and stability condition of Theorem 1, while for all other z we obtain the standard frequency response as well as the graph frequency one. As one can see, recursion (31) is an ARMA_1 filter in the graph domain as well as in the time domain. Observe also that

[4] This is a standard way to derive the frequency response of the system.

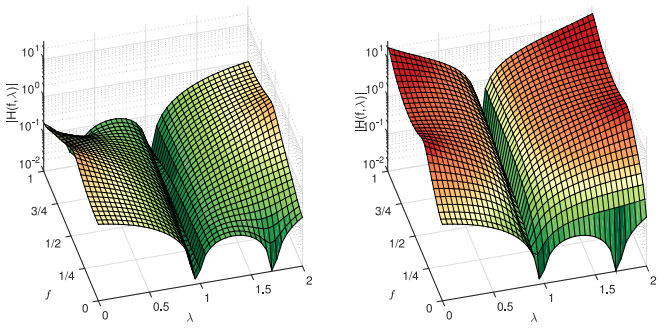

Fig. 3. The joint graph and temporal frequency response of a parallel and a periodic graph filter, both designed to approximate an ideal low pass (step) response with cut-off frequency $\lambda_c = 0.5$ and $K = 3$ w.r.t. the normalized graph Laplacian. The temporal frequencies f are normalized ($\times\pi$ rad/sample).

the poles of $H(z,\lambda)$ obey the fixed relationship $z = \lambda\psi$. This yields an interesting insight: the temporal frequency response of the filter differs along each graph frequency λ, meaning that temporal dynamics affecting signals lying in low graph frequency eigenspaces are dampened to a smaller extent.

As Theorems 4 and 5 below show, the results are readily generalized to higher order filters.

Theorem 4: The joint graph and temporal frequency transfer function of a parallel ARMA$_K$ is

$$H(z,\lambda) = \sum_{k=1}^{K} \frac{\varphi^{(k)} z^{-1}}{1 - \psi^{(k)} \lambda z^{-1}} + c z^{-1}, \tag{35}$$

subject to the stability conditions of Theorem 2.

Theorem 5: The joint graph and temporal frequency transfer function of a periodic ARMA$_K$ is

$$H(z,\lambda) = \frac{\sum_{k=0}^{K-1} \left(\prod_{\tau=k+1}^{K-1} \theta_\tau + \psi_\tau \lambda \right) \varphi_k z^{k-K}}{1 - \left(\prod_{\tau=0}^{K-1} \theta_\tau + \psi_\tau \lambda \right) z^{-K}} + c z^{-1}, \tag{36}$$

subject to the stability conditions of Theorem 3.

(The proofs are deferred to the appendix.)

As in the first order case, Theorems 4 and 5 describe completely the behavior of the parallel and periodic implementations. We can see that both filters are ARMA$_K$ filters in the graph and temporal domain. In particular, the parallel and periodic filters have up to K distinct poles abiding respectively to

$$z = \psi^{(k)} \lambda \quad \text{and} \quad z = \sqrt[K]{\prod_{\tau=0}^{K-1} \theta_\tau + \psi_\tau \lambda}. \tag{37}$$

To provide further insight, Fig. 3 plots the joint graph and temporal frequency response of a parallel and a periodic graph filter of third order, both designed (only in the graph domain) to approximate an ideal low pass response with cut-off frequency $\lambda_c = 0.5$. In the figure, the horizontal axis measures the graph frequency with smaller λ corresponding to lower variation terms. The temporal axis on the other hand measures the normalized temporal frequency f such that, for $f = 0$, one obtains the standard graph frequency response.

We make two observations: *First, both graph filters ensure almost the same frequency response as for the static case ($f = 0$) for low temporal variations $f \leq 1/8$.* This suggests that these filters are more appropriate for slow temporal variations. Whereas for graph signals lying in eigenspaces with λ close to $\lambda = 1$ all temporal frequencies are damped. This is a phenomenon that transcends the filter implementation (parallel or periodic) and the particular filter coefficients. It is attributed to the shifting of the Laplacian in the design phase and to the multiplicative relation of the response poles. *Second, the parallel concatenation of ARMA$_1$ filters results in a more stable implementation that is more fit to tolerate temporal dynamics than the periodic implementation.* As shown in Figure 3, for $\lambda = 0$ and $\lambda = 2$, temporal fluctuations with frequencies exceeding $1/8$ cause the periodic filter output to blow up by an order of magnitude, effectively rendering the periodic implementation unusable. The poor stability of the periodic implementation is also seen from (36), where the θ_τ terms tend to push the poles closer to the unit circle, and it is the price to pay for its small memory requirements. Due to its superior stability properties and convenient form (less coefficients and simpler design), we suggest the parallel implementation for dealing with time-varying graph signals.

B. Time-Varying Graphs and Signals

Time variations on the graph topology bring new challenges to the graph filtering problem. *First, they render approaches that rely on knowledge of the graph spectrum ineffective.* Approaches which ensure stability by designing the poles to lie outside the set of the Laplacian eigenvalues of a given graph, may lead to unstable filters in a different graph where some eigenvalues may over-shoot one of the poles. Due to their different design philosophy, the presented ARMA graph filters handle naturally the aforementioned issues. We can, for instance, think that the different graph realizations among time enjoy an upper bound on their maximum eigenvalue λ_{max}. In case this is not possible, or difficult to determine, we can always work with the normalized Laplacian and thus take $\lambda_{max} = 2$. In this way, by designing the filter coefficients in order to ensure stability w.r.t. λ_{max}, we automatically impose stability for all different graph realizations. Furthermore, by designing once the filter coefficients for a continuous range of frequencies, the ARMA recursions also preserve the desired frequency response for different graph realizations.

The second major challenge is characterizing the graph filter behavior. Time-varying affine systems are notoriously difficult to analyze when they possess no special structure [28]. To this end, we devise a new methodology for time-varying graph filter analysis. We show that a decomposition basis always exists, over which ARMA$_1$ graph filters (and as a consequence parallel ARMA$_K$ filters) have the same frequency response as in the static case. Furthermore, this decomposition basis depends only on the sequence of graph realizations.

In case of a time-varying graph topology, yet with a fixed number of nodes N, as well as a time-varying graph signal, the ARMA$_1$ recursion (6) can be written as

$$\boldsymbol{y}_{t+1} = \psi \boldsymbol{L}_t \boldsymbol{y}_t + \varphi \boldsymbol{x}_t \tag{38a}$$

$$\boldsymbol{z}_{t+1} = \boldsymbol{y}_{t+1} + c\boldsymbol{x}_t, \tag{38b}$$

where the time-varying graph is shown by indexing $\boldsymbol{L}_t$ with the subscript t. Expanding the recursion we find that, for any

sequence of graph realizations $\{G_0, G_1, \ldots, G_t\}$ with corresponding Laplacians $\{\boldsymbol{L}_0, \boldsymbol{L}_1, \ldots, \boldsymbol{L}_t\}$, the output signal is given by

$$\boldsymbol{z}_{t+1} = \psi^{t+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t,0)\boldsymbol{y}_0 + \varphi\sum_{\tau=0}^{t}\psi^{\tau}\boldsymbol{\Phi}_{\boldsymbol{L}}(t,t-\tau+1)\boldsymbol{x}_{t-\tau} + c\boldsymbol{x}_t, \tag{39}$$

where $\boldsymbol{\Phi}_{\boldsymbol{L}}(t,t') = \boldsymbol{L}_t\boldsymbol{L}_{t-1}\ldots\boldsymbol{L}_{t'}$ for $t \geq t'$, and $\boldsymbol{\Phi}_{\boldsymbol{L}}(t,t') = \boldsymbol{I}$ otherwise.

Since the output $\boldsymbol{z}_t$ depends on the entire sequence of graph realizations, the spectrum of any individual Laplacian is insufficient to derive the graph frequency of the filter. To extend the spectral analysis to the time-varying setting, we define a joint Laplacian matrix $\boldsymbol{L}_{tv}$ that encompasses all the individual shifted graph Laplacians. The intuition behind our approach is to think of a time-varying graph as one large graph $\mathcal{G}_{tv}$ that contains all nodes of the graphs $\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_t$, as well as directional edges connecting the nodes at different timesteps. We then interpret the spectrum of the associated Laplacian matrix $\boldsymbol{L}_{tv}$ as the basis for our time-varying graph Fourier transform. This idea is a generalization of the joint graph construction [29], used to define a Fourier transform for graph signals which change with time (though in the previous work the graph was considered time-invariant). Similar to [29], we will construct $\mathcal{G}_{tv}$ by replicating each node $u_i \in V$ once for each timestep t. Denote the t-th replication of the i-th node as $u_{t,i}$. For each t and i, $\mathcal{G}_{tv}$ will then contain directed edges between $u_{t-1,j}$ and $u_{t,i}$ with u_j being a neighbor of u_i in $\mathcal{G}_{t-1}$. Therefore, in contrast to previous work, here the edges between nodes u_i and u_j are a function of time. By its construction, $\mathcal{G}_{tv}$ captures not only the topology of the different graphs, but also the temporal relation between them: since the exchange of information between two neighbors incurs a delay of one unit of time, at each timestep t, a node has access to the values of its neighbors at $t-1$.

To proceed, define $\boldsymbol{P}$ to be the $(t+1)\times(t+1)$ cyclic shift matrix with ones below the diagonal and construct $\boldsymbol{L}_{tv}$ as the $N(t+1)\times N(t+1)$ permuted block-diagonal matrix

$$\boldsymbol{L}_{tv} = \text{blkdiag}[\boldsymbol{L}_0, \boldsymbol{L}_1, \ldots, \boldsymbol{L}_t](\boldsymbol{P}\otimes\boldsymbol{I}), \tag{40}$$

For consistency with the established theory on GFT, when $t=0$ and the graph is time-invariant, we define $\boldsymbol{P}=1$. Let $\boldsymbol{e}_\tau$ be the $(t+1)$-dimensional canonical unit vector with $(e_\tau)_i = 1$ if $i=\tau$ and $(e_\tau)_i = 0$, otherwise. Defining $\boldsymbol{s} = [\boldsymbol{x}_0^\top, \boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_t^\top]^\top$ as the vector of dimension $N(t+1)$ which encompasses all input graph signals, we can then write

$$\boldsymbol{\Phi}_{\boldsymbol{L}}(t,t-\tau+1)\boldsymbol{x}_{t-\tau} = (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\,\boldsymbol{L}_{tv}^{\tau}\,\boldsymbol{s}. \tag{41}$$

In those cases when the non-symmetric matrix $\boldsymbol{L}_{tv}$ enjoys an eigendecomposition, we have $\boldsymbol{L}_{tv} = \boldsymbol{U}\lambda\boldsymbol{U}^{-1}$ with $(\lambda_k, [\boldsymbol{U}]_k)$ the k-th eigenpair. Specifically, λ_k is the k-th diagonal element of λ and $[\boldsymbol{U}]_k$ is the k-th column of $\boldsymbol{U}$. The total number of eigenpairs of $\boldsymbol{L}_{tv}$ is $K = N\times(t+1)$. To ease the notation, we will denote as $[\boldsymbol{U}]_k^{-1}$ the respective k-th column of $\boldsymbol{U}^{-1}$.

Substituting (41) into the second term of (39) and rearranging the sums, we get

$$\begin{aligned}
&\varphi\sum_{\tau=0}^{t}\psi^{\tau}\boldsymbol{\Phi}_{\boldsymbol{L}}(t,t-\tau+1)\boldsymbol{x}_{t-\tau} = \varphi(\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{\tau=0}^{t}(\psi\boldsymbol{L}_{tv})^{\tau}\,\boldsymbol{s}\\
&= \varphi(\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{\tau=0}^{t}\sum_{k=1}^{K}(\psi\lambda_k)^{\tau}\,\langle\boldsymbol{s},[\boldsymbol{U}]_k^{-1}\rangle[\boldsymbol{U}]_k\\
&= (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{k=1}^{K}\varphi\,\frac{1-(\psi\lambda_k)^{t+1}}{1-\psi\lambda_k}\langle\boldsymbol{s},[\boldsymbol{U}]_k^{-1}\rangle[\boldsymbol{U}]_k.
\end{aligned}\tag{42}$$

Similarly,

$$\begin{aligned}
\psi^{t+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t,0)\boldsymbol{y}_0 &= (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\,(\psi\boldsymbol{L}_{tv})^{t+1}\,(\boldsymbol{e}_{t+1}\otimes\boldsymbol{y}_0)\\
&= (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{k=1}^{K}(\psi\lambda_k)^{t+1}\langle\boldsymbol{e}_{t+1}\otimes\boldsymbol{y}_0,[\boldsymbol{U}]_k^{-1}\rangle[\boldsymbol{U}]_k
\end{aligned}\tag{43}$$

as well as

$$c\,\boldsymbol{x}_t = (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{k=1}^{K}c\,\langle\boldsymbol{s},[\boldsymbol{U}]_k^{-1}\rangle[\boldsymbol{U}]_k. \tag{44}$$

Without loss of generality, when t is sufficiently large we can ignore terms of the form $(\psi\lambda_k)^{t+1}$ as long as $|\psi\lambda_k|<1$, which also indicates that the impact of the filter initialization $\boldsymbol{y}_0$ on the filter output vanishes with time.This condition is met when $\|\psi\boldsymbol{L}_{tv}\|<1$, which as a direct consequence of Gershgorin's circle theorem, this stability condition is met if, for every τ, the sum of the elements of each row of $\boldsymbol{L}_\tau$, in absolute value, is smaller than $|1/\psi|$ (which also implies that the eigenvalues of $\boldsymbol{L}_\tau$ are bounded by $|1/\psi|$). For the (shifted) normalized Laplacian this means that $|\psi|<2$ ($|\psi|<1$), matching exactly the stability conditions of the static case. Under this sufficient condition, the filter output approaches

$$\boldsymbol{z}_{t+1} \approx (\boldsymbol{e}_{t+1}^\top\otimes\boldsymbol{I})\sum_{k=1}^{K}\left(\frac{\varphi}{1-\psi\lambda_k}+c\right)\langle\boldsymbol{s},[\boldsymbol{U}]_k^{-1}\rangle[\boldsymbol{U}]_k. \tag{45}$$

Notice that the ARMA_1 retains the same graph frequency response as in the time-invariant case (7), now expressed in the basis of $\boldsymbol{L}_{tv}$. It is not difficult to show that the ARMA_1 graph filter converges asymptotically. Let us denote the distance between the filter output at two different time instants $t_1 > t_2$ as

$$\epsilon_{t_1,t_2} = \frac{\|\boldsymbol{z}_{t_1}-\boldsymbol{z}_{t_2}\|}{x_{max}}. \tag{46}$$

where $x_{max} = \max_{t=1,\ldots,t_1}\|\boldsymbol{x}_t\|$ constitutes an upper bound on the energy of the input. We can now claim

Theorem 6: Given the ARMA_1 recursion (39) and given that the graph Laplacians are uniformly bounded for every t $\|\boldsymbol{L}_t\|\leq\varrho$, the distance ϵ_{t_1,t_2} between the filter output at time instants t_1 and t_2 is upper-bounded as

$$\begin{aligned}
\epsilon_{t_1,t_2} \leq \|\boldsymbol{y}_0\|\,\frac{|\psi\varrho|^{t_1}+|\psi\varrho|^{t_2}}{x_{max}} + |\varphi|\frac{|\psi\varrho|^{t_2}-|\psi\varrho|^{t_1}}{1-|\psi\varrho|}\\
+|c|\,\frac{\|\boldsymbol{x}_{t_1-1}-\boldsymbol{x}_{t_2-1}\|}{x_{max}}.
\end{aligned}\tag{47}$$

(The proof is deferred to the appendix.)

For simplicity, set $c = 0$ and consider t_1 big enough such that the term $|\psi\varrho|^{t_1} \approx 0$. Then, directly from (47) we can find the value of t_2 such that the error between the two is smaller than a desired positive constant ε, i.e.,

$$t_2 \geq \log(\alpha/\varepsilon) \quad \Rightarrow \quad \epsilon_{t_1,t_2} \leq \varepsilon, \tag{48}$$

with $\alpha = \|\boldsymbol{y}_0\|/x_{max} + |\varphi|/(1-|\psi\varrho|)$.

The results of Theorem 6 can be extended to the general ARMA_K graph filter. For the parallel implementation, we can proceed in the same way as for the ARMA_1 by considering that the output signal is the sum of K ARMA_1 graph filters. Meanwhile, for the periodic implementation we can see that its form (53), after one cyclic period, is analogous to (38).

The main result of Theorem 6 stands in the fact that the ARMA output will not diverge as long as the graph Laplacians of each realization $\mathcal{G}_t$ has uniformly bounded spectral norm and from (48) the distance decreases exponentially. Further, for t big enough and if $\boldsymbol{L}_{tv}$ enjoys an eigendecompositon the result in (45) gives us insights where the ARMA output converges. Numerical results suggest that the obtained output is generally close to the designed frequency response of the ARMA filter.

VI. Numerical Results

To illustrate our results we simulate two different case-studies: one with a fixed graph and a time-varying graph signal, and one where both the graph and graph signal are time-varying. In the latter case, the ARMA performance is also compared to the state-of-the-art FIR filters designed in a universal manner [6]. With the first case-study, we aim to show how the proposed filters operate on graph signals that have spectral content in both graph and temporal frequency domains. Meanwhile, with the second the goal is to illustrate the ARMA performance when the underlying graph topology is not static anymore, but varies with time. For all our simulations, the ARMA filters, if not differently mentioned, are initialized to zero (i.e., $\boldsymbol{y}_0 = \boldsymbol{0}$ and $\boldsymbol{y}_0^{(k)} = \boldsymbol{0}$ for all k) and the filter design is performed in a universal setting.

A. Variations on the Graph Signal

In this subsection, we present simulation results for time-varying signals. We consider a 0.5-bandlimited graph signal $\boldsymbol{u}_t$ oscillating with a fixed temporal frequency $\pi/10$, meaning that

$$\langle \boldsymbol{u}_t, \boldsymbol{\phi}_n \rangle = \begin{cases} e^{\jmath\pi t/10} & \text{if } \lambda_n < 0.5 \\ 0 & \text{otherwise,} \end{cases} \tag{49}$$

where λ_n is the n-th eigenvalue of the normalized graph Laplacian and t it the time index. The signal is then corrupted with a second interfering signal $\boldsymbol{v}_t$, oscillating with a temporal frequency $9\pi/10$ with graph spectrum defined in the following in two different ways.. In addition, the signal at each node is corrupted with i.i.d. Gaussian noise $\boldsymbol{n}_t$, with zero mean and variance $\sigma^2 = 0.1$. We then attempt to recover $\boldsymbol{u}_t$ by filtering it with a parallel ARMA_5 graph filter, effectively canceling the interference $\boldsymbol{v}_t$ and attenuating the out of band noise. The ARMA filter is designed only in the graph frequency domain based on the GFT of $\boldsymbol{u}_t$, i.e., to approximate an ideal low-pass filter in the graph domain with cut-off frequency $\lambda_c = 0.5$. Regarding the temporal part, we exploit the property of the filter to preserve

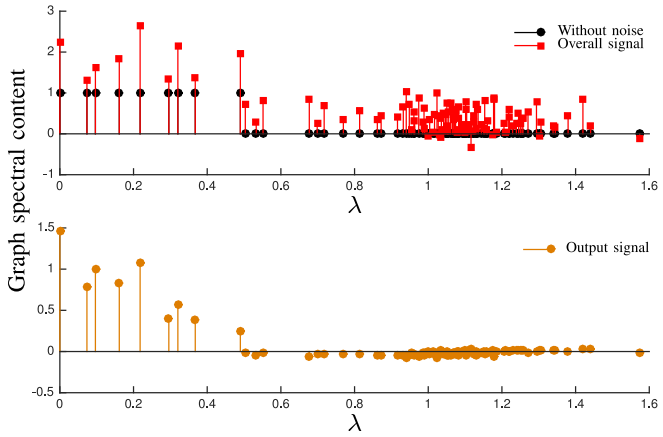

Fig. 4. Graph spectral content of the input signal as well as of the overall signal affected by interference and noise (a) (top), and of the filter output signal (b) (bottom). The output signal graph spectrum is shown for $t = 100$.

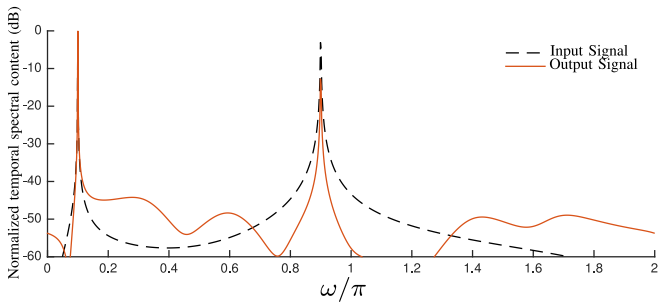

Fig. 5. Average time spectral content over all nodes of the input and output signal. The values are normalized with respect to the maximum.

the same graph frequency response as the static case for low temporal oscillations, while attenuating the contribution of high temporal frequencies. Our simulations were conducted using a random geometric graph G composed of 100 nodes placed randomly in a square area, with any two nodes being connected if they are closer than 15% of the maximum distance in the area, with an average degree of 11.8. Depending on whether the interference is correlated with the signal or not, we distinguish between two scenarios:

i) Correlated Signal Interference: In this scenario, the interference is self-induced, meaning that at a given instant t, $\boldsymbol{v}_t$ and $\boldsymbol{u}_t$ share the same graph spectrum, but oscillating at a higher temporal frequency (due for instance to electronics problems). To provide intuition, in Fig. 4.(a), we show the graph spectral content of $\boldsymbol{u}_0$ and $\boldsymbol{u}_0 + \boldsymbol{v}_0 + \boldsymbol{n}_0$. We can see that once corrupted by noise and interference, the graph signal presents significant spectral content across the graph spectrum, thus loosing its bandlimited nature. Meanwhile, Fig. 4.(b) depicts the real part of the graph spectral content of the filter output after 100 iterations (i.e., well after the initial state is forgotten). Even though the figure cannot capture the effect of dynamics (as it solely focuses on $t = 100$), it does show that all frequencies above $\lambda_c = 0.5$ have been attenuated and the interference contribution in the band is reduced. To illustrate the filtering of the temporal frequencies of the signal, in Fig. 5 we show the average spectrum over all nodes

of the input and output signal. To increase visibility, the values in the figure are normalized with respect to the maximum. We can see that, the content relative to the interfering frequency $9\pi/10$ of the output signal is attenuated around 13 dB with respect to the main temporal frequency content of $\pi/10$.

ii) Uncorrelated signal interference: Let us now consider a more involved scenario, in which the interfering graph signal satisfies

$$\langle \boldsymbol{v}_t, \boldsymbol{\phi}_n \rangle = e^{j9\pi t/10} e^{-\lambda_n}, \tag{50}$$

i.e., it is a signal having a heat kernel-like graph spectrum oscillating in time with a pulsation $\omega = 9\pi/10$. We will examine two types of errors: i) The first compares for each time t the ARMA_5 output GFT $\hat{\boldsymbol{z}}_t$ to that of the signal of interest

$$e_t^{(total)} = \frac{\|\hat{\boldsymbol{z}}_t - \hat{\boldsymbol{u}}_t\|}{\|\hat{\boldsymbol{u}}_t\|}. \tag{51}$$

Achieving a small error $e_t^{(total)}$ is a very challenging problem since an algorithm has to simultaneously overcome the addition of noise and the interference, while at the same time operating in a time-varying setting (see Fig. 6). ii) The second error focuses on interference and compares $\boldsymbol{z}_t$ to the output $\boldsymbol{z}_t^*$ of the same ARMA_5 operating on $\boldsymbol{u}_t + \boldsymbol{n}_t$ (but not $\boldsymbol{u}_t + \boldsymbol{v}_t + \boldsymbol{n}_t$)

$$\text{e}_t^{(interf)} = \frac{\|\hat{\boldsymbol{z}}_t - \hat{\boldsymbol{z}}_t^*\|}{\|\hat{\boldsymbol{z}}_t^*\|}, \tag{52}$$

where $\hat{\boldsymbol{z}}_t^*$ is the GFT of $\boldsymbol{z}_t^*$.

We can see from Fig. 6 that after a few iterations this error becomes relatively small, which means that the output spectrum of the ARMA recursion when the signal is affected by interference is similar to when the interference-less signal is used. This gives a first insight, that using the ARMA recursion we can manage multiple signals on a graph by simply making them orthogonal in the temporal frequency domain. By a specific design of the filter coefficients, one can distributively operate on the graph signal of interest and ignore the others. Such a result cannot be achieved with FIR filters for two reasons: (i) they suffer from handling time-varying input signals, and (ii) the FIR filters do not operate on the temporal frequency content of the graph signals, thus such a distinction between overlapping signals is difficult to achieve.

The above results illustrate the conclusions of Section V, and also quantify how much we can attenuate the signal at a specific graph/temporal frequency.

B. Variations on the Graph Topology

We examine the influence of graph variations for two filtering objectives. The first, which corresponds to denoising, can be computed *exactly* using ARMA. In the second objective, the graph filter is designed to *approximate* an ideal low-pass graph filter, i.e., a filter that eliminates all graph frequency components higher than some specific λ_c. In addition, we employ two different types of graph dynamics: *random edge failures*, where the edges of a graph disappear at each iteration with a fixed probability, as well as the standard model of *random waypoint* mobility [30]. The above setup allows us to test and compare universal ARMA and FIR graph filters (designed using the least-squares method) over a range of scenarios, each having different characteristics.

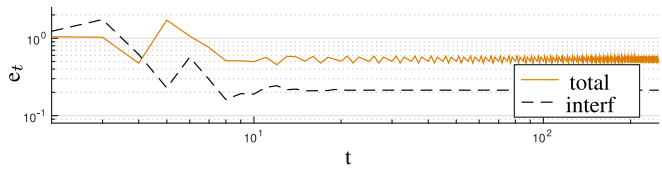

Fig. 6. Error of the ARMA recursion when the time-varying input signal is affected by uncorrelated interference.

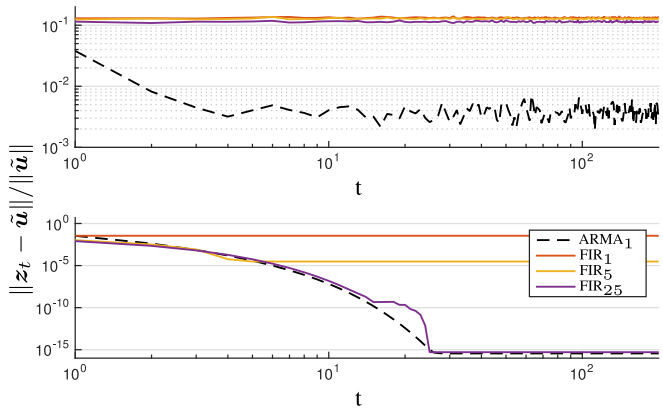

Fig. 7. Normalized error related to the solution of the denoising problem in a distributed way with graph filters. Results relative to random time-varying graph (top) and static graph (bottom). We compare the results of ARMA_1 with different FIR graph filters. The FIR_K output at time t is calculated as $\boldsymbol{y}_t = \sum_{k=0}^{K} h_k \Phi_{\boldsymbol{L}}(t, t-k+1)\boldsymbol{x}$ and is not arrested after K time instants.

Exact design (denoising): We simulate the denoising problem (as defined by (17), with $w = 0.5$ and $K = 1$) over the same graph topology of Section VI-A, where the probability that an edge goes down at each time instant is $p = 0.05$. The input signal $\boldsymbol{x} = \boldsymbol{u} + \boldsymbol{n}$ is given by a linear combination of a smooth signal $\boldsymbol{u}$ and noise $\boldsymbol{n}$. To ensure that the graph signal is smooth, we set its spectrum, w.r.t. the initial graph, as $\langle \boldsymbol{u}, \boldsymbol{\phi}_n \rangle = e^{-5\lambda_n}$. The noise $\boldsymbol{n}$ on the other hand is i.i.d. Gaussian distributed with zero mean and unit variance.

To compare the results, we calculate the normalized error between the graph filter output and the analytical solution of the optimization problem (17) solved w.r.t. the initial graph. In Fig. 7, we plot the normalized error of solving the denoising problem via distributed graph filtering. We consider an ARMA_1 graph filter (designed according to Section IV-B with $\boldsymbol{y}_0 = \boldsymbol{x}$) and we compare its performance with FIR graph filters of different orders. As expected, we can see that in the static case the ARMA graph after K iterations has the same performance as the FIR_K filter and thwy both match the solution of the optimization problem. On the other hand, in the random time-varying graph the ARMA filter outperforms all the FIRs. This is mainly due to its implementation strategy, which allows the ARMAs to handle the graph variations better. Also note that the result obtained from the ARMA_1 in the time-varying scenario quantifies the theoretical derivations in (45) and Theorem 6. Indeed, we can see that the obtained output is close (up to an order 10^{-3}) to the desired frequency response and the convergence is linear.

We can see that, for both the random time-varying and static graph the ARMA graph filter gives a lower error with respect to the solution of the optimization problem. As we have seen before, for static graphs the ARMA filter matches correctly the analytical solution. Meanwhile, when the graph is generated randomly it approximates quite well the latter. On the other hand, the performance of the FIR filters is limited by the fact that they only approximate the solution of the optimization problem. Notice that the FIR output is given after K time instants and then the filter is reset, hence the spikes in the figure.

Approximate design (ideal low pass): We use graph filters of increasing orders, specifically $K = 2, 4$ and 6, to universally approximate a low-pass graph filter with frequency response $g^*(\lambda) = 1$ if $\lambda < 0.5$, and zero otherwise. We consider a graph with 100 nodes living in a square of 1000×1000 meters, with a communication range of 180 meters. We simulated node mobility according to the random waypoint model [30] with a constant speed selected in $[0, 3]$ m/s.

We start with a scenario where only the graph topology changes in time whereas the graph signal remains invariant. Then, we simulate a more general case, where both the graph topology and the graph signal are time-varying. For both scenarios, we perform 20 distinct runs, each lasting 10 minutes and consisting of 600 iterations (one iteration per second). We then compare the response error $\|g - g^*\|/\|g^*\|$ of the ARMA filters with that of the analogous FIR filters while accounting for the initialization phase (we ignore the first 100 iterations). More specifically, at each time instant, we compute $g(\lambda_n) = \hat{y}_n/\hat{x}_n$, where the points $\hat{x}_n \approx 0$ are not considered. Then, it is compared with the desired frequency response at the particular graph frequency λ_n, i.e., $g^*(\lambda_n)$. The statistical significance of our results stems not only by the 20 distinct repetitions, but also by the large number of graph topologies experienced in each run.

Time-varying graph, constant graph signal: For this scenario, $\boldsymbol{x}$ is a random vector with entries selected uniformly distributed in [0, 1]. In Fig. 8 (top) we show the response error for increasingly higher node speeds. As expected, the error increases with speed. Nevertheless, the ARMA filters show a better performance in comparison to their analogous FIR filters. This indicates that the proposed approach handles better time-vaying settings than the FIR filters. Further, we can see that higher order ARMA filters approximate better the desired frequency response (smaller error) when the graph is static. On the other hand, when mobility is present, higher order ARMA recursions lead to a rough approximation due to their slower convergence and the fact that the poles go closer to the unit circle (larger coefficients).

Time-varying graph and graph signal: To conclude, we simulate the more general case where both the graph structure and the graph signal change in time. Simulating a target tracking scenario, we let the signal at each node take a value of zero, unless a node was within 100 meters from a target point, residing at the middle of the 1000×1000 meter simulation area, in which case the node's value was set to one. In Fig. 8 (bottom) we show the response error as a function of the node's speed. It is not surprising that letting the graph signal change over time makes the graph filtering problem harder and the corresponding errors of all graph filters larger. As expected, the error increases with speed. Nevertheless, the ARMA filters show a better performance in comparison to their analogous FIR filters for all cases other than when $K = 2$ and speed is zero (the latter is an artifact of the Shank's method).

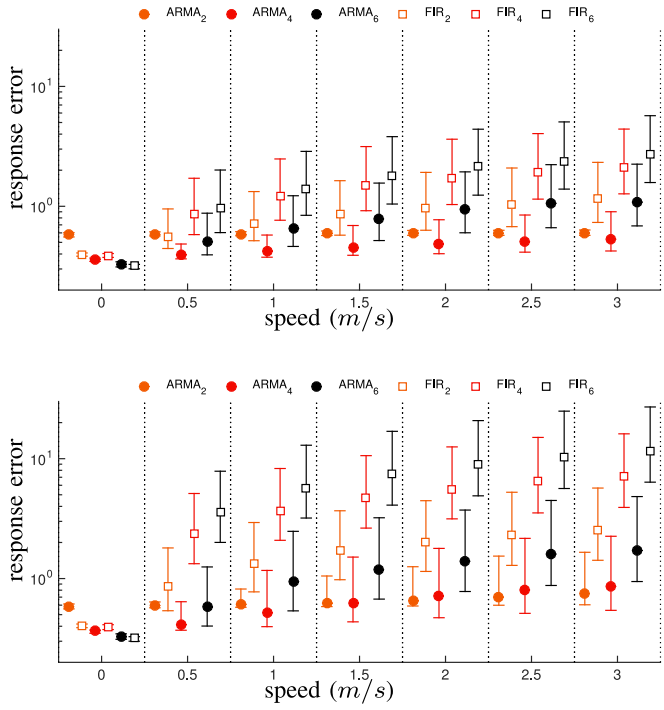

Fig. 8. The effects of the variations only on the graph topology (top) and on both graph and graph signal (bottom). The response error is calculated as $\|g(\lambda) - g^*(\lambda)\|/\|g^*(\lambda)\|$. Each error bar shows the standard deviation of the approximation error over 20 runs. A small horizontal offset is included to improve visibility.

VII. Conclusion

In this work, we presented the ARMA recursion as way of implementing IIR graph filters in a distributed way. We showed two different options to approximate any desired graph frequency response with an ARMA filter of order K, namely the parallel and periodic implementations. Experiments show that, our Shanks-based design method produces stable filter, which can approximate arbitrary well any desired graph frequency response. Furthermore, they attain linear convergence. The proposed ARMA graph filters were shown to provide solutions for two important graph filtering tasks: (i) Tikhonov and Wiener graph denoising and (ii) graph signal interpolation under smoothness assumptions.

Characterized by a rational frequency response, ARMA graph filters can track time-varying input signals. In this case, we showed that our filters naturally extend to a 2-dimensional frequency space simultaneously operating in the graph- and time-frequency domain. In this way, we can distributedly filter a signal jointly in both domains, instead of operating on each of them separately, which leads to higher costs. Though we did not provide solutions for the joint design problem, we illustrated that, due to a connection between the poles in the graph domain and those in the Z-domain, graph filters which are designed only w.r.t. the graph frequency domain, are characterized by a specific temporal behavior. Further, we characterized the

TABLE I
RESIDUES r_k AND POLES p_k OF PARALLEL ARMA$_K$ FILTER, FOR $K = 3, 5$ AND 7

order	r_0, p_0	r_1, p_1	r_2, p_2	r_3, p_3	r_4, p_4	r_5, p_5	r_6, p_6
K=3	10.954 + 0i, -6.666 + 0i	1.275 + 1.005i, 0.202 + 1.398i	1.275 - 1.005i, 0.202 - 1.398i	-	-	-	-
K=5	-7.025 + 0i, -3.674 + 0i	-1.884 - 1.298i, -0.420 + 1.269i	-1.884 + 1.298i, -0.420 - 1.269i	1.433 - 1.568i, 0.703 + 1.129i	1.433 + 1.568i, 0.703 + 1.129i	-	-
K=7	-46.398 + 0i, –3.842 + 0i	-20.207 - 8.343i, 0.102 + 1.427i	-20.207 + 8.343i, 0.102 + 1.427i	-5.205 + 4.946i, -0.785 + 1.128i	-5.205 - 4.946i, -0.785 + 1.128i	3.124 - 10.622i, 0.902 + 1.011i	3.124 + 10.622i, 0.902 - 1.011i

ARMA recursion when also the graph structure varies in time and proved that the linear convergence can be guaranteed also in this setting.

Our future research will be based on finding analytical stable design methods for both 1 and 2-dimensional ARMA recursions. Furthermore, we are also interested to extend the proposed 2-dimensional graph filter to a separable case in order to obtain a disjoint filter design in each domain.

APPENDIX

Table I

For completeness and reproducibility, we include in Table I the filter coefficients of a parallel ARMA$_K$ filter approximating the step function with cut-off $\lambda_c = 0.5$ (i.e., filter response equal to 1 for $\lambda < \lambda_c$ and zero otherwise) for $K = 3, 5, 7$. Higher order filters are omitted due to space considerations.

Proof of Theorem 3

Define matrices $\boldsymbol{\Gamma}_t = \theta_t \mathbf{I} + \psi_t \boldsymbol{L}$ and $\boldsymbol{\Phi}_\Gamma(t, t') = \boldsymbol{\Gamma}_t \boldsymbol{\Gamma}_{t-1} \cdots \boldsymbol{\Gamma}_{t'}$ if $t \geq t'$, whereas $\boldsymbol{\Phi}_\Gamma(t, t') = \mathbf{I}$ otherwise. The output at the end of each period can be re-written as a time-invariant system

$$\boldsymbol{y}_{(i+1)K} = \overbrace{\boldsymbol{\Phi}_\Gamma(K-1, 0)}^{\triangleq \boldsymbol{A}} \boldsymbol{y}_{iK} + \overbrace{\sum_{k=0}^{K-1} \boldsymbol{\Phi}_\Gamma(K-1, k+1) \varphi_k \boldsymbol{x}}^{\triangleq \boldsymbol{B}} \tag{53a}$$

$$\boldsymbol{z}_{(i+1)K} = \boldsymbol{y}_{(i+1)K} + c\boldsymbol{x}. \tag{53b}$$

Both $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same eigenvectors $\boldsymbol{\phi}_n$ as $\boldsymbol{L}$. Notice that (53) resembles (6) and we can proceed in an identical manner. As such, when the maximum eigenvalue of $\boldsymbol{A}$ is bounded by $|*|\, \lambda_{max}(\boldsymbol{A}) < 1$, the steady state of (53b) is

$$\boldsymbol{z} = (I - \boldsymbol{A})^{-1} \boldsymbol{B}\boldsymbol{x} + c\boldsymbol{x} = \sum_{n=1}^{N} \left(c + \frac{\lambda_n(\boldsymbol{B})}{1 - \lambda_n(\boldsymbol{A})} \right) \hat{x}_n \boldsymbol{\phi}_n. \tag{54}$$

To derive the exact response, we exploit the backward product in the definition of $\boldsymbol{\Phi}_\Gamma(t_1, t_2)$ and we obtain

$$\lambda_n(\boldsymbol{\Phi}_\Gamma(t_1, t_2)) = \prod_{\tau=t_1}^{t_2} \lambda_n(\boldsymbol{\Gamma}_t) = \prod_{\tau=t_1}^{t_2} (\theta_\tau + \psi_\tau \lambda_n), \tag{55}$$

which, by the definition of $\boldsymbol{A}$ and $\boldsymbol{B}$, yields the desired frequency response. The linear convergence rate and stability condition follow from the linear convergence of (53b) to $\boldsymbol{y}$ with rate $|\lambda_{max}(\boldsymbol{A})|$.

Proof of Theorem 4

The recursion of a parallel ARMA$_K$ with time-varying input is

$$\boldsymbol{y}_{t+1}^{(k)} = \psi^{(k)} \boldsymbol{L} \boldsymbol{y}_t^{(k)} + \varphi^{(k)} \boldsymbol{x}_t \ (\forall k) \tag{56a}$$

$$\boldsymbol{z}_{t+1} = \sum_{k=1}^{K} \boldsymbol{y}_{t+1}^{(k)} + c\boldsymbol{x}_t, \tag{56b}$$

where $\boldsymbol{y}_t^{(k)}$ is the state of the kth ARMA$_1$, whereas $\boldsymbol{x}_t$ and $\boldsymbol{z}_t$ are the input and output graph signals, respectively. Using the Kronecker product the above takes the more compact form

$$\boldsymbol{y}_{t+1} = (\boldsymbol{\Psi} \otimes \boldsymbol{L})\, \boldsymbol{y}_t + \boldsymbol{\varphi} \otimes \boldsymbol{x}_t \tag{57a}$$

$$\boldsymbol{z}_{t+1} = (\mathbf{1}^\mathsf{T} \otimes \boldsymbol{I}_N) \boldsymbol{y}_{t+1} + c\, \boldsymbol{x}_t, \tag{57b}$$

with $\boldsymbol{y}_t = \left[\boldsymbol{y}_t^{(1)\top}, \boldsymbol{y}_t^{(2)\top}, \cdots, \boldsymbol{y}_t^{(K)\top}\right]^\top$ the $NK \times 1$ stacked state vector, $\boldsymbol{\Psi} = \text{diag}(\psi^{(1)}, \psi^{(2)}, \cdots, \psi^{(K)})$ a diagonal $K \times K$ coefficient matrix, $\boldsymbol{\varphi} = (\varphi^{(1)}, \varphi^{(2)}, \cdots, \varphi^{(K)})^\top$ a $K \times 1$ coefficient vector, and $\mathbf{1}$ the $K \times 1$ one-vector. We therefore have

$$\begin{aligned} \boldsymbol{y}_{t+1} &= (\boldsymbol{\Psi} \otimes \boldsymbol{L})^t\, \boldsymbol{y}_0 + \sum_{\tau=0}^{t} (\boldsymbol{\Psi} \otimes \boldsymbol{L})^\tau\, (\boldsymbol{\varphi} \otimes \boldsymbol{x}_{t-\tau}) \\ &= \left(\boldsymbol{\Psi}^t \otimes \boldsymbol{L}^t\right) \boldsymbol{y}_0 + \sum_{\tau=0}^{t} (\boldsymbol{\Psi}^\tau \boldsymbol{\varphi}) \otimes (\boldsymbol{L}^\tau \boldsymbol{x}_{t-\tau}). \end{aligned}$$

Notice that, when the stability condition $\left\|\psi^{(k)} \boldsymbol{L}\right\| < 1$ is met, $\lim_{t\to\infty} \left\|\left(\boldsymbol{\Psi}^t \otimes \boldsymbol{L}^t\right)\right\| \boldsymbol{y}_0 = 0$. Hence, for sufficiently large t, the ARMA$_k$ output is

$$\begin{aligned} \lim_{t\to\infty} \boldsymbol{z}_{t+1} &= \lim_{t\to\infty} \sum_{\tau=0}^{t} (\mathbf{1}^\mathsf{T} \otimes \boldsymbol{I}_N)\, (\boldsymbol{\Psi}^\tau \boldsymbol{\varphi}) \otimes (\boldsymbol{L}^\tau \boldsymbol{x}_{t-\tau}) + c\, \boldsymbol{x}_t \\ &= \lim_{t\to\infty} \sum_{\tau=0}^{t} \left(\mathbf{1}^\mathsf{T} \boldsymbol{\Psi}^\tau \boldsymbol{\varphi}\right) \otimes (\boldsymbol{L}^\tau \boldsymbol{x}_{t-\tau}) + c\, \boldsymbol{x}_t \\ &= \lim_{t\to\infty} \sum_{\tau=0}^{t} \sum_{k=1}^{K} \varphi^{(k)} \left(\psi^{(k)} \boldsymbol{L}\right)^\tau \boldsymbol{x}_{t-\tau} + c\, \boldsymbol{x}_t, \end{aligned}$$

where we have used the Kronecker product property $(\boldsymbol{A} \otimes \boldsymbol{B})(\boldsymbol{C} \otimes \boldsymbol{D}) = (\boldsymbol{AC}) \otimes (\boldsymbol{BD})$ and expressed the Kronecker

product as the sum of K terms. The transfer matrix $\boldsymbol{H}(z)$ is obtained by taking the Z-transform in both sides and re-arranging the terms

$$\boldsymbol{H}(z) = z^{-1}\sum_{k=1}^{K}\varphi^{(k)}\sum_{\tau=0}^{\infty}\left(\psi^{(k)}\boldsymbol{L}\right)^{\tau} z^{-\tau} + cz^{-1}.$$

Finally, applying the GFT and using the properties of geometric series we obtain the joint transfer function in closed-form expression

$$\begin{aligned} H(z,\mu) &= z^{-1}\sum_{k=1}^{K}\varphi^{(k)}\sum_{\tau=0}^{\infty}\left(\psi^{(k)}\lambda\right)^{\tau} z^{-\tau} + cz^{-1} \\ &= \sum_{k=1}^{K}\frac{\varphi^{(k)}z^{-1}}{1-\psi^{(k)}\lambda z^{-1}} + cz^{-1} \end{aligned}$$

and our claim follows.

Proof of Theorem 5

Recall for comodity $\boldsymbol{\Gamma}_t = \theta_t\mathbf{I} + \psi_t\boldsymbol{L}$ and $\boldsymbol{\Phi}_{\Gamma}(t,t') = \boldsymbol{\Gamma}_t\boldsymbol{\Gamma}_{t-1}\cdots\boldsymbol{\Gamma}_{t'}$ if $t \geq t'$, whereas $\boldsymbol{\Phi}_{\Gamma}(t,t') = \mathbf{I}$ otherwise. Then, expanding recursion (13) for a time-varying input signal, we find that at the end of the i-th period, the filter output is $\boldsymbol{z}_{iK} = \boldsymbol{y}_{iK} + c\boldsymbol{x}_{iK-1}$, where

$$\boldsymbol{y}_{i+1K} = \boldsymbol{\Phi}_{\Gamma}(iK-1,0)\boldsymbol{y}_0 + \sum_{k=0}^{iK-1}\boldsymbol{\Phi}_{\Gamma}(iK-1,k+1)\varphi_k\boldsymbol{x}_k.$$

For sufficiently large i and assuming that the stability condition of Theorem 3 holds, the first term approaches the zero vector and can be ignored without any loss of generality.

We proceed by restricting the input graph signal to $\boldsymbol{x}_{iK} = x_{iK}\boldsymbol{\phi}$, where $\lambda, \boldsymbol{\phi}$ is an eigenpair of $\boldsymbol{L}$ (similarly $\boldsymbol{y}_{iK} = y_{iK}\boldsymbol{\phi}$ and $\boldsymbol{z}_{iK} = z_{iK}\boldsymbol{\phi}$). For compactness we introduce the shorthand notation $\lambda_k = \theta_k + \lambda\psi_k$ and $L = \prod_{\tau=0}^{K-1}\lambda_\tau$. We then have

$$y_{iK} = \sum_{k=0}^{iK-1}\left(\prod_{\tau=k+1}^{iK-1}\lambda_\tau\right)\varphi_k x_k,$$

which, after taking the Z-transform, becomes

$$\begin{aligned} \frac{Y(z)}{X(z)} &= \sum_{k=0}^{iK-1}\left(\prod_{\tau=k+1}^{iK-1}\lambda_\tau\right)\varphi_k z^{k-iK} \\ &= \sum_{j=0}^{i-1}L^{i-j-1}z^{(j-i)K}\left(\sum_{k=0}^{K-1}\left(\prod_{\tau=k+1}^{iK-1}\lambda_\tau\right)\varphi_k z^{k}\right). \end{aligned}$$

The last step exploited the periodicity of coefficients in order to group the common terms of periods $j = 1,\ldots,i-1$. In the limit, the first term approaches

$$\lim_{i\to\infty}\sum_{j=0}^{i-1}L^{i-j-1}z^{(j-i)K} = \lim_{i\to\infty}L^{-1}\sum_{j=0}^{i-1}\left(\frac{L}{z^K}\right)^{i-j} = \frac{1}{z^K - L}$$

Putting everything together, we find that the joint transfer function of the filter is

$$H(z,\mu) = \frac{Z(z)}{X(z)} = \frac{\sum_{k=0}^{K-1}\left(\prod_{\tau=k+1}^{K-1}\lambda_\tau\right)\varphi_k z^k}{z^K - L} + cz^{-1}$$

and, after normalization, the claim (36) follows.

Proof of Theorem 6

We start the proof by substituting the expression (39) for t_1 and t_2 into the numerator of (46). Then, we can write

$$\begin{aligned} \|\boldsymbol{z}_{t_1+1} - \boldsymbol{z}_{t_2+1}\| &= \|\psi^{t_1+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_1,0)\boldsymbol{y}_0 - \psi^{t_2+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_2,0)\boldsymbol{y}_0 \\ &\quad + \varphi\sum_{\tau=0}^{t_1}\psi^{\tau}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_1,t_1-\tau+1)\boldsymbol{x}_{t_1-\tau} + c\boldsymbol{x}_{t_1} \\ &\quad - \varphi\sum_{\tau=0}^{t_2}\psi^{\tau}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_2,t_2-\tau+1)\boldsymbol{x}_{t_2-\tau} - c\boldsymbol{x}_{t_2}\|. \end{aligned} \quad (58)$$

Rearranging the terms, we have

$$\begin{aligned} \|\boldsymbol{z}_{t_1+1} - \boldsymbol{z}_{t_2+1}\| &= \|\psi^{t_1+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_1,0)\boldsymbol{y}_0 - \psi^{t_2+1}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_2,0)\boldsymbol{y}_0 \\ &\quad + \varphi\sum_{\tau=t_2+1}^{t_1}\psi^{\tau}\boldsymbol{\Phi}_{\boldsymbol{L}}(t_1,t_1-\tau+1)\boldsymbol{x}_{t_1-\tau} + c(\boldsymbol{x}_{t_1} - \boldsymbol{x}_{t_2})\| \end{aligned}$$

By using the Cauchy-Schwarz property, the triangle inequality of the spectral norm, and a uniform bound ϱ on the eigenvalues of matrices $\boldsymbol{M}_t$, the above expression simplifies

$$\begin{aligned} \|\boldsymbol{z}_{t_1+1} - \boldsymbol{z}_{t_2+1}\| &\leq \left(|\psi\varrho|^{t_1+1} + |\psi\varrho|^{t_2+1}\right)\|\boldsymbol{y}_0\| \\ &\quad + |\varphi|\sum_{\tau=t_2+1}^{t_1}|\psi\varrho|^{\tau}\|\boldsymbol{x}_{t_1-\tau}\| + |c|\,\|\boldsymbol{x}_{t_1} - \boldsymbol{x}_{t_2}\|. \end{aligned} \quad (59)$$

Leveraging the fact that $|\psi\varrho| < 1$, as well as that $\|\boldsymbol{x}_t\| \leq x_{max}$ for every t, we can express the sum in a closed form

$$\sum_{\tau=t_2+1}^{t_1}|\psi\varrho|^{\tau}\|\boldsymbol{x}_{t_1-\tau}\| \leq x_{max}\left(\frac{|\psi\varrho|^{t_2+1} - |\psi\varrho|^{t_1+1}}{1-|\psi\varrho|}\right). \quad (60)$$

We obtain the desired bound on ϵ_{t_1,t_2} by dividing the above expressions with x_{max} and adjusting the indices.

REFERENCES

[1] A. Loukas, A. Simonetto, and G. Leus, "Distributed autoregressive moving average graph filters," *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 1931–1935, Nov. 2015.

[2] 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 Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[3] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[4] M. G. Rabbat and V. Gripon, "Towards a spectral characterization of signal supported on small-world networks," in *Proc. 2014 IEEE Int. Conf. Acoust., Speech, Signal Process.*, Firenze, Italy, May 2014, pp. 4826–4830.

[5] F. Zhang and E. R. Hancock, "Graph spectral image smoothing using the heat kernel," *Pattern Recognit.*, vol. 41, no. 11, pp. 3328–3342, 2008.

[6] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. Workshops*, 2011, pp. 1–8.
[7] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*. New York, NY, USA: Springer-Verlag, 2003, pp. 144–158.
[8] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, vol. 2, pp. 912–919.
[9] M. Belkin and P. Niyogi, "Semi-supervised learning on Riemannian manifolds," *Mach. Learn.*, vol. 56, nos. 1–3, pp. 209–239, 2004.
[10] S. K. Narang, A. Gadde, and A. Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. 2013 IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 5445–5449.
[11] A. Loukas, M. A. Zúñiga, I. Protonotarios, and J. Gao, "How to identify global trends from local decisions? Event region detection on mobile networks," in *Proc. Int. Conf. Comput. Commun.*, 2014, pp. 1177–1185.
[12] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmonic Anal.*, vol. 30, no. 2, pp. 129–150, 2011.
[13] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning graphs from signal observations under smoothness prior," *IEEE Trans. Signal Process.*, Jun. 2014, submitted to be published.
[14] A. Sandryhaila, S. Kar, and J. M. F. Moura, "Finite-time distributed consensus through graph filters," in *Proc. 2014 IEEE Int. Conf. Acoust., Speech, Signal Process*., 2014, pp. 1080–1084.
[15] S. Safavi and U. A. Khan, "Revisiting finite-time distributed algorithms via successive nulling of eigenvalues," *IEEE Signal Process. Lett.*, vol. 22, no. 1, pp. 54–57, Jan. 2015.
[16] S. Segarra, A. G. Marques, and A. Ribeiro, "Distributed implementation of linear network operators using graph filters," in *Proc. 53rd Allerton Conf. Commun. Control Comput.*, Sep. 2015, pp. 1406–1413.
[17] A. Loukas, M. Cattani, M. A. Zúñiga, and J. Gao, "Graph scale-space theory for distributed peak and pit identification," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2015, pp. 118–129.
[18] X. Shi, H. Feng, M. Zhai, T. Yang, and B. Hu, "Infinite impulse response graph filters in wireless sensor networks," *IEEE Signal Process. Lett.*, vol. 22, no. 8, pp. 1113–1117, Aug. 2015.
[19] A. Loukas, M. A. Zúñiga, M. Woehrle, M. Cattani, and K. Langendoen, "Think globally, act locally: On the reshaping of information landscapes," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2013, pp. 265–276.
[20] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004, sec. 9.3, pp. 466–475.
[21] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*. Hoboken, NJ, USA: Wiley, 2009.
[22] J. L. Shanks, "Recursion filters for digital processing," *Geophysics*, vol. 32, no. 1, pp. 33–51, 1967.
[23] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Proc. Global Conf. Signal Inf. Process.*, 2014, pp. 872–876.
[24] B. Girault, P. Goncalves, E. Fleury, and A. S. Mor, "Semi-supervised learning for graph to signal mapping: A graph signal wiener filter interpretation," in *Proc. 2014 IEEE Int. Conf. Acoust., Speech, Signal Process*., 2014, pp. 1115–1119.
[25] C. Zhang, D. Florencio, and P. Chou, "Graph signal processing—A probabilistic framework," Tech. Rep. MSR-TR-2015–31, Apr. 2015. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=243326
[26] Y. Mao, G. Cheung, and Y. Ji, "Image interpolation for DIBR view synthesis using graph fourier transform," in *Proc. 3DTV-Conf. True Vis. Capture, Transmiss. Display 3D Video*, 2014, pp. 1–4.
[27] T. Zhang, A. Popescul, and B. Dom, "Linear prediction models with graph regularization for web-page categorization," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 821–826.
[28] B. Touri, "Product of random stochastic matrices and distributed averaging," Ph.D. dissertation, Ind. Eng., Univ. Illinois Urbana-Champaign, Champaign, IL, USA, 2011.
[29] A. Loukas and D. Foucard, "Frequency analysis of temporal graph signals," *CoRR*, vol. abs/1602.04434, 2016. [Online]. Available: http://arxiv.org/abs/1602.04434
[30] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "Bonnmotion: A mobility scenario generation and analysis tool," in *Proc. Int. ICST Conf. Simul. Tools Techn.*, 2010, Art. no. 51.

Elvin Isufi (S'15) was born in Albania in 1989. He received the Master of Science degree (*cum laude*) in electronic and telecommunication engineering from the University of Perugia, Perugia, Italy, in 2014. Since November 2014, he has been working toward the Ph.D. degree on signal processing on graphs at Delft University of Technology, Delft, The Netherlands. From November 2013 to August 2014, he was a visiting member in Circuits and Systems Group, Delft University of Technology, where he worked on his master thesis. His research interests include signal processing on graphs, network coding, and underwater communications.

Andreas Loukas received the Doctorate degree in computer science from Delft University of Technology, Delft, The Netherlands, where he focused on distributed algorithms for information processing, and a Diploma in computer science from the University of Patras, Patras, Greece. He is a Research Scientist jointly hosted by the LTS2 and LTS4 Signal Processing Labs of the École Polytechnique Fédérale de Lausanne. His research interest include the intersection of data analysis, graph theory, and signal processing.

Andrea Simonetto (M'12) received the Ph.D. degree in systems and control from Delft University of Technology, Delft, The Netherlands, in 2012. He is currently a Postdoctoral Researcher with the ICTEAM institute, Université Catholique de Louvain, Leuven, Belgium. He was a Postdoctoral Researcher with the Electrical Engineering Department, Delft University of Technology. His current research interests include distributed estimation, control, and optimization.

Geert Leus (F'12) received the M.Sc. and Ph.D. degrees in applied sciences from the Katholieke Universiteit Leuven, Leuven, Belgium, in June 1996 and May 2000, respectively. He is currently an "Antoni van Leeuwenhoek" Full Professor in the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, The Netherlands. His research interests include the area of signal processing for communications. He received the 2002 IEEE Signal Processing Society Young Author Best Paper Award and the 2005 IEEE Signal Processing Society Best Paper Award. He is a Fellow of EURASIP. He was the Chair of the IEEE Signal Processing for Communications and Networking Technical Committee and an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE SIGNAL PROCESSING LETTERS, and the EURASIP *Journal on Advances in Signal Processing*. He is currently a Member-at-Large to the Board of Governors of the IEEE Signal Processing Society and a member of the IEEE Sensor Array and Multichannel Technical Committee. He finally serves as the Editor -in-Chief of the EURASIP *Journal on Advances in Signal Processing*.