Distributed Autoregressive Moving Average Graph Filters

Andreas Loukas, Andrea Simonetto, and Geert Leus

***Abstract*—We introduce the concept of autoregressive moving average (ARMA) filters on a graph and show how they can be implemented in a distributed fashion. Our graph filter design philosophy is independent of the particular graph, meaning that the filter coefficients are derived irrespective of the graph. In contrast to finite-impulse response (FIR) graph filters, ARMA graph filters are robust against changes in the signal and/or graph. In addition, when time-varying signals are considered, we prove that the proposed graph filters behave as ARMA filters in the graph domain and, depending on the implementation, as first or higher order ARMA filters in the time domain.**

***Index Terms*—Distributed time-varying computations, graph filters, graph Fourier transform, signal processing on graphs.**

I. Introduction

THE emerging field of signal processing on graphs [1]–[4] focuses on the extension of classical discrete signal processing techniques to the graph setting. Arguably, the greatest breakthrough of the field has been the extension of the Fourier transform from time signals and images to graph signals, i.e., signals defined on the nodes of irregular graphs. By providing a graph-specific definition of frequency, the graph Fourier transform (GFT) enables us to design filters for graphs: analogously to classical filters, graph filters process a graph signal by amplifying or attenuating its components at specific graph frequencies. Graph filters have been used for a number of signal processing tasks, such as denoising [5], [6], centrality computation [7], graph partitioning [8], event-boundary detection [9], and graph scale-space analysis [10].

Distributed implementations of filters on graphs emerged as a way of increasing the scalability of computation [3], [11], [12]. Nevertheless, being inspired by finite impulse response (FIR) graph filters, these methods are sensitive to time variations (e.g., time-varying signals and/or graphs). An alternative approach, namely distributed infinite impulse response (IIR) graph filters, was recently proposed by Shi *et al.* [13]. Compared to FIR graph filters, IIR filters have the potential to achieve better interpolation or extrapolation properties around the known graph frequencies. Yet the issue of time variations has so far been unresolved.

Manuscript received March 10, 2015; revised June 19, 2015; accepted June 19, 2015. Date of publication June 23, 2015; date of current version June 25, 2015. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Guy Gilboa.

The authors are with the Faculty of EEMCS, Delft University of Technology, 2826 CD Delft, The Netherlands (e-mail: a.loukas@tudelft.nl; a.simonetto@tudelft.nl; g.j.t.leus@tudelft.nl).

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/LSP.2015.2448655

In a different context, we introduced graph-independent IIR filter design, or what we will label here as *universal* IIR filter design (in fact, prior to [13]) using an autoregressive process called the potential kernel [9], [14]. In this letter, we will 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. 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 signal is time-varying). Specifically, our design extends naturally to time-varying signals leading to 2-dimensional ARMA filters: an ARMA filter in the graph domain of arbitrary order and a first order AR (for the periodic implementation) or a higher order ARMA (for the parallel implementation) filter in the time domain, which opens the way to a deeper understanding of graph signal processing, in general. We conclude the letter by displaying preliminary results suggesting that our ARMA filters not only work for continuously time-varying signals, but also degrade gracefully with continuously *time-varying graphs* (e.g., due to mobility).

II. Graph Filters

Consider a graph $G = (V, E)$ of N nodes and let $\boldsymbol{x}$ be a signal defined on the graph, whose i-th component x_i represents the value of the signal at the i-th node, denoted as $u_i \in V$.

Graph Fourier Transform (GFT). The GFT transforms a graph signal into the graph frequency domain: the forward and inverse GFTs of $\boldsymbol{x}$ are $\hat{x}_n = \langle \boldsymbol{x}, \boldsymbol{\phi}_n \rangle$ and $x_n = \langle \hat{\boldsymbol{x}}, \boldsymbol{\phi}_n \rangle$, where $\langle , \rangle$ denotes the inner product. Vectors $\{\boldsymbol{\phi}_n\}_{n=1}^N$ form an orthonormal basis and are commonly chosen as the eigenvectors of a graph Laplacian $\boldsymbol{L}$, such as the discrete Laplacian $\boldsymbol{L}_{\mathrm{d}}$ or Chung's normalized Laplacian $\boldsymbol{L}_{\mathrm{n}}$. For an extensive review of the properties of the GFT, we refer to [2], [4].

To avoid any restrictions on the generality of our approach, in the following we present our results for a *general basis matrix* $\boldsymbol{L}$. We only require that $\boldsymbol{L}$ is *symmetric* and *1-local*: for all $i \neq j$, $\boldsymbol{L}_{ij} = 0$ whenever u_i and u_j are not neighbors and $L_{ij} = L_{ji}$ otherwise.

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

$$\boldsymbol{F}\boldsymbol{x} = \sum_{n=1}^{N} h(\lambda_n)\hat{x}_n\boldsymbol{\phi}_n. \tag{1}$$

Let λ_{min} and λ_{max} be the minimum and maximum eigenvalues of $\boldsymbol{L}$ over *all* possible graphs. The graph frequency response

$h : [\lambda_{min}, \lambda_{max}] \to \mathbb{C}$ controls how much $\boldsymbol{F}$ amplifies the signal component of each graph frequency

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

Distributed graph filters. We are interested in how we can filter a signal with a graph filter $\boldsymbol{F}$ in a *distributed* way, 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. The corresponding filter coefficients are thus independent of the graph and universally applicable.

FIR_K filters. It is well known that we can *approximate* $\boldsymbol{F}$ in a distributed way by using a K-th order polynomial of $\boldsymbol{L}$. Define FIR_K as the K-th order approximation given by

$$\boldsymbol{F}_K = h_0 \mathbf{I} + \sum_{k=1}^{K} h_k \boldsymbol{L}^k,$$

where the coefficients h_i are found by minimizing the least-squares objective $\int_\lambda |\sum_{k=0}^{K} h_k \lambda^k - h^*(\lambda)|^2 \mathrm{d}\lambda$. 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.

The computation of FIR_K is easily performed distributedly. Since $\boldsymbol{L}^K \boldsymbol{x} = \boldsymbol{L}(\boldsymbol{L}^{K-1}\boldsymbol{x})$, each node u_i can compute the Kth-term from the values of the $(K-1)$th-term in its neighborhood. The algorithm terminates after K iterations, and, in total, each node exchanges $\Theta(K \deg u_i)$ bits and stores $\Theta(\deg u_i)$ bits in its memory. However, FIR_K filters exhibit poor performance when the signal or/and graph are time-varying and when there exists asynchronicity among the nodes[1]. In order to overcome these issues and provide a more solid foundation for graph signal processing, we study ARMA graph filters.

III. ARMA Graph Filters

To present universal ARMA graph filters, we first assume that that the graph signal $\boldsymbol{x}$ stays constant over time. We extend our analysis to time-varying signals in Section IV.

A. Distributed Computation

We start by presenting a simple recursion that converges to a filter with a 1st order rational frequency response. We then propose two generalizations with K-th order responses[2]. Using the first, which entails running K 1st order filters in parallel, a node u_i attains fast convergence at the price of exchanging and storing $\Theta(K \deg u_i)$ bits per iteration[3]. By using periodic coefficients, the second algorithm reduces the number of bits exchanged and stored to $\Theta(\deg u_i)$, at almost equivalent (or even *faster*) convergence time.

[1]This because, first the distributed averaging is paused after K iterations, and thus the filter output is *not a steady state*; second the input signal is only considered during the first iteration. To track time-varying signals, the computation should be restarted at each time step, increasing the communication and space complexities to $\Theta(K^2 \deg u_i)$ bits and $\Theta(K \deg u_i)$ bits.

[2]Note that similar structures were independently developed in [13], although based on a different design methodology.

[3]Any values stored are overwritten during the next iteration.

ARMA_1 filters. We will obtain our first ARMA graph filter as an extension of the potential kernel [14]. Consider the following 1st order recursion:

$$\boldsymbol{y}_{t+1} = \psi \boldsymbol{M} \boldsymbol{y}_t + \varphi \boldsymbol{x} \quad \text{and} \quad \boldsymbol{y}_0 \text{ arbitrary}, \tag{3}$$

where $\boldsymbol{y}_t$ if the filter output at iteration t, the coefficients φ, ψ are (for now) arbitrary complex numbers, and $\boldsymbol{M}$ is the translation of $\boldsymbol{L}$ with the minimal spectral radius: $\boldsymbol{M} = \frac{\lambda_{max}-\lambda_{min}}{2}\mathbf{I} - \boldsymbol{L}$. From Sylvester's matrix theorem, matrices $\boldsymbol{M}$ and $\boldsymbol{L}$ have the same eigenvectors and the eigenvalues μ_n of $\boldsymbol{M}$ differ by a translation to those of $\boldsymbol{L}$: $\mu_n = (\lambda_{max} - \lambda_{min})/2 - \lambda_n$.

Proposition 1: The frequency response of ARMA_1 *is* $g(\mu) = \frac{r}{\mu - p}, \quad s.t. \quad p > \frac{\lambda_{max}-\lambda_{min}}{2}$, with the residue r and the pole p given by $r = -\varphi/\psi$ and $p = 1/\psi$, respectively. Recursion (3) converges to it linearly, irrespective of the initial condition $\boldsymbol{y}_0$ and matrix $\boldsymbol{L}$.

Proof: The proof follows from Theorem 1 in [14], in which we replace P with $\boldsymbol{M}$ and $1 - \varphi$ with ψ. □

Recursion (3) leads to a very efficient distributed implementation: 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/space complexity at each iteration is $\Theta(\deg u_i)$ bits.

Parallel ARMA_K filters. We can attain a larger variety of responses by simply adding the output of multiple 1st order filters. Denote with the superscript k the terms that correspond to the k-th ARMA_1 filter ($k = 1, 2, \ldots, K$).

Corollary 1: The frequency response of a parallel ARMA_K is

$$g(\mu) = \sum_{k=1}^{K} \frac{r^{(k)}}{\mu - p^{(k)}} \quad \text{s.t.} \quad |p^{(k)}| > \frac{\lambda_{max} - \lambda_{min}}{2},$$

with $r^{(k)} = -\varphi^{(k)}/\psi^{(k)}$ and $p^{(k)} = 1/\psi^{(k)}$, respectively. Recursion (3) converges to it linearly, irrespective of the initial condition $\boldsymbol{y_o}$ and matrix $\boldsymbol{L}$.

Proof: (Sketch) From Proposition 1, at steady state, we have

$$\boldsymbol{y} = \sum_{k=1}^{K} \boldsymbol{y}^{(k)} = \sum_{k=1}^{K} \sum_{n=1}^{N} \left(\frac{r^{(k)}}{\mu_n - p^{(k)}} \right) \hat{x}_n \boldsymbol{\phi}_n,$$

and switching the sum operators the claim follows. □

The frequency response of a parallel ARMA_K is therefore a rational function with numerator and denominator polynomials of orders $K-1$ and K, respectively[4]. At each iteration, node u_i exchanges and stores $\Theta(K \deg u_i)$ bits.

Periodic ARMA_K filters. We can decrease the memory requirements of the parallel implementation by letting the filter coefficients vary in time. Consider the output of the time-varying recursion

$$\boldsymbol{y}_{t+1} = (\theta_t \mathbf{I} + \psi_t \boldsymbol{M}) \boldsymbol{y}_t + \varphi_t \boldsymbol{x} \quad \text{and} \quad \boldsymbol{y}_0 \quad \text{arbitrary}, \tag{4}$$

[4]By choosing the coefficients properly, we can generalize the rational function to have any degree smaller than K in the denominator. By adding an extra input, we can also obtain order K in the numerator.

This paper previously published in IEEE Signal Processing Letters

every K iterations, where 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 $[0, t/K]$.

Proposition 2: The frequency response of a periodic ARMA_K filter is

$$g(\mu) = \frac{\sum_{\tau=0}^{K-1} \prod_{\sigma=K-\tau}^{K-1} (\theta_\sigma + \psi_\sigma \mu)\, \varphi_{K-\tau-1}}{1 - \left(\prod_{\tau=0}^{K-1} \theta_\tau + \psi_\tau \mu \right)},$$

s.t. the stability constraint $|\prod_{\tau=0}^{K-1} \theta_\tau + \psi_\tau \frac{\lambda_{max}-\lambda_{min}}{2}| < 1$. Recursion (4) converges to it linearly, irrespective of the initial condition $\boldsymbol{y}_0$ and matrix $\boldsymbol{L}$.

Proof: Define matrices $\boldsymbol{\Gamma}_t = \theta_t \mathbf{I} + \psi_t \boldsymbol{M}$ and $\boldsymbol{\Phi}_{t_1,t_2} = \boldsymbol{\Gamma}_{t_1} \boldsymbol{\Gamma}_{t_1-1} \cdots \boldsymbol{\Gamma}_{t_2}$ if $t_1 \geq t_2$ and $\boldsymbol{\Phi}_{t_1,t_2} = \mathbf{I}$ otherwise. The output at the end of each period can be re-written as a time-invariant system

$$\boldsymbol{y}_{(i+1)K} = \boldsymbol{A}\boldsymbol{y}_{iK} + \boldsymbol{B}\boldsymbol{x}, \tag{5}$$

with $\boldsymbol{A} = \boldsymbol{\Phi}_{K-1,0}$, $\boldsymbol{B} = \sum_{\tau=0}^{K-1} \boldsymbol{\Phi}_{K-1,K-\tau} \varphi_{K-\tau-1}$. Assuming that $\boldsymbol{A}$ is non-singular, both $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same eigenvectors $\boldsymbol{\phi}_n$ as $\boldsymbol{M}$ (and $\boldsymbol{L}$). As such, when $|\lambda_{max}(\boldsymbol{A}) < 1$, the steady state of (5) is

$$\boldsymbol{y} = (I - \boldsymbol{A})^{-1} \boldsymbol{B}\boldsymbol{x} = \sum_{n=1}^{N} \frac{\lambda_n(\boldsymbol{B})}{1 - \lambda_n(\boldsymbol{A})} \hat{x}_n \boldsymbol{\phi}_n.$$

To derive the exact response, notice that

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

which, by the definition of $\boldsymbol{A}$ and $\boldsymbol{B}$, yields the desired frequency response. The linear convergence rate follows from the linear convergence of (5) to $mbiy$ with rate $\gamma = |\lambda_{max}(\boldsymbol{A})|$. □

By some algebraic manipulation, we can see that the frequency responses of periodic and parallel ARMA_K filters are equivalent at steady state. In the periodic version, each node u_i stores $\Theta(\deg(u_i))$ bits, as compared to $\Theta(K \deg u_i)$ bits in the parallel one. The low-memory requirements of the periodic ARMA_K render it suitable for resource constrained devices.

B. Filter Design

Given a graph frequency response $g^* : [\mu_{min}, \mu_{max}] \to \mathbb{C}$ and a filter order K, our objective is to find the complex polynomials $p_b(\mu)$ and $p_a(\mu)$ of order $K-1$ and K, respectively, that minimize

$$\int_\mu |\frac{p_b(\mu)}{p_a(\mu)} - g^*(\mu)|^2 \mathrm{d}\mu = \int_\mu \left| \frac{\sum_{k=0}^{K-1} b_k \mu^k}{1 + \sum_{k=1}^{K} a_k \mu^k} - g^*(\mu) \right|^2 \mathrm{d}\mu,$$

while ensuring that the chosen coefficients result in a stable system (see constraints in Corollary 1 and Proposition 2).

Remark 1: Whereas g^* is a function of μ, the desired frequency response $h^* : [\lambda_{min}, \lambda_{max}] \to \mathbb{C}$ is often a function of λ. We attain $g^*(\mu)$ by simply mapping the user-provided response to the domain of μ: $g^*(\mu) = h^*((\lambda_{max} - \lambda_{min})/2 - \lambda)$.

Remark 2: Even if we constrain ourselves to pass-band filters and we consider only the set of $\boldsymbol{L}$ for which $(\lambda_{max} - \lambda_{min})/2 = 1$, it is impossible to design our coefficients based on classical design methods developed for IIR filters (e.g., Butterworth, Chebyshev). 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.

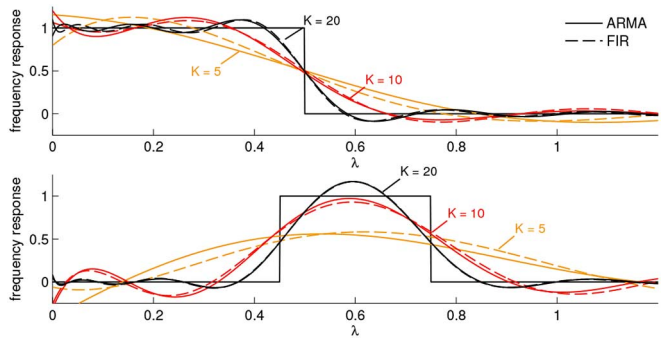

Fig. 1. The frequency response of ARMA_K filters designed by Shank's method and the FIR responses of corresponding order. Here, h^* is a step function (top) and a window function (bottom).

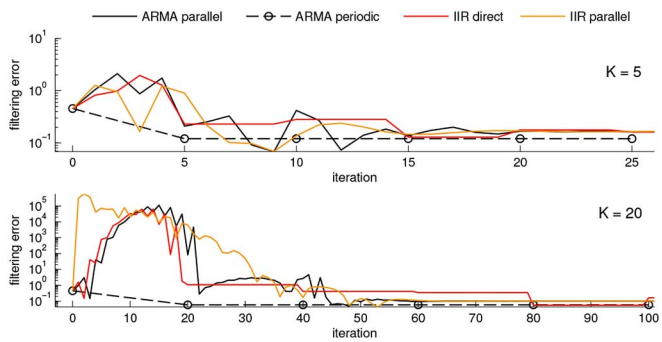

Fig. 2. Convergence comparison of ARMA filters w.r.t. the IIR filters of [13]. The filtering error is $\|\boldsymbol{y}_t - \boldsymbol{y}^*\|_2 / \|\boldsymbol{y}^*\|_2$, where $\boldsymbol{y}^*$ is the desired output.

Design method. Similar to Shank's method [15], we approximate the filter coefficients in two steps:

1) We determine $\{a_k\}_{k=1}^K$, by finding a $\hat{K} > K$ order polynomial approximation $\hat{g}(\mu) = \sum_{k=0}^{\hat{K}} g_k \mu^k$ of $g^*(\mu)$ using polynomial regression, and solving the coefficient-wise system of equations $p_a(\mu)\hat{g}(\mu) = p_b(\mu)$.
2) We determine $\{b_k\}_{k=1}^{K-1}$ by solving the constrained least-squares problem of minimizing $\int_\mu |p_b(\mu)/p_a(\mu) - g^*(\mu)|^2 \mathrm{d}\mu$, w.r.t. $p_b(\mu)$ and s.t. the stability constraints.

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). In the first step of our design, we computed the FIR filter $\hat{g}$ as a Chebyshev approximation of g^* of order $\hat{K} = K + 1$. ARMA responses closely approximate the optimal FIR responses for the corresponding orders (dashed lines). Fig. 2 compares the convergence of our recursions w.r.t. the IIR design of [13] in the same low-pass setting of Fig. 1 (top), running in a network of $n = 100$ nodes[5]. We see how our periodic implementation (only plotted at the end of each period when it is valid) obtains faster convergence. The error of filters increases significantly at the beginning for $K = 20$, due to the filter coefficients, which are very large.

[5]We do not consider the cascade form of [13] since every module in the cascade requires many iterations, leading to a slower implementation.

IV. Time Variations

We now focus on ARMA_K filters and study their behavior when the signal is changing in time, thereby showing how our design extends naturally to the analysis of time-varying signals. We start by ARMA_1 filters: indicate with $\boldsymbol{x}_t$ the graph signal at iteration t. We re-write ARMA_1 recursion as

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

The graph signal $\boldsymbol{x}_t$ can still be decomposed into its graph Fourier coefficients, only now they will be time-varying, i.e., we will have $\hat{x}_{n,t}$. Under the stability condition $\|\psi \boldsymbol{M}\| < 1$, for each of these coefficients we can write its respective graph frequency *and* temporal frequency transfer function as

$$H_1(z, \mu) = \frac{\varphi}{z - \psi \mu}, \tag{7}$$

where z stems from the z-transform. The transfer functions $H_1(z, \mu)$ characterize completely the behavior of ARMA_1 graph filters when the graph does not change (e.g., there is no mobility): when $z \to 1$, we obtain back the constant $\boldsymbol{x}$ result of Proposition 1, while for all the other z we obtain the regular temporal frequency response as well as the graph frequency one. As one can see, 1st order filters are universal ARMA_1 in the graph domain (they do not depend on the particular choice of $\boldsymbol{L}$), as well as 1st order AR filters in the time domain. This result generalizes to parallel and periodic ARMA_K filters.

Parallel ARMA_K. Similarly to Corollary 1, we have:

Proposition 3: Under the same stability conditions of Corollary 1, the transfer function $H_K(z, \mu)$ from the input $\boldsymbol{x}_t$ to the output $\boldsymbol{y}_t$ of a parallel ARMA_K implementation is

$$H_K(z, \mu) = \sum_{k=1}^{K} \frac{\varphi^{(k)}}{z - \psi^{(k)} \mu}.$$

Proof: The recursion (3) for the parallel implementation reads

$$\boldsymbol{y}_{t+1}^{(k)} = \psi^{(k)} \boldsymbol{M} \boldsymbol{y}_t^{(k)} + \varphi^{(k)} \boldsymbol{x}_t, \quad k = 1, \dots, K \tag{8}$$

while the output is $\boldsymbol{y}_t = \sum_{k=1}^{K} \boldsymbol{y}_t^{(k)}$. This can be written in a compact form as

$$\boldsymbol{w}_{t+1} = \boldsymbol{A} \boldsymbol{w}_t + \boldsymbol{B} \boldsymbol{x}_t, \quad \boldsymbol{y}_t = \boldsymbol{C} \boldsymbol{w}_t, \tag{9}$$

where $\boldsymbol{w}_t$ is the stacked version of all the $\boldsymbol{y}_t^{(k)}$, while

$$\boldsymbol{A} = \text{blkdiag}[\psi^{(1)} \boldsymbol{M}, \dots, \psi^{(K)} \boldsymbol{M}], \boldsymbol{B} = [\varphi^{(1)} \mathbf{I}, \dots, \varphi^{(K)} \mathbf{I}]^\top,$$

and $\boldsymbol{C} = \mathbf{1}^\top \otimes \mathbf{I}$. Under the same stability conditions of Corollary 1, the transfer matrix between $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ is

$$\boldsymbol{H}(z) = \boldsymbol{C}(z\mathbf{I} - \boldsymbol{A})^{-1} \boldsymbol{B} = \sum_{k=1}^{K} \varphi^{(k)} (z\mathbf{I} - \psi^{(k)} \boldsymbol{M})^{-1},$$

where we have used the block diagonal structure of $\boldsymbol{A}$. By applying the Graph Fourier transform, the claim follows. □

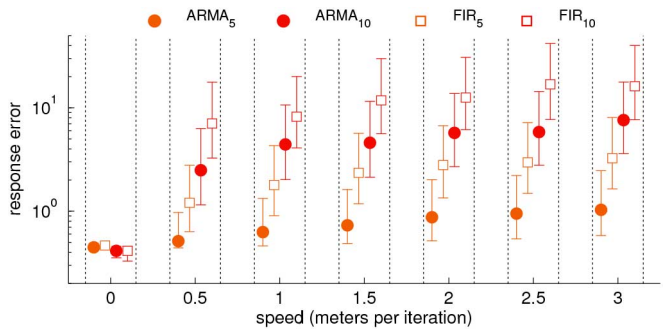

Fig. 3. The effect of node mobility inducing a time-varying signal and graph. Each error bar depicts the standard deviation of the filtering error over ten runs. The response error is $\|g(\mu) - g^*(\mu)\|_2 / \|g^*(\mu)\|_2$. A small horizontal offset was included to improve visibility.

Proposition 3 characterizes the parallel implementation completely: our filters are universal ARMA_K in the graph domain, as well as in the time domain.

Periodic ARMA_K. For tractability, time-varying signals in the periodic implementation will be analyzed assuming that we keep the input $\boldsymbol{x}_t$ *fixed* during the whole period K (though this needs not always hold in practice).

Proposition 4: Let $\boldsymbol{x}_{iK}$ be a sampled version of the input signal $\boldsymbol{x}_t$, sampled at the beginning of each period. Under the same stability conditions of Proposition 2, the transfer function for periodic ARMA_K filters from $\boldsymbol{x}_{iK}$ to $\boldsymbol{y}_{iK}$ is

$$H_K(z, \mu) = \frac{\sum_{\tau=0}^{K-1} \prod_{\sigma=K-\tau}^{K-1} (\theta_\sigma + \psi_\sigma \mu) \varphi_{K-\tau-1}}{z - \left(\prod_{\tau=0}^{K-1} \theta_\tau + \psi_\tau \mu \right)}. \tag{10}$$

Proof: (Sketch) One writes the recursion (5) substituting $\boldsymbol{x}$ with $\boldsymbol{x}_{Kt}$, and proceeds as in the proof of Proposition 2. □

As in the parallel case, this proposition describes completely the behavior of the periodic implementation. In particular, our filters are ARMA_K filters in the graph domain whereas 1st order AR filters in the time domain. The design of $H_K(z, \mu)$ to accommodate both ARMA_K requirements and bandwidth for time-varying signals is left for future research.

Time-varying graphs. We conclude with a preliminary result showcasing that the precision of ARMA graph filters degrades gracefully when signal and graph are both continuously varying. Under the same setting of Fig. 1, we consider $\boldsymbol{x}_t$ to be the node degree, while moving the nodes by a random waypoint model [16] for a duration of 600 seconds. In this way, by defining the graph as a disk graph, the graph and the signal are continuously changing. The node degree is widely used in the literature on graph filters for tasks such as robotic coordination and crowd density estimation [14], [17]. In Fig. 3, we depict the response error after 100 iterations (i.e., well after convergence) in different *mobility* settings: the speed is defined in meters per iteration and the nodes live in a box of 1000×1000 meters with a communication range of 180 meters. As our preliminary results indicate, our designs can tolerate better time-variations. Future research will focus on characterizing and exploiting this property from the design perspective.

REFERENCES

[1] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs: Frequency analysis," *Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, 2014.

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

[3] A. Sandryhaila, S. Kar, and J. M. Moura, "Finite-time distributed consensus through graph filters," in *Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1080–1084, IEEE.

[4] 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, 2013.

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

[6] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Global Conf. Signal and Information Processing (GlobalSIP)*, 2015, IEEE.

[7] L. Page, S. Brin, R. Motwani, and T. Winograd, The pagerank citation ranking: Bringing order to the web Stanford Univ., Stanford, CA, USA, Tech. Rep., 1999.

[8] F. Chung, "The heat kernel as the pagerank of a graph," *Proc. Nat. Acad. Sci.*, vol. 104, no. 50, pp. 19 735–19 740, 2007.

[9] 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 *Int. Conf. Computer Communications (INFOCOM)*, 2014.

[10] A. Loukas, M. Cattani, M. A. Zúñiga, and J. Gao, "Graph scale-space theory for distributed peak and pit identification," in *Int. Conf. Information Processing in Sensor Networks (IPSN)*, 2015, ACM/IEEE.

[11] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *Int. Conf. Distributed Computing in Sensor Systems and Workshops (DCOSS)*, 2011, pp. 1–8, IEEE.

[12] S. Safavi and U. Khan, "Revisiting finite-time distributed algorithms via successive nulling of eigenvalues," *IEEE Signal Process. Lett.*, vol. 22, no. 1, pp. 54–57, Jan. 2015.

[13] 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, 2015.

[14] A. Loukas, M. A. Zúñiga, M. Woehrle, M. Cattani, and K. Langendoen, "Think globally, act locally: On the reshaping of information landscapes," in *Int. Conf. Information Processing in Sensor Networks (IPSN)*, 2013, ACM/IEEE.

[15] J. L. Shanks, "Recursion filters for digital processing," *Geophysics*, vol. 32, no. 1, pp. 33–51, 1967.

[16] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "Bonnmotion: A mobility scenario generation and analysis tool," in *Int. ICST Conf. Simulation Tools and Techniques (SIMUTools)*, 2010, ICST.

[17] A. Loukas, M. Woehrle, P. Glatz, and K. G. Langendoen, "On distributed computation of information potentials," in *ACM Int. Workshop on Foundations of Mobile Computing (FOMC)*, 2012.

This paper previously published in IEEE Signal Processing Letters