Filter Design for Autoregressive Moving Average Graph Filters

Jiani Liu, *Student Member, IEEE*, Elvin Isufi, *Student Member, IEEE*, and Geert Leus, *Fellow, IEEE*

***Abstract*—In the field of signal processing on graphs, graph filters play a crucial role in processing the spectrum of graph signals. This paper proposes two different strategies for designing autoregressive moving average (ARMA) graph filters on both directed and undirected graphs. The first approach is inspired by Prony's method, which considers a modified error between the modeled and the desired frequency response. The second technique is based on an iterative approach, which finds the filter coefficients by iteratively minimizing the true error (instead of the modified error) between the modeled and the desired frequency response. The performance of the proposed algorithms is evaluated and compared with finite impulse response (FIR) graph filters, on both synthetic and real data. The obtained results show that ARMA filters outperform FIR filters in terms of approximation accuracy and they are suitable for graph signal interpolation, compression, and prediction.**

***Index Terms*—Signal processing on graphs, autoregressive moving average graph filters (ARMA), iterative processing, Prony's method.**

I. Introduction

GRAPH signal processing (GSP) extends classical digital signal processing to signals that live on the vertices of irregular graphs [1], [2]. Similar to the frequency analysis of temporal signals, the definition of a Fourier-like transform for graph signals [3] is a handle to process these signals in the so-called graph frequency domain, rather than only in the vertex domain [4]. In this analogy, the frequency components of the graph signal characterize, now, the signal variation over the graph. The graph Fourier transform (GFT) has been defined in two ways, i.e., the projection of the graph signal onto the graph Laplacian eigenspace, see e.g., [1], or onto the eigenspace of the adjacency matrix, see e.g., [4]. The first approach suits better undirected graphs characterized by *real-valued* graph frequencies, whilst the second approach is preferred for directed graphs characterized by *complex-valued* graph frequencies. Note that instead of the graph Laplacian or adjacency matrix, other so-called graph shift operators can also be considered [5].

Together with the GFT, graph filters are a key tool to process the graph signal spectrum, i.e., to amplify or attenuate different graph frequencies. Graph filters find applications in graph signal denoising [6]–[8], smoothing [9], classification [10], sampling [11], recovery [12], and graph clustering [13]. Further, they serve as a basic building block for trilateral graph filters [8], graph filter banks [14], [15], and graph wavelets [16]–[19]. Finite impulse response (FIR) graph filters [4], [20]–[22], direct analogs of temporal FIR filters, are implemented as a polynomial in the graph shift operator, e.g., the graph Laplacian matrix [1], the adjacency matrix [3], or any modification thereof. To accurately match some prescribed specifications in the graph frequency domain, FIR filters require a high filter order leading to a high implementation cost. Furthermore, being matrix polynomials of the graph shift operator, their accuracy remains limited. This issue is especially present when the desired graph frequency response is characterized by sharp transitions, e.g., a step function, which is often required in clustering, graph filter banks, or to discriminate patients with different levels of adaptability in brain networks [23].

FIR filter design is already a well-established theory. One of the most popular approaches to fit the graph frequency response of the FIR filter to a desired spectrum is through solving a linear least squares (LLS) fitting problem [4], which can be carried out for undirected as well as directed graphs. However, since the graph (and thus the set of graph frequencies) is not always perfectly known, techniques have been established to design the FIR filter coefficients without the knowledge of the graph spectrum, by fitting the frequency response over a continuous range of graph frequencies (we call this the universal design approach). The Chebyshev polynomial technique is a popular method in this context, but has only been introduced for undirected graphs in [20]. In this paper, we extend the LLS approach to a universal design method by gridding not only the real line (for undirected graphs), but also the complex plane (for directed graphs) and by subsequently solving the LLS problem for the graph frequencies that are on the grid.

An alternative to FIR graph filters are infinite impulse response (IIR) graph filters, such as the autoregressive moving average (ARMA) graph filters [24], or the gradient descent IIR graph filters [25]. These filters are characterized by a rational graph frequency response, which brings more degrees of freedom to the design. However, the aforementioned works focus on a distributed implementation, which only leads to the modeled

Manuscript received November 22, 2017; revised May 11, 2018 and June 21, 2018; accepted June 21, 2018. Date of publication July 8, 2018; date of current version February 5, 2019. This work was supported by the Circuits and Systems Group, Delft University of Technology, The Netherlands. The work of J. Liu was supported by the China Scholarship Council. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. S. Barbarossa. *(Corresponding author: Jiani Liu.)*

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, Department of Microelectronics, Delft University of Technology, Delft 2628 CD, The Netherlands (e-mail: j.liu-1@tudelft.nl; e.isufi-1@tudelft.nl; g.j.t.leus@tudelft.nl).

Digital Object Identifier 10.1109/TSIPN.2018.2854627

frequency response after an infinite number of iterations [24], [25]. Further, the distributed implementation limits the filter approximation accuracy due to convergence constraints.

To fully exploit the benefits of the rational frequency response, in this paper, we focus on a centralized ARMA filter implementation. In a centralized fashion, the ARMA output can be simply found by solving a linear system of equations, which can be carried out efficiently with first order methods [26] or conjugate gradient (CG) [27]. Based on this centralized implementation, we also propose new ARMA graph filter design methods, which can be adopted when the graph is known or in a universal fashion by gridding the frequency domain (as done for the LLS FIR filter design). The proposed ARMA design and implementation methods work for undirected as well as directed graphs. Throughout this work, we will mainly use FIR filters as a benchmark to assess the performance of the proposed ARMA filters, being their direct competitors, and propose ARMA filters as an alternative for the aforementioned applications.

The paper contribution is threefold:

i) We extend the universal LLS strategy to design FIR graph filters from undirected to directed graphs. For either the normalized Laplacian (undirected graph) or normalized adjacency (directed graph) matrix, we respectively sample the real interval from zero to two or the complex unit disc. The first is done uniformly, whereas the second is done uniformly in amplitude and phase such that the obtained graph frequencies either appear in complex conjugate pairs or are purely real-valued. After the grid points have been determined, LLS is used to fit the response on these grid points.

ii) We present an efficient centralized ARMA filter implementation. ARMA filtering of graph signals is written as a linear system of equations, which can be solved by efficient off-the-shelf algorithms, such as CG [27]. We propose the details of this implementation algorithm and present some simulation results.

iii) We propose two ARMA graph filter design strategies, which can be applied to both directed and undirected graphs. The first one is inspired by Prony's method [28], where a modified error between the modeled and the desired frequency response is minimized. Meanwhile, the second approach minimizes the true error iteratively following the Steigliz-McBride idea [28]. As an initial condition, we can use the solution from the first method, thereby, potentially improving the approximation accuracy of that solution. The two proposed methods can also be extended to a universal design by gridding the graph frequency domain as mentioned earlier.

Several numerical tests validate our findings with both synthetic and real data. We show that the ARMA filters outperform FIR filters in terms of approximation accuracy, even with fewer filter coefficients. In our tests with the real Molene temperature dataset, the ARMA graph filters are used for interpolation purposes (on an undirected graph). With the same dataset, ARMA filters are also utilized to compress (on a directed graph) and predict (on both a directed and undirected graph) the graph signal. The results show that the error resulting from our ARMA filter design is lower than that resulting from an FIR filter with the same number of filter coefficients.

Paper outline: Section II reviews some basic concepts of signal processing on graphs and introduces the concept of universal graph filter design. In Section III, we introduce the ARMA graph filtering, and the related ARMA filter implementation. Section IV contains the filter design problem and the proposed design strategies, while the simulation results are shown in Section V. Finally, the conclusions are drawn in Section VI.

Notation: We indicate by normal letters a or A a scalar variable; a bold lowercase letter $\mathbf{a}$ will represent a vector variable and a bold uppercase letter $\mathbf{A}$ a matrix variable. We indicate the absolute value of a by $|a|$ and the 2-norm of the vector $\mathbf{a}$ and matrix $\mathbf{A}$ by $\|\mathbf{a}\|$ and $\|\mathbf{A}\|$, respectively. a_i or $[\mathbf{a}]_i$ represents the i-th entry of $\mathbf{a}$, and similarly $A_{i,j}$ or $[\mathbf{A}]_{i,j}$ represents the (i,j)-th entry of $\mathbf{A}$. $\mathbf{a}^{(i)}$ will indicate the value of $\mathbf{a}$ after the i-th iteration. $\mathbf{A}^{\dagger}$ represents the pseudo-inverse of the matrix $\mathbf{A}$. We use "$\circ$" to represent the element-wise Hadamard product. We indicate the transpose and Hermitian of the matrix $\mathbf{A}$ by $\mathbf{A}^{\mathrm{T}}$ and $\mathbf{A}^{\mathrm{H}}$, respectively. The complex conjugate of $\mathbf{a}$ and $\mathbf{A}$ are represented as $\mathbf{a}^*$ and $\mathbf{A}^*$, respectively.

II. Problem Statement

This section recalls some background information that will be used throughout the paper. We start with some preliminaries about GSP and graph filtering. Then, we formulate the general problem of graph filter design for some prescribed spectral requirements on both undirected and directed graphs. The notions of universal design and a review of the challenges in designing FIR graph filters conclude the section.

A. Preliminaries

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ the set of N nodes (vertices) and $\mathcal{E}$ the set of E edges. The local structure of $\mathcal{G}$ is captured by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, where $[\mathbf{A}]_{j,i} \neq 0$ if there exists an edge between the nodes v_i and v_j, or by the discrete graph Laplacian $\mathbf{L}_{\mathrm{d}} = \mathbf{D} - \mathbf{A} \in \mathbb{R}^{N\times N}$, where $\mathbf{D}$ is the diagonal degree matrix with diagonal entries defined as $[\mathbf{D}]_{i,i} = \sum_{j=1}^{N}[\mathbf{A}]_{i,j}$ (in-degree matrix) or $[\mathbf{D}]_{i,i} = \sum_{j=1}^{N}[\mathbf{A}]_{j,i}$ (out-degree matrix). Note that for an undirected graph $\mathcal{G}$, every edge between v_i and v_j leads to a similar edge between v_j and v_i, and thus $\mathbf{A}$ is symmetric, i.e., $[\mathbf{A}]_{i,j} = [\mathbf{A}]_{j,i}$. This means that also the discrete graph Laplacian $\mathbf{L}_{\mathrm{d}}$ is symmetric and there is no difference between using the in-degree or out-degree matrix. For directed graphs $\mathcal{G}$, such properties do not hold.

Throughout this paper we will use the adjacency matrix $\mathbf{A}$ as a representative for directed graphs, while for undirected graphs we use as alternative the discrete graph Laplacian $\mathbf{L}_{\mathrm{d}} = \mathbf{D} - \mathbf{A}$. More specifically, we will use their normalized counterparts, i.e., the normalized adjacency matrix $\mathbf{A}_{\mathrm{n}} = \mathbf{A}/\|\mathbf{A}\|$ for directed graphs and the normalized Laplacian matrix $\mathbf{L}_{\mathrm{n}}{=}\mathbf{D}^{-1/2}\mathbf{L}_{\mathrm{d}}\mathbf{D}^{-1/2}$, for undirected graphs. Note that other alternatives can also be used. In short, every one of these graph representations can be referred to as a so-called graph shift operator $\mathbf{S}$, an operator that forms the basis for processing graph signals, as we will see next. In this paper, we restrict ourselves to graphs for which $\mathbf{S}$ is real-valued and diagonalizable, and thus enjoys an eigenvalue decomposition $\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$, with $\mathbf{U}$ the

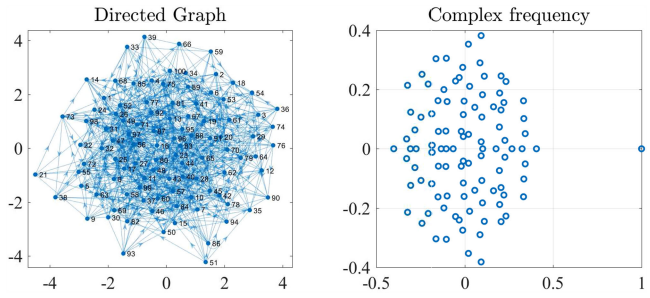


Fig. 1. Directed graph of $N = 100$ nodes with $E = 752$ edges having different weights in the interval $[0, 3]$. Complex-valued frequencies are generated by the eigenvalue decomposition of the normalized adjacency matrix $\mathbf{A}_{\mathrm{n}}$. The "largest" frequency has magnitude one. Some frequencies live on the real axis while the remaining frequencies appear as conjugate pairs in the complex plane.

eigenvector matrix containing as columns the so-called graph modes $\mathbf{u}_1$ up to $\mathbf{u}_N$ and $\mathbf{\Lambda}$ the diagonal eigenvalue matrix containing as diagonal entries the so-called graph frequencies λ_1 up to λ_N (note in this context that $\|\mathbf{S}\| = \max_n |\lambda_n| = |\lambda_{\max}|$).

For undirected graphs, $\mathbf{S}$ is symmetric and normal. The graph frequencies are in this case real-valued, and since $\mathbf{S}$ is real-valued, the graph modes are assumed to be real-valued as well (note that in some cases, they can be chosen to be complex-valued, e.g., for undirected circulant graphs, but that is not assumed in this paper). Specifically, for an undirected graph with $\mathbf{S} = \mathbf{L}_{\mathrm{n}}$, the graph frequencies are in the real interval from zero to two. They can be ordered from small to large, where a smaller value indicates a lower frequency [1].

For directed graphs, the graph frequencies are complex-valued but since $\mathbf{S}$ is real-valued they either appear in complex conjugate pairs or are purely real-valued. Moreover, the related graph modes also appear in complex conjugate pairs or are purely real-valued. Specifically, for the shift operator $\mathbf{S} = \mathbf{A}_{\mathrm{n}}$, the graph frequencies are in the complex unit disc. They can be ordered by the graph total variation of the related graph modes, which is defined as $\mathrm{TV}_{\mathcal{G}}(\mathbf{u}_n) = |1 - \lambda_n / |\lambda_{\max}|| \, \|\mathbf{u}_n\|_1$. In other words, the frequencies are ordered according to the similarity between the nth graph mode and its graph shifted version. Graph frequencies closer to the point $(1, 0)$ in the complex plane will represent lower frequencies in this context [4]. See Fig. 1 for an example of a directed graph and its complex-valued graph frequencies.

We will indicate with the vector $\mathbf{x} \in \mathbb{R}^{N \times 1}$ the real-valued graph signal, i.e., a signal living on the nodes of the graph $\mathcal{G}$, where each value x_i is associated to the node v_i. To obtain the graph frequency representation of $\mathbf{x}$, the eigenvector matrix $\mathbf{U}$ is used to transform the signal into the graph Fourier domain. Specifically, the GFT $\hat{\mathbf{x}}$ of $\mathbf{x}$ and its inverse are, respectively, $\hat{\mathbf{x}} = \mathbf{U}^{-1}\mathbf{x}$ and $\mathbf{x} = \mathbf{U}\hat{\mathbf{x}}$. The following property can now be stated.

Property 1: For either an undirected or directed graph $\mathcal{G}$, let us denote $\hat{x}_n$ as the nth frequency coefficient of the graph signal $\mathbf{x}$. Then, the frequency coefficient $\hat{x}_n$ related to the real-valued graph frequency (mode) λ_n ($\mathbf{u}_n$) is real-valued as well. Meanwhile, the frequency coefficients $\hat{x}_n$ and $\hat{x}_{n'}$ related to the complex conjugate pair of graph frequencies (modes) λ_n and $\lambda_{n'}$ ($\mathbf{u}_n$ and $\mathbf{u}_{n'}$) form a complex conjugate pair as well.

This property is built on the fact that for a real-valued matrix $\mathbf{S}$, eigenvalues and eigenvectors appear in complex conjugate pairs [29], [30]. This also means that if the columns $\mathbf{u}_n$ and $\mathbf{u}_{n'}$ in the matrix $\mathbf{U}$ form a complex conjugate pair, the related rows in the matrix $\mathbf{U}^{-1}$ form a complex conjugate pair. Thus, with $\mathbf{U}^{-1}\mathbf{x}$, the frequency coefficients $\hat{x}_n$ and $\hat{x}_{n'}$ appear as a complex conjugate pair.

For a more in-depth analysis on the basics of the GFT and the ordering of graph frequencies we redirect the reader to [1], [3], and [4].

B. Graph Filtering

A graph filter $\mathbf{G}$ is a function $g(\cdot)$ applied to the shift operator $\mathbf{S}$, i.e., $\mathbf{G} = g(\mathbf{S})$, that allows an eigendecomposition of $\mathbf{G}$ in the form $\mathbf{G} = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^{-1}$, where $g(\mathbf{\Lambda})$ is a diagonal matrix that highlights the filter impact on the graph frequencies $\mathbf{\Lambda}$. More specifically, the filter output $\mathbf{y}$ for a filter input $\mathbf{x}$ can be written as $\mathbf{y} = \mathbf{G}\mathbf{x}$, which in the graph frequency domain can be translated into $\hat{\mathbf{y}} = g(\mathbf{\Lambda})\hat{\mathbf{x}}$, where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ represent the GFT of the input and output signal, respectively. Hence, $g(\mathbf{\Lambda})$ has on the diagonal the frequency response of the filter, which at frequency λ_n we denote as $[g(\mathbf{\Lambda})]_{n,n} = \hat{g}_n$.

Throughout this paper, we will consider different parametrizations of the graph filter function $g(\cdot)$, and thus we will often explicitly write this function as $g(\cdot; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector that contains the graph filter parameters, i.e., filter coefficients, zeros and poles, or any other set of filter parameters. Correspondingly, we can also write $\hat{g}_n$ explicitly as $\hat{g}_n(\boldsymbol{\theta})$. Assuming now that the desired frequency response at frequency λ_n is given by $\hat{h}_n$, the filter parameters $\boldsymbol{\theta}$ can be found by solving

$$\min_{\boldsymbol{\theta}} \sum_{n=1}^{N} |\hat{h}_n - \hat{g}_n(\boldsymbol{\theta})|^2. \tag{1}$$

The desired frequency response $\hat{h}_n$ can originate from different scenarios. For instance, when we focus on graph filter design, i.e., when we want to design a low pass filter to smooth or denoise a graph signal, the desired frequency response $\hat{h}_n$ basically indicates how much we want to attenuate a specific graph mode and thus it will generally be real-valued and symmetric w.r.t. the real axis (for both undirected and directed graphs). Also when we want to do graph signal prediction, as done in [3], we basically want to design an all-pass filter and set $\hat{h}_n$ to be one (and thus real-valued) everywhere. In this case, the cost function (1) will also be weighted, as we will show in the simulations, but the filter design methods that we derive later on can easily be adapted to this weighting. However, for some GSP applications, such as compression, the desired frequency response $\hat{h}_n$ will be the GFT of the signal, for which Property 1 holds.

In any case, whatever the scenario (filter design, prediction, smoothing, denoising, or compression) or type of graph (undirected or directed), the following property holds.

Property 2: As mentioned above, $\hat{h}_n$ is selected either as real-valued and symmetric w.r.t. the real frequency axis, or as

the GFT of a signal. The latter means that $\hat{h}_n$ is real-valued if λ_n is real-valued while $\hat{h}_n$ and $\hat{h}_{n'}$ form a complex conjugate pair if λ_n and $\lambda_{n'}$ form a complex conjugate pair (this is due to Property 1). Put differently, either way we select $\hat{h}_n$, if λ_n is real-valued, then $\hat{h}_n$ is real-valued whereas if λ_n and $\lambda_{n'}$ form a complex conjugate pair, then $\hat{h}_n$ and $\hat{h}_{n'}$ form a complex conjugate pair as well.

C. Universal Filter Design

Since estimating the graph frequencies entails some additional complexity, graph filters are often designed with no explicit knowledge of the graph or the graph frequencies. The desired frequency response is assumed to be a function over a continuous range of frequencies (the real line for undirected graphs or the complex plane for directed graphs). Solving the filter design problem for such a scenario is referred to as *universal filter design*. Following the same LLS approach as in (1), this universal filter design problem can be tackled by discretizing the related continuous frequency range into a finite set of graph frequencies. Then, problem (1) can be solved for this finite set of graph frequencies instead of for the true graph frequencies.

For undirected graphs with $\mathbf{S} = \mathbf{L}_{\text{n}}$, we can consider for instance N different grid points in the interval $[0, 2]$. Note that depending on the graph, we obtain a different eigenvalue spread, e.g., the eigenvalues of an Erdős Rényi graph [31] are in general closely spread around 0 and more widely spread around p, the link probability of the graph (see Fig. 3(f)). However, since we want to be independent of the graph topology, we consider a uniformly-spaced grid in our design. As an example, we show the graph spectrum for an ideal low pass graph filter with cut-off frequency $\lambda_c = 1$ in Fig. 2(a) left.

Alternatively, for directed graphs with $\mathbf{S} = \mathbf{A}_{\text{n}}$, the graph frequencies lie in the complex unit disc. Again trying to avoid any dependence on the graph, we suggest gridding this disc by N complex conjugate pairs of points, as shown in Fig. 2(a) right. Fig. 2(b) again shows an example of an ideal low pass filter in this context. The cut-off frequency λ_c is here defined as the distance from the point $(1, 0)$ in the complex plane, and it is set as $\lambda_c = 1$ in Fig. 2(b). All graph frequencies with a distance to $(1, 0)$ that is smaller than λ_c will be part of the passband since they yield the "smaller" frequencies.

D. FIR Graph Filters

From [3], an FIR graph filter $\mathbf{G}$ of order K can be expressed as a K-th order polynomial in the graph shift operator

$$\mathbf{G} = g(\mathbf{S}; \boldsymbol{\theta}) = \sum_{k=0}^{K} g_k \mathbf{S}^k, \tag{2}$$

with $\boldsymbol{\theta} = \mathbf{g} = [g_0, \ldots, g_K]^T$ collecting the FIR filter coefficients. The filter frequency response at frequency λ_n can be expressed as

$$\hat{g}_n = \sum_{k=0}^{K} g_k \lambda_n^k. \tag{3}$$

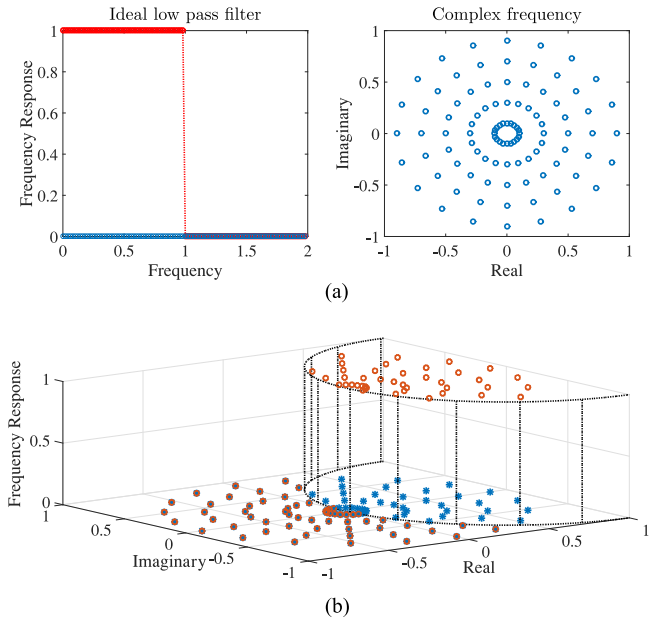


Fig. 2. (a) (Left) Ideal low pass filter response of universal design for undirected graph ($N = 100$). (Right) Universal gridding for directed graph associated with the normalized adjacency matrix $\mathbf{A}_{\text{n}}$ ($N = 100$). (b) Ideal low pass filter response of universal design for directed graph with $N = 100$. The complex frequencies lying inside the circle with radius 1 centered at $(1, 0)$ are "small" frequencies.

By stacking the filter frequency response in $\hat{\mathbf{g}} = [\hat{g}_1, \ldots, \hat{g}_N]^{\text{T}}$, we obtain the relation

$$\hat{\mathbf{g}} = \boldsymbol{\Psi}_{K+1} \mathbf{g}, \tag{4}$$

where $\boldsymbol{\Psi}_{K+1}$ is the $N \times (K + 1)$ Vandermonde matrix with entries $[\boldsymbol{\Psi}]_{n,k} = \lambda_n^{k-1}$. Assuming the desired frequency response is given by $\hat{\mathbf{h}} = [\hat{h}_1, \ldots, \hat{h}_N]^{\text{T}}$, problem (1) can now be written as the following linear least squares (LLS) problem

$$\min_{\mathbf{g}} \|\hat{\mathbf{h}} - \boldsymbol{\Psi}_{K+1} \mathbf{g}\|^2. \tag{5}$$

The solution of this LLS problem is given by

$$\mathbf{g} = \boldsymbol{\Psi}_{K+1}^{\dagger} \hat{\mathbf{h}}, \tag{6}$$

where $\boldsymbol{\Psi}_{K+1}^{\dagger}$ is the pseudo-inverse of $\boldsymbol{\Psi}_{K+1}$. As shown in [4], [21], $\boldsymbol{\Psi}_{K+1}$ needs to be well-conditioned for this approach to work well. This will only be the case for small graph sizes N and/or small filter orders K. Note that to improve the conditioning, close eigenvalues could be grouped together under the assumption that the desired filter response on those eigenvalues is equal. In any case, the FIR filter order K needs to be small and because of the nature of the polynomial fitting problem, this will lead to a limited accuracy of the FIR filter.

For (3) to make sense as a graph filter that will be applied to a real-valued graph signal $\mathbf{x}$, we want the FIR filter coefficients $\mathbf{g}$ to be real-valued. The next Proposition shows that this is the case.

Proposition 1: Under Property 2, the FIR filter coefficients $\mathbf{g}$ obtained by solving (5) are real-valued.

Proof: The proof can be found in Appendix A. ■

III. ARMA Graph Filter and Implementation

To improve the approximation accuracy and reduce the number of required filter coefficients w.r.t. the FIR filter, we now consider applying an ARMA filter to the graph signal $\mathbf{x}$. In this section, we first introduce the ARMA graph filtering problem. Then, a centralized ARMA filter implementation is presented, and some issues related to the corresponding filter design problem are highlighted. Solutions to this ARMA filter design problem are presented in Section IV.

A. ARMA Graph Filter

From [24], and similar to temporal ARMA filters [28], an ARMA graph filter is characterized by a rational polynomial in the graph shift operator

$$\mathbf{G} = g(\mathbf{S};\boldsymbol{\theta}) = \left(\sum_{p=0}^{P} a_p \mathbf{S}^p\right)^{-1} \sum_{q=0}^{Q} b_q \mathbf{S}^q, \tag{7}$$

where $\boldsymbol{\theta} = [\mathbf{a}^T, \mathbf{b}^T]^T$ with $\mathbf{a} = [a_0, \ldots, a_P]^T$ and $\mathbf{b} = [b_0, \ldots, b_Q]^T$ collecting the ARMA filter coefficients. This allows us to express the filter frequency response at frequency λ_n as

$$\hat{g}_n = \frac{\sum_{q=0}^{Q} b_q \lambda_n^q}{\sum_{p=0}^{P} a_p \lambda_n^p}. \tag{8}$$

Stable ARMA filters are obtained when $\sum_{p=0}^{P} a_p \mathbf{S}^p$ is invertible, or equivalently, when $\sum_{p=0}^{P} a_p \lambda_n^p$ is different from zero for all $n = 1, 2, \ldots, N$. This stability condition is less critical as in the time domain, which is mainly due to the fact that a graph signal is finite-length whereas a temporal signal is infinite-length. Hence, there is no big risk of the filter output growing unbounded.

Note that for simplicity reasons we define the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ in an ambiguous way since multiplying both $\mathbf{a}$ and $\mathbf{b}$ with the same constant will not change the ARMA graph filter. Hence, whenever we design $\mathbf{a}$ and $\mathbf{b}$, we will remove this ambiguity by constraining the first AR coefficient to be one, i.e., $a_0 = 1$, which is rather standard.

B. Implementation of ARMA Graph Filter

From (7), it is clear that the relation between the output $\mathbf{y}$ and the input $\mathbf{x}$ of an ARMA graph filter is given by

$$\left(\sum_{p=0}^{P} a_p \mathbf{S}^p\right)\mathbf{y} = \left(\sum_{q=0}^{Q} b_q \mathbf{S}^q\right)\mathbf{x}. \tag{9}$$

Hence, by defining the matrices

$$\mathbf{P} = \sum_{p=0}^{P} a_p \mathbf{S}^p, \qquad \mathbf{Q} = \sum_{q=0}^{Q} b_q \mathbf{S}^q, \tag{10}$$

we can express (9) in the compact form

$$\mathbf{P}\mathbf{y} = \mathbf{Q}\mathbf{x}. \tag{11}$$

To compute the filter output $\mathbf{y}$ in (11), we can first calculate the right-hand side denoted for commodity as $\mathbf{z} = \mathbf{Q}\mathbf{x}$ (which corresponds to pre-filtering $\mathbf{x}$ with an FIR filter) and then $\mathbf{y}$ is found by simply solving the linear system

$$\mathbf{P}\mathbf{y} = \mathbf{z}. \tag{12}$$

Note that there are several efficient methods to solve (12), like first order methods [26], the power method [32], and conjugate gradient (CG) [27]. Their computational cost reduces significantly for sparse matrices $\mathbf{S}$, i.e., for sparse graphs [33].

In this work we consider the CG method [27] to implement ARMA graph filters in the vertex domain. As shown in Algorithm 1, the CG approach has a computational complexity that scales linearly in the number of edges E. Specifically, we first need to compute $\mathbf{z} = \mathbf{Q}\mathbf{x}$, which by following the efficient implementation [34] requires Q multiplications with the shift operator $\mathbf{S}$ since the terms can be computed as $\mathbf{S}^k\mathbf{x} = \mathbf{S}(\mathbf{S}^{k-1}\mathbf{x})$ leading to an overall complexity of $O(QE)$. Then, in each iteration i of the CG it is required to compute the term $\mathbf{P}\mathbf{d}^{(i)}$, which if computed in the same way as $\mathbf{z}$ requires a computational effort of order $O(PE)$. Thus, if considering that the CG is arrested after T iterations, the overall implementation cost of the ARMA graph filter is of order $O((PT+Q)E)$. We would like to highlight that the ARMA filter output with CG is computed without explicitly building the matrices $\mathbf{P}$ and $\mathbf{Q}$, and only considering their application to a specific vector.

In Section V, we analyze the tradeoff between the computational implementation cost and approximation accuracy induced by the CG approach.

Algorithm 1: Conjugate Gradient.

Input: $\mathbf{y}^{(0)}$, $\mathbf{x}$, coefficients a_p, b_q
accuracy ε, number of iterations T
Initialization: $\mathbf{z}$, $\mathbf{P}\mathbf{y}^{(0)}$ (using $\mathbf{S}^k\mathbf{y}^{(0)} = \mathbf{S}(\mathbf{S}^{k-1}\mathbf{y}^{(0)})$)
$\mathbf{d}^{(0)} = \mathbf{r}^{(0)} = \mathbf{z} - \mathbf{P}\mathbf{y}^{(0)}$,
$\delta^{(0)} = \delta^{new} = \mathbf{r}^{(0)T}\mathbf{r}^{(0)}$
Iteration: while $i < T$ and $\delta^{new} > \varepsilon^2\delta^{(0)}$
$\omega^{(i)} = \frac{\delta_{new}}{\mathbf{d}^{(i)T}\mathbf{P}\mathbf{d}^{(i)}}$
$\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \omega^{(i)}\mathbf{d}^{(i)}$,
$\mathbf{r}^{(i+1)} = \mathbf{r}^{(i)} - \omega^{(i)}\mathbf{P}\mathbf{d}^{(i)}$
$\delta^{old} = \delta^{new}$, $\delta^{new} = \mathbf{r}^{(i+1)T}\mathbf{r}^{(i+1)}$
$\varphi^{(i+1)} = \frac{\delta^{new}}{\delta^{old}}$, $\mathbf{d}^{(i+1)} = \mathbf{r}^{(i+1)} + \varphi^{(i+1)}\mathbf{d}^{(i)}$
$i = i + 1$
Output: $\mathbf{y}^{(i+1)}$

IV. ARMA Graph Filter Design

This section contains the proposed ARMA filter design methods. We start with a discussion of the ARMA design problem, followed by two approaches inspired by Prony's method, and finally an iterative approach.

A. ARMA Design Problem

As discussed in Subsection II-B, we would like to find the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ such that a desired frequency

response $\hat{h}_n$ is matched, where the latter can be a desired filter shape (for filter design, smoothing, or denoising) or the GFT of a graph signal (for compression or prediction). In this context, note that many desired responses $\hat{h}_n$ already have the shape of an ARMA filter, e.g., for Tikhonov denoising or interpolation, which means no explicit fitting is required in that case.

More specifically, adapting (1) to our ARMA filter design problem, we want to minimize the following error

$$e_n = \hat{h}_n - \frac{\sum_{q=0}^{Q} b_q \lambda_n^q}{\sum_{p=0}^{P} a_p \lambda_n^p}. \tag{13}$$

Since (13) is nonlinear in $\mathbf{a}$ and $\mathbf{b}$, classical approaches like Prony's method [28] consider minimizing the following modified error

$$e'_n = \hat{h}_n \left(\sum_{p=0}^{P} a_p \lambda_n^p \right) - \sum_{q=0}^{Q} b_q \lambda_n^q. \tag{14}$$

The latter is clearly not equivalent to (13) but it is linear in $\mathbf{a}$ and $\mathbf{b}$.

In the sequel, our goal will be to find $\mathbf{a}$ and $\mathbf{b}$ that minimize (13) or (14) in the mean square sense, subject to $a_0 = 1$ as mentioned before. Similar to the FIR filter, if we want the ARMA filter to make sense as a graph filter that will be applied to a real-valued graph signal $\mathbf{x}$, we want the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ to be real-valued. We will show that this is the case for the different proposed approaches. Finally, note that, similar to Prony's method [28], the non-convex stability constraint $\sum_{p=0}^{P} a_p \lambda_n^p \neq 0$ will be ignored in the rest of the paper, but it can easily be checked after the design.

B. Methods Inspired by Prony

Prony's LS: To start, let us first stack e_n from (13) in the vector $\mathbf{e} = [e_1, \ldots, e_N]^{\mathrm{T}}$, which can be expressed as

$$\mathbf{e} = \hat{\mathbf{h}} - \mathrm{diag}(\mathbf{\Psi}_{P+1}\mathbf{a})^{-1}\mathbf{\Psi}_{Q+1}\mathbf{b}. \tag{15}$$

As we mentioned before, this nonlinear function is hard to handle and thus we focus on the modified error. Stacking e'_n from (14) in the vector $\mathbf{e}' = [e'_1, \ldots, e'_N]^{\mathrm{T}}$, we obtain the simpler linear expression

$$\mathbf{e}' = \hat{\mathbf{h}} \circ (\mathbf{\Psi}_{P+1}\mathbf{a}) - \mathbf{\Psi}_{Q+1}\mathbf{b} \tag{16}$$

$$= [\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a} - \mathbf{\Psi}_{Q+1}\mathbf{b}, \tag{17}$$

where "$\circ$" represents the element-wise Hadamard product and $\mathbf{1}_{P+1}$ is the $(P+1) \times 1$ all-one vector.

Minimizing $\|\mathbf{e}'\|^2$ over $\mathbf{a}$ and $\mathbf{b}$ leads to the following LLS problem

$$\min_{\mathbf{a},\mathbf{b}} \left\| [\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}}), -\mathbf{\Psi}_{Q+1}] \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \right\|^2, \text{ s.t. } a_0 = 1, \tag{18}$$

which can be solved efficiently. The next Proposition shows that the obtained $\mathbf{a}$ and $\mathbf{b}$ vectors are real-valued.

Proposition 2: Under Property 2, the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ obtained by solving (18) are real-valued.

Proof: The proof is similar to the proof of Proposition 1.■

Prony's projection: Since Prony's LS approach addresses the modified error (14) and not the desired error (13), we here consider a way to partly overcome this limitation, and potentially improve the approximation accuracy of (18). We use the orthogonal subspace projection approach [35] to rephrase (16) as a function of only the denominator coefficients $\mathbf{a}$. Then, with the obtained solution for $\mathbf{a}$, the original error (13) can be minimized to find the numerator coefficients $\mathbf{b}$. This approach can be interpreted as Shanks' method similar to that used in [24].

Let us start by considering the orthogonal projection matrix onto the orthogonal complement of the range of $\mathbf{\Psi}_{Q+1}$

$$\mathbf{P}^{\perp}_{\mathbf{\Psi}_{Q+1}} = \mathbf{I}_N - \mathbf{\Psi}_{Q+1}\mathbf{\Psi}^{\dagger}_{Q+1}, \tag{19}$$

where $\mathbf{\Psi}_{Q+1}$ is better conditioned than $\mathbf{\Psi}_{K+1}$ used to design an FIR graph filter, because $Q < K$ and removing columns from a tall matrix improves its condition number. Then, the modified error (16) can be reshaped as

$$\mathbf{e}'' = \mathbf{P}^{\perp}_{\mathbf{\Psi}_{Q+1}}[\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a} - \mathbf{P}^{\perp}_{\mathbf{\Psi}_{Q+1}}\mathbf{\Psi}_{Q+1}\mathbf{b}, \tag{20}$$

where the second term on the right hand side of (20) is zero. As shown in [36], [35], this projection operator preserves the solution for $\mathbf{a}$ when minimizing (20) instead of (16). Hence, after the projection, the LLS problem for solving $\mathbf{a}$ becomes

$$\min_{\mathbf{a}} \|\mathbf{P}^{\perp}_{\mathbf{\Psi}_{Q+1}}[\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a}\|^2, \text{ s.t. } a_0 = 1. \tag{21}$$

The reason why we prefer solving (21) over (18) for finding a solution for $\mathbf{a}$ is the computational complexity. Finally, the vector $\mathbf{b}$ can be obtained using (13) after plugging in the solution for $\mathbf{a}$ obtained from (21). In other words, $\mathbf{b}$ is found by solving

$$\min_{\mathbf{b}} \|\hat{\mathbf{h}} - \mathrm{diag}(\mathbf{\Psi}_{P+1}\mathbf{a})^{-1}\mathbf{\Psi}_{Q+1}\mathbf{b}\|^2. \tag{22}$$

As before, we can again show that this solution for $\mathbf{a}$ and $\mathbf{b}$ is real-valued.

Proposition 3: Under Property 2, the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ obtained by solving (21) and (22) are real-valued.

Proof: The proof is similar to the proof of Proposition 1.■

We would like to remark that this version of Prony's projection approach has a conceptual difference with the method presented in [24]. While in [24] the desired frequency response is first fitted with an FIR filter and then the denominator coefficients are found to match that response, we here aim at approaching directly the desired response rather than its FIR approximation. In parallel to the classical literature [28], our approach can be considered as a reshaping of the Padé approximation which first is solved for the denominator coefficients $\mathbf{a}$ and then for the numerator coefficients $\mathbf{b}$. As we show in Section V, the Prony's projection approach improves in general the approximation accuracy of (18).

C. Iterative Approach

In this section, we present the iterative approach to design the ARMA coefficients. The idea consists of updating recursively the filter coefficients, while minimizing the original error (13). We first reformulate the problem to make it amenable to our iterative approach and then use a variant of the Steigliz-McBride

method [28] to implement an iterative algorithm that can be utilized for finding the ARMA graph filter coefficients.

Problem reformulation: The focus in the previous section was on solving (14). This of course comes with a lack of optimality, since our aim is to solve (13). In the iterative approach, instead, we focus directly on minimizing (13).

To ease the notation, let us define

$$\beta_n = \sum_{q=0}^{Q} b_q \lambda_n^q \quad \text{and} \quad \alpha_n = \sum_{p=0}^{P} a_p \lambda_n^p,$$

and rewrite the original error (13) as

$$e_n = \hat{h}_n - \frac{\beta_n}{\alpha_n}. \tag{23}$$

Then, by defining $\gamma_n = 1/\alpha_n$, we have

$$e_n = \hat{h}_n - \beta_n \gamma_n, \tag{24}$$

which can be equivalently expressed as

$$e_n = (\hat{h}_n \alpha_n - \beta_n)\gamma_n. \tag{25}$$

Note that the expression (25) is linear in α_n, β_n and γ_n, if each of them is treated as a separate variable. To avoid inversion issues when $\alpha_n = 0$, we can consider $\gamma_n = 1/(\alpha_n + \rho)$ for some $\rho \approx 0$. Note that if γ_n is fixed, e_n becomes linear in the variables α_n and β_n. This will be our starting point to minimize e_n recursively. In each iteration, having found a new set of solutions for α_n, β_n we can then find a_p and b_q as well as update γ_n.

To follow the convention of the previous sections, we write (25) in a more convenient vector form, by defining the vectors $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^{\mathrm{T}}$, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_N]^{\mathrm{T}}$, and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_N]^{\mathrm{T}}$. Then, the error vector $\mathbf{e} = [e_1, \ldots, e_N]^{\mathrm{T}}$ containing the original error for all graph frequencies can be written as

$$\mathbf{e} = [\hat{\mathbf{h}} \circ \boldsymbol{\alpha} - \boldsymbol{\beta}] \circ \boldsymbol{\gamma}. \tag{26}$$

Iterative algorithm: Let $\boldsymbol{\alpha}^{(i)}$ and $\boldsymbol{\beta}^{(i)}$ respectively denote the estimates of the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, at the i-th iteration. We can then find the value of $\boldsymbol{\gamma}$ as an element-wise inversion of $\boldsymbol{\alpha}^{(i)}$, which we label as $\boldsymbol{\gamma}^{(i)}$,

$$\boldsymbol{\gamma}^{(i)} = \left[\frac{1}{\alpha_1^{(i)} + \rho} \; \frac{1}{\alpha_n^{(i)} + \rho} \; \cdots \; \frac{1}{\alpha_N^{(i)} + \rho} \right]^T. \tag{27}$$

Using this value for $\boldsymbol{\gamma}$, we obtain the updated error

$$\mathbf{e}^{(i+1)} = (\hat{\mathbf{h}} \circ \boldsymbol{\alpha}) \circ \boldsymbol{\gamma}^{(i)} - \boldsymbol{\beta} \circ \boldsymbol{\gamma}^{(i)}, \tag{28}$$

which is linear in the unknown variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Minimizing this error leads to the updated values $\boldsymbol{\alpha}^{(i+1)}$ and $\boldsymbol{\beta}^{(i+1)}$. This procedure is then repeated till a desirable solution is obtained.

To formalize this iteration, and express it as a direct function of the true filter coefficients $\mathbf{a}$ and $\mathbf{b}$, we can reformulate (28) as

$$\mathbf{e}^{(i+1)} = \mathbf{H}^{(i)}\mathbf{a} - \mathbf{B}^{(i)}\mathbf{b}, \tag{29}$$

where $\mathbf{H}^{(i)} = (\boldsymbol{\gamma}^{(i)}\mathbf{1}_{P+1}^{\mathrm{T}}) \circ \boldsymbol{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})$ and $\mathbf{B}^{(i)} = (\boldsymbol{\gamma}^{(i)}\mathbf{1}_{Q+1}^{\mathrm{T}}) \circ \boldsymbol{\Psi}_{Q+1}$. The specific derivations that lead to (29) can be found in Appendix B.

Algorithm 2: Iterative Approach.

Input: $\mathbf{a}^{(0)}$, $\hat{\mathbf{h}}$, number of iterations τ, threshold δ_c

Initialization: $\boldsymbol{\gamma}^{(0)}, \mathbf{H}^{(0)}, \mathbf{B}^{(0)}, \hat{\mathbf{g}}^{(0)}, \mathbf{e}^{(0)}$

Iteration : while $i < \tau$ and $\delta < \delta_c$

  solve $\min_{\mathbf{a},\mathbf{b}} \left\| \left[\mathbf{H}^{(i)}, -\mathbf{B}^{(i)} \right] \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \right\|^2$ s.t. $a_0 = 1$.

  return $\mathbf{a}^{(i+1)}$, $\mathbf{b}^{(i+1)}$

  compute $\hat{\mathbf{g}}^{(i+1)}$, $\mathbf{e}^{(i+1)}$, $\delta = \|\mathbf{e}^{(i+1)} - \mathbf{e}^{(i)}\|$

  update $\boldsymbol{\gamma}^{(i+1)}$

  $i = i + 1$

Output: $\mathbf{a}^{(i+1)}$, $\mathbf{b}^{(i+1)}$

With this in place, the filter coefficients at the $(i+1)$-th iteration are found by solving

$$\min_{\mathbf{a},\mathbf{b}} \left\| \left[\mathbf{H}^{(i)}, -\mathbf{B}^{(i)} \right] \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \right\|^2 \text{ s.t. } a_0 = 1. \tag{30}$$

The solutions $\mathbf{a}^{(i+1)}$ and $\mathbf{b}^{(i+1)}$ are again real-valued as shown in the following Proposition.

Proposition 4: Under Property 2, the ARMA filter coefficients $\mathbf{a}$ and $\mathbf{b}$ obtained by solving (30) are real-valued.

Proof: The proof is similar to the proof of Proposition 1.■

For the above two design methods, the design cost of Prony's method is related to the LLS solution which requires $O((P+Q+1)^2 N)$ operations, while for the iterative approach, the total design cost is τ times leading to a cost of $O(\tau(P+Q+1)^2 N)$. Since the number of nodes N is much smaller than the number of edges E, the design cost is smaller than the implementation cost. Algorithm 2 summarizes the iterative approach.

Remark 1: We stop the iterations when δ, representing the error difference between two successive iterations, is smaller than a given threshold δ_c. However, depending on the specific combination of P and Q, the method does not always converge fast enough or it does not converge at all. For those cases, we consider a maximum number of iterations τ and search for the minimum error over all iterations. We then assume that this iteration provides the solution to the problem. As we will see in the numerical section, for a fixed order K, the best performance for $P + Q \leq K$ always leads to a significant improvement in approximation accuracy over the former methods. However, for a fixed order K, some combinations of P, Q yield instabilities around the cut-off frequency. The latter is especially present in Prony's method. Therefore, a search over different combinations of P, Q is recommended.

Remark 2: For $\boldsymbol{\gamma}^{(0)} = \mathbf{1}$, the LLS procedure (18) can be seen as a special case of the iterative approach. With $\boldsymbol{\gamma}^{(0)} = \mathbf{1}$, the formulation of the iterative approach degenerates into the LLS solution, and the approximation error changes from the original error (13) to the modified error (14). However, since Prony's projection approach leads to better results that Prony's LS approach, we prefer the latter to initialize the iterative approach.

V. Numerical Data

In this section, we present our numerical evaluation of the proposed methods and compare them with the FIR graph filters. The performance is tested with both synthetic and real data. Our tests with the Molene dataset[1] show that ARMA filters are more suitable than FIR filters for lossy data compression, where we can save up to 50% of memory with very little error. Further, we apply ARMA filters in the context of prediction (as in [3]) and we show that ARMA graph filters outperform FIR graph filters, where with only 4 bits we achieve a reconstruction error of 10^{-3}. Throughout our simulations we make use of the GSPBox [37].

A. Synthetic Simulation Results

In this section, we evaluate the performance of the proposed design algorithms in approximating a desired frequency response. The performance is assessed for two different settings, namely a *universal* filter design (see Section II-C) and a filter design for an Erdős Rényi (ER) graph. For both cases we consider $N = 100$ grid points / nodes.[2] In both settings, the goal is to approximate the ideal low-pass frequency response introduced in Section II and illustrated in Fig. 2.

Universal design: For the universal design, we follow the approach discussed in Section II. For an undirected graph, we consider $\mathbf{S} = \mathbf{L}_{\rm n}$ and sample the interval $[0, 2]$ uniformly. For a directed graph, we consider $\mathbf{S} = \mathbf{A}_{\rm n}$ and sample the complex unit disc uniformly in amplitude and phase. We assume $N = 100$ grid points for both types of graphs.

Design for Erdős Rényi graph: For the undirected ER graph [31], we assume that a pair of nodes is connected with a probability $p = 0.1$ and the shift operator is again $\mathbf{S} = \mathbf{L}_{\rm n}$. Due to the graph randomness, we always average the results over 100 different realizations.

In the sequel, we analyze the design methods proposed in Section IV and compare them to the related FIR filter design. If not mentioned otherwise, we design the FIR filter using the LLS approach of (6) (FIR-LLS, or simply FIR). The universal FIR design for undirected graphs sometimes also follows the Chebyshev design of [20] (FIR-Cheby). We compare the ARMA(P, Q) filter to a FIR(K) graph filter where $P + Q \leq K$ is satisfied. We look for all combinations of P and Q that satisfy $P + Q \leq K$ and pick the combination leading to the best result. Since we want the overall order of the designed ARMA graph filter to be small, we only investigate the range $2 \leq K \leq 30$. We measure the approximation accuracy with the root normalized mean square error (RNMSE) of the frequency response of the filter:

$$\text{RNMSE} = \frac{\|\hat{\mathbf{h}} - \hat{\mathbf{g}}\|}{\|\hat{\mathbf{h}}\|}. \tag{31}$$

Note that, for a directed graph with complex frequencies, since the filter response can be complex-valued, we only compute the approximation error for the amplitude (absolute value) of the filter response under the assumption that the desired frequency response is real.

Performance analysis: In Fig. 3 we show the RNMSE for the Prony's inspired methods and the iterative approach. Specifically, the depicted RNMSE in Fig. 3(a) (b) and (e) are related to the best combination (P, Q) for each particular K such that $P + Q = K$. The iterative approach is initialized with the solution of Prony's projection method (21) and (22), to show its potential in improving the RNMSE. Additionally, the FIR, ARMA$_K$ [24] and IIR [25] performances are plotted as a benchmark.

Based on these results, we can make the following observations:

i) We can notice that the FIR (FIR-LLS or FIR-Cheby) approximation errors for both universal designs (Fig. 3(a), (b)) and the design for the ER graph (Fig. 3(e)) are the highest, except when $K \leq 5$. Further, the FIR approximation accuracy, even when designed for the specific set of ER graph frequencies, does not improve with the order K. We believe that this effect is due to the eigenvalue spread of the ER graph, since some of its eigenvalues are more closely spaced than in a uniform grid (see e.g., Fig. 3(f)).

ii) Compared to Prony's method, the iterative approach has a larger design cost but improves the approximation for higher order K. Prony's method gives a comparable performance to the iterative approach only up to $K = 8$. We see that Prony's LS approach is not suitable for the ER graph when $K \leq 5$, while for a universal design approach its performance is close to that of Prony's projection method. This highlights that the LS approach should be avoided in graphs that have closely spaced eigenvalues. On the other hand, this issue is overcome by Prony's projection method which gives a small RNMSE also for values $K \leq 5$.

iii) As an example, we take the order $K = 16$ to show the difference in performance between FIR graph filters and ARMA graph filters in Fig. 3(c), (d). It is remarkable to highlight that the iterative approach outperforms the FIR by several orders, where the latter has a comparable performance only for $K \leq 3$. Such a finding shows that the ARMA graph filters are more suitable for applications demanding higher approximation accuracies.

iv) We observe a smaller RNMSE for undirected graphs compared to directed graphs. This is because we can do a fitting on the real line instead of in the complex plane. In contrast to undirected graphs, notice that for directed graphs, as shown in Fig. 3(b), all ARMA graph filter design approaches yield a similar performance.

v) As highlighted in Fig. 3(a), an important role is played by the MA order Q (which is generally larger than P). We observe that a higher Q improves the stability of the ARMA filters, specifically for Prony's projection method and the iterative approach where the numerator coefficients are found by minimizing the true error.

vi) If the frequencies are different, the Vandermonde matrix $\mathbf{\Psi}$ is theoretically full rank (invertible) but generally ill-conditioned. Although this issue is encountered for both FIR and ARMA graph filters, ARMA filters improve the condition-

[1] Access to the raw data is through the link: https://donneespubliques.meteofrance.fr/donnees_libres/Hackathon/RADOMEH.tar.gz

[2] We remark that more grid points / nodes, i.e., $N = 300, 1000$, result in similar errors and trends as for $N = 100$.

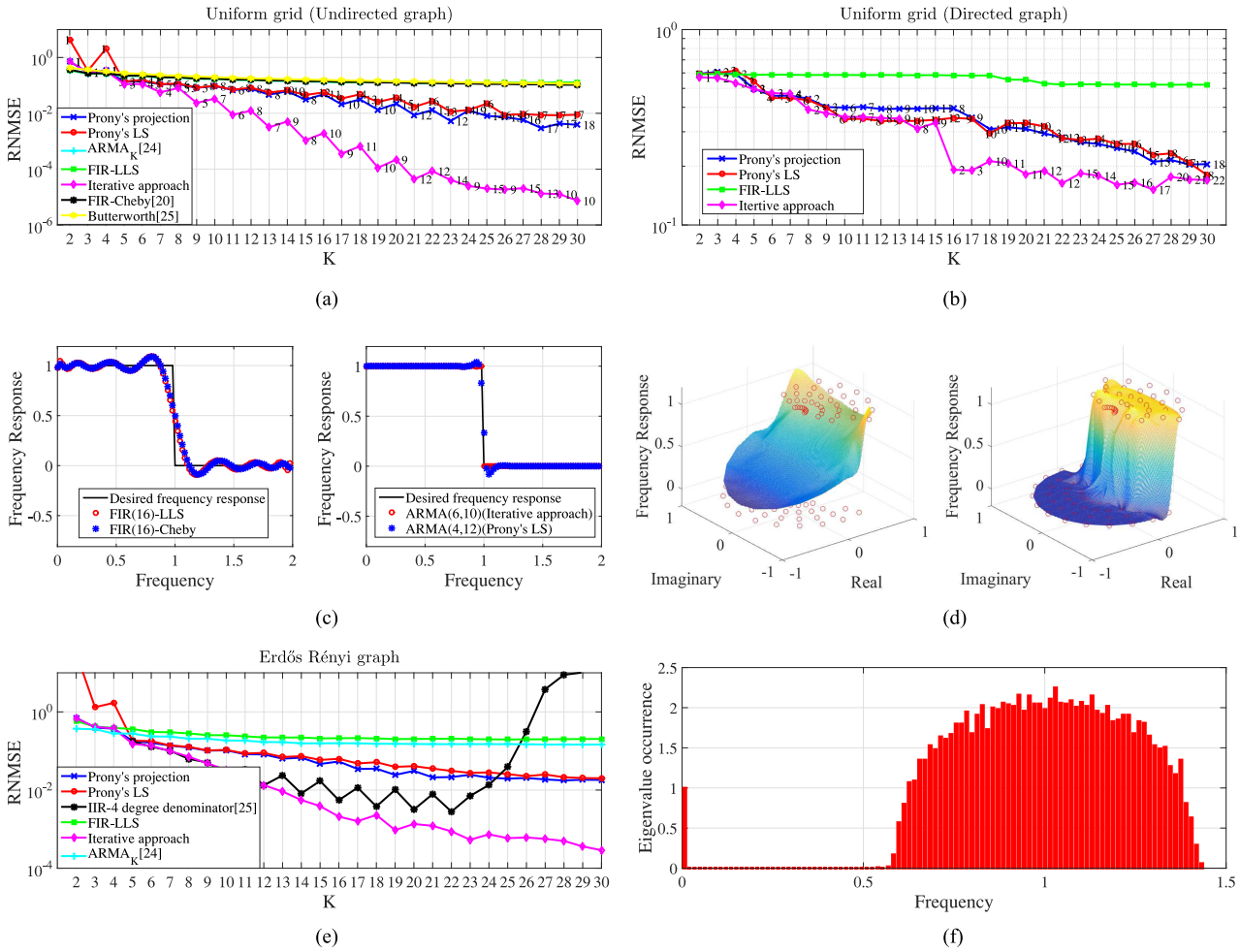


Fig. 3. RNMSE of the proposed design methods for different orders K (such that $P+Q=K$) in approximating an ideal low-pass frequency response. (a) Universal design by gridding the spectrum in $N=100$ ($\mathbf{S}=\mathbf{L}_{\rm n}$) points. For the ARMA filters, the order Q is shown in the plot. (b) Universal design with $N=100$ ($\mathbf{S}=\mathbf{A}_{\rm n}$) points. (c) Comparison of FIR and ARMA with same order $K=16$ for an undirected graph. The graph filters correspond to Fig. 3(a). (d) Comparison of FIR and ARMA with same order $K=16$ for a directed graph. The FIR graph filter (left) and ARMA graph filter (right) correspond to the green and pink lines in Fig. 3(b). The desired frequency response is shown in the plot as red points. (e) Results for the average of 100 Erdős Rényi graphs with $N=100$ nodes and $p=0.1$. (f) Eigenvalue occurrence of 100 Erdős Rényi graph realizations.

ing of the matrix because the filter orders P and Q can be selected much lower than the FIR filter order K. Hence, the solution of our design methods has uniqueness, but there might be a conditioning problem when the orders are increased.

vii) For the universal design (Fig. 3(a)) and ER graph (Fig. 3(e)), we also compare our approach with the methods in [24], [25]. The ARMA$_K$ graph filter [24] has the same order for the nominator and denominator, therefore, we adopt the same value K as the order for both the nominator and denominator. Note that this leads to a total order that is twice the order of our ARMA(P,Q) (recall that $K=P+Q$). For the universal design, we further compare our approach with the universal Butterworth filter [25]. The IIR graph filter [25] is then tested on the ER graph. We follow the scenario of [25] and use a denominator of degree 4, leading to a nominator of degree $(K-4)$. The results show that for low orders ($K<12$), the IIR graph filter [25] has a similar performance to our iterative approach. However, with an increasing order $K>12$, our design method offers a better approximation accuracy.

Iterative approach: We now analyze in more detail the iterative approach to highlight its benefits in improving the ARMA filter accuracy compared to Prony's projection approach. We consider two cases with monotonic convergence, namely, an ARMA$(9,10)$ (characterized by an RNMSE of order 10^{-2} in Prony's projection method, Fig. 3(a)) and an ARMA$(4,9)$ (characterized by an RNMSE of order 10^{-1} in Prony's projection method, Fig. 3(a)) which are considered due to their low orders. For both cases, we initialize the iterations with the solution of Prony's projection method. Note that ARMA$(9,10)$ is the best combination P,Q of order $K=19$, while ARMA$(4,9)$ is not the best combination for order $K=13$. We also consider two filters, the ARMA$(9,11)$ and ARMA$(14,9)$ to illustrate that even without monotonic convergence, the approximation accuracies can be improved with our iterative approach.

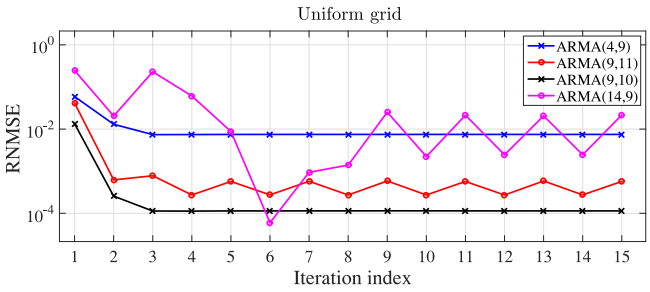


Fig. 4. RNMSE of the iterative approach for the universal design with $N = 100$ points. Performance evaluation for different ARMA filters which are a few particular cases illustrating monotonic convergence, nonmonotonic convergence, and no convergence.

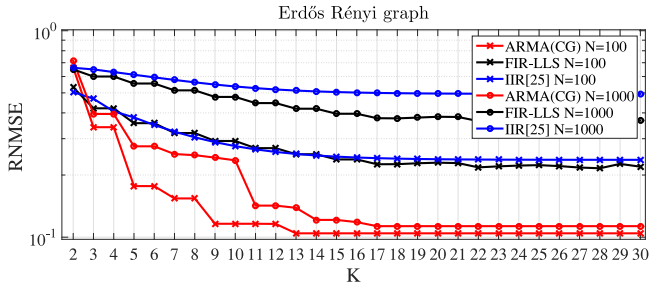


Fig. 5. RNMSE of the ARMA graph filter implementation on an Erdős Rényi graph with $N = 100$ and $N = 1000$. Performance evaluation for the tradeoff between computational cost and approximation accuracy. For CG, the complexity of the ARMA implementation is limited by $PT + Q \leq K$.

In Fig. 4 we show the approximation error as a function of the iteration index and we can immediately notice that for those filters with monotonic convergence, the approximation errors reduce in a few iterations. More specifically, for the ARMA$(9, 10)$ the iterative approach reduces the error from 10^{-2} to 10^{-4}. It is also worth noticing that using the iterative approach, the ARMA$(9, 10)$ outperforms also the ARMA$(11, 17)$, which is the best filter that can be designed with Prony's projection method (within the considered range). Similarly, the iterative approach improves the approximation accuracy for the low order filter ARMA$(4, 9)$. Indeed, its performance is now comparable with all other ARMAs and FIRs with much greater orders. As we mentioned in the previous section, for the non-converging filters, we pick the best approximation result during the iterative procedure, e.g., the performance in the 6-th iteration of the ARMA$(14, 9)$ filter, which is better than the performance of the ARMA$(9, 10)$ filter.

We remark that the above results concern the approximation accuracy of the filter irrespective of their implementation costs. In the sequel, we address some implementation aspects.

CG implementation performance: We now aim at analyzing the ARMA implementation performance using the CG approach w.r.t. its implementation cost. We implement the universally designed ARMA filter using CG on the ER graph with link probability $p = 0.1$ and consider two different sizes: $N = 100$ and $N = 1000$. We again use the universally designed FIR and IIR graph filters as benchmarks. The ARMA filter coefficients are designed universally using the iterative approach with 100 grid points, whereas the FIR filter is designed using LLS also with 100 grid points and the IIR filter following the Butterworth approach [25]. The filter is applied to a white input and the desired frequency response (low pass filter) is compared to the division of the filter output and the input in the frequency domain. In Fig. 5, we show the performance of the ARMA filter (Algorithm 1) when the CG is halted after T iterations such that $PT + Q \leq K$ holds, i.e., the ARMA filter has a smaller or the same implementation cost compared to the FIR filter. For the CG, we set $\varepsilon = 10^{-3}$. The IIR filter has the same order K as the FIR filter and is given a maximum number of iterations of $T = 30$. The results show that the ARMA filter has a lower approximation error than other alternatives with a similar or smaller complexity. Since we here compare the filters for a similar implementation complexity, the RNMSE gap is smaller compared to the previous scenario in Fig. 3(a) and (e). To highlight the benefits of the universal design approach, we consider the ER graph with two different sizes. In Fig. 5, we notice that when increasing the filter order ($K > 16$), the performance of the ARMA graph filter for different size graphs becomes similar. Even for the case with $N = 1000$, the universal design based on 100 grid points is a wise choice and yields a good performance.

Although the aforementioned results are obtained using synthetic data, they highlight the potential of ARMA filters to improve the performance w.r.t. FIR graph filters. The above results can be useful in practice for spectral clustering, building graph filter banks, or designing graph wavelets, where we propose the use of ARMA filters instead of FIR filters.

As we will see next, this improvement in performance of ARMA filters is also present in real data applications.

B. Graph Signal Interpolation

We now illustrate the performance of ARMA graph filters in interpolating the missing values in the Molene weather data set. The data set contains hourly observations of temperature measurements collected in January 2014 in the region of Brest (France). The undirected graph containing the 32 cities (nodes) is built according to [38], which accounts for the smoothness of the data w.r.t. the graph structure. We consider that a portion of the graph signal is missing and by exploiting the smoothness prior we aim to reconstruct the overall graph signal from noisy measurements.

Experimental set up: Given $\mathbf{x}'$ the observed signal and $\mathbf{x}$ the original graph signal, this interpolation problem is formulated as [39], [40]:

$$\min_{\mathbf{x}} \|\mathbf{T}(\mathbf{x} - \mathbf{x}')\|_2^2 + \omega \mathbf{x}^T \mathbf{L}_n \mathbf{x} \tag{32}$$

where $\mathbf{T}$ is a diagonal matrix with $T_{ii} = 1$ if x_i is known and $T_{ii} = 0$ otherwise; ω is the weight for the prior. The optimal solution of (32) is

$$\tilde{\mathbf{x}} = (\mathbf{T} + \omega \mathbf{L}_n)^{-1} \mathbf{x}', \tag{33}$$

which by considering $\mathbf{P} = \mathbf{T} + \omega \mathbf{L}_n$ is solved through the ARMA graph filter (12). We consider the CG to implement

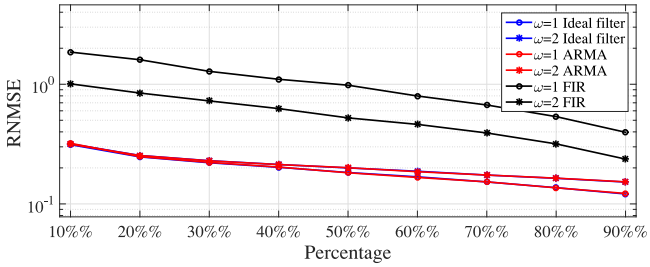


Fig. 6. RNMSE of the ARMA graph filter for interpolation of the Molene data set, where $\omega = 1, 2$. As two comparisons, the ideal graph filter and FIR filter with order $K = 20$ are shown with the same values of ω.

(33) where ε is set to 10^{-2} and the maximum number of iterations T to 20. As a comparison, for the FIR graph filter, the coefficients are found as the solution of

$$\min_{g_k} \left\| (\mathbf{T} + \omega \mathbf{L}_n)^{-1} - \sum_{k=0}^{K} g_k \mathbf{L}_n^k \right\|_F^2 \tag{34}$$

where the g_k values represent the FIR coefficients.

Results: In Fig. 6, we show the RNMSE between the reconstructed signal $\tilde{\mathbf{x}}$ and the original one $\mathbf{x}$ as a function of the portion of missing data. Additionally, to construct the observed signal $\mathbf{x}'$, we add a zero-mean Gaussian noise with variance $\sigma^2 = 10^{-2}$ to the original signal $\mathbf{x}$ and randomly wipe off signals up to the specific percentage. The performance is averaged over all 744 observations. We plot the numerical RNMSE for different percentages and two ω values. These results show that the RNMSE reduces for the ARMA graph filter when the percentage of the known values increases. As a comparison, we notice that the ARMA graph filter offers a similar performance to the ideal graph filter. The FIR graph filter ($K = 20$) yields a worse result in this case.

C. Data Compression With Graph Filters

Our goal, in this subsection, is to show that ARMA filters of low orders can be used to represent the data and perform compression.

Experimental set up: We consider fitting a small order ARMA graph filter to each data realization and then store the filter coefficients instead of the actual data. We now create the graph as a directed 6-nearest neighbor connection. In the directed graph, each vertex is connected to its six closest nodes by means of directed edges [4]. The weight of the edge between v_m and v_n is given as

$$[\mathbf{A}]_{n,m} = \frac{e^{-d_{n,m}^2}}{\sqrt{\sum_{k \in \mathcal{N}_n} e^{-d_{n,k}^2} \sum_{l \in \mathcal{N}_m} e^{-d_{m,l}^2}}} \tag{35}$$

where $d_{n,m}$ represents the geometric distance between nodes v_n and v_m and $\mathcal{N}_n, \mathcal{N}_m$ represent the sets of neighbors of node v_n and v_m. Note that the resulting matrix $\mathbf{A}$ is normal, i.e., $\|\mathbf{A}\| = 1$. For every data realization $\mathbf{x}$, we take the GFT to have $\hat{\mathbf{x}}$ and fit it to an ARMA(P, Q) graph filter. The filter coefficients are derived using the iterative approach with the initial condition given by Prony's projection method. We measure the compression performance as the RNMSE between the compressed signal and the real one $\mathbf{x}$. As a benchmark, we again consider the FIR(K) with $K = P + Q$.

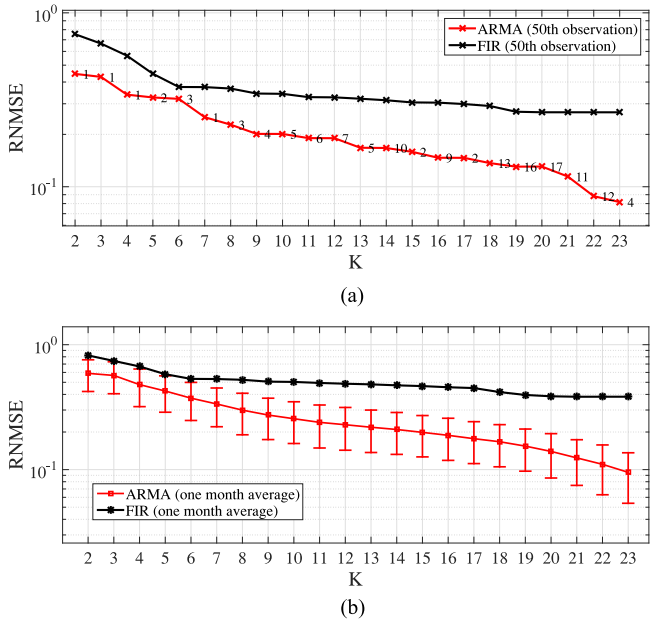


Fig. 7. RNMSE between the data spectrum and the filter frequency response as a function of filter order K. (a) Illustration of the RNMSE of the ARMA graph filter and the same order FIR filter for the 50th observation. The order Q is shown in the plot and $P + Q = K$. (b) Average RNMSE over all 744 temperature realizations (one month) for different filter orders. For the ARMA filter, each error bar shows the standard deviation of the approximation error for order K.

Results: In Fig. 7(a), we show the RNMSE as a function of K for the 50-th observation. We observe that the ARMA filter achieves a smaller RNMSE than the FIR filter even for small orders K. As expected, when K approaches N, we have a smaller error but we also see that the gap in performance between the ARMA and FIR filters increases. This result goes in line with what we obtained in the previous section for synthetic data.

To further quantify the above observations, Fig. 7(b) depicts the average performance over all observations. We still notice that the ARMA graph filters achieve a smaller RNMSE than FIR graph filters and that the RNMSE decreases for higher values of K. With the above approach, a compression ratio of 25% ($K = 23$) is achieved with an RNMSE of 10^{-1}. Note that next to signal compression, the ARMA model can also be used to reconstruct the graph power spectrum of stationarity graph signals from a subset of the nodes [41].

Remark 3: To achieve further compression one can exploit also the stationarity of the signal over time. Thus, instead of fitting a graph filter to each individual observation, one approach may consider fitting a joint graph-temporal filter [42], [43] to the time-varying data.

D. Linear Prediction With ARMA Filters

Inspired by [3], we also test linear prediction (LP) on graphs using ARMA graph filters. We consider the Molene data set

and again compare the ARMA graph filters with the FIR graph filters [3]. The considered problem contains two parts, namely the forward (prediction) part and the backward (synthesis) part. In the forward filtering, the residual between the graph signal and the filter frequency response is calculated and quantized. Next, the backward filter considers building an approximation of the graph signal from the quantized residual. For the ARMA filters we use a variant of the iterative approach to find the filter coefficients, while for the FIR filter we follow [3]. For the graph shift operator $\mathbf{S}$, we consider both the directed graph created by (35) and the undirected graph [38].

Experimental set up: For the ARMA filter, given the graph signal $\mathbf{x}$, the residual $\mathbf{r}$ related to signal prediction is given by

$$\mathbf{r} = \mathbf{x} - g(\mathbf{S})\mathbf{x} = \mathbf{x} - \left(\sum_{p=0}^{P} a_p \mathbf{S}^p\right)^{-1} \left(\sum_{q=0}^{Q} b_q \mathbf{S}^q\right) \mathbf{x}. \quad (36)$$

Notice that next to the constraint $a_0 = 1$ we had before, it is important to set $b_0 = 0$ in order to avoid the trivial solution. Similar to Prony's method, we can derive also a modified residual as

$$\mathbf{r}' = \left(\sum_{p=0}^{P} a_p \mathbf{S}^p\right) \mathbf{x} - \left(\sum_{q=0}^{Q} b_q \mathbf{S}^q\right) \mathbf{x}. \quad (37)$$

To relate this prediction problem to filter design, we can look at the residual and modified residual in the frequency domain, leading to

$$\hat{\mathbf{r}} = \hat{\mathbf{x}} \circ (\mathbf{1}_N - \text{diag}(\mathbf{\Psi}_{P+1}\mathbf{a})^{-1}\mathbf{\Psi}_{Q+1}\mathbf{b}), \quad (38)$$

and

$$\hat{\mathbf{r}}' = \hat{\mathbf{x}} \circ [\mathbf{1}_N \circ (\mathbf{\Psi}_{P+1}\mathbf{a}) - \mathbf{\Psi}_{Q+1}\mathbf{b}]. \quad (39)$$

Hence, up to the element-wise multiplication with $\hat{\mathbf{x}}$, this residual $\hat{\mathbf{r}}$ and modified residual $\hat{\mathbf{r}}'$ look like the error $\mathbf{e}$ in (15) and modified error in $\mathbf{e}'$ in (16), respectively, with $\hat{\mathbf{h}}$ replaced by the all-one vector $\mathbf{1}_N$. As a result, all previous design methods can still be used. They only need to be adapted with an appropriate weighting (coming from $\hat{\mathbf{x}}$) and with the constraint $b_0 = 0$.

Once the filter coefficients that (approximately) minimize the residual $\mathbf{r}$ are found, this residual is quantized with B bits (resulting in $\mathbf{r}_\text{q}$) and forwarded. Then, by applying the backward filter $\mathbf{H} = (\mathbf{I} - g(\mathbf{S}))^{-1}$ to the residual, the approximated signal $\widetilde{\mathbf{x}} = \mathbf{H}\mathbf{r}_\text{q}$ is constructed at the receiving side.

We consider ARMA graph filters for $K \leq 10$ ($K = P + Q$) and for every order K, the residual $\mathbf{r}$ is quantized with different numbers of bits. From the B bits, we spend one bit on the sign, $b = \lceil \log_2(\max(|\mathbf{r}|_i)) \rceil$ bits on the integer part, and the rest of the $(B - b - 1)$ bits on the decimal fraction.

Results: We quantify the performance in terms of RNMSE between the predicted signal $\tilde{\mathbf{x}}$ and the original one $\mathbf{x}$.

The average approximation error over all 744 realizations is shown in Fig. 8(a) as a function of the number of bits (B) used in the quantization for $K = 3$. We can notice that in a direct comparison with the FIR filters the approximation error of the ARMA graph filters is more than one order of magnitude lower. For both filters, as expected, more quantization bits B lead to a better approximation accuracy. Such findings suggest once again that ARMA filters are more suitable than FIR filters for applications demanding higher approximation accuracies.

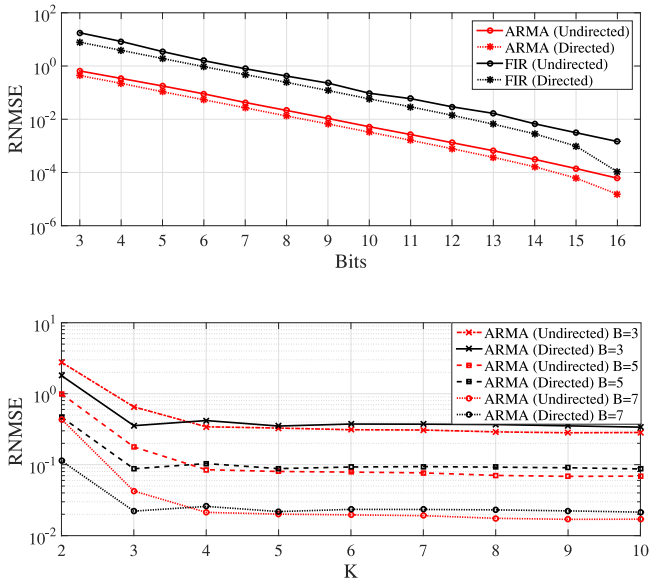


Fig. 8. Average RNMSE of linear prediction on the Molene temperature data set. (a) Average RNMSE of the approximated signal as a function of the number of bits (B) for filter order $K = 3$. (b) Average RNMSE of the estimated signal for different order ARMA filters evaluated for $B = 3, 5, 7$.

To better highlight the performance of the ARMA filters, in Fig. 8(b) we show the RNMSE as a function of the filter order K for different values of B. These results show that the approximation error for $K > 4$ remains constant, similar to what was observed for FIR filters in [3]. This observation suggests that small order filters are preferred for this application. Note that the performance for directed and undirected graphs is almost the same. The directed graph gives the best performance with $K = 3$ while for $K > 3$ the undirected graph gives a lower error. To conclude, we can say that using an ARMA graph filter with $K = 4$ and $B = 7$ (instead of 16 bits) we can reconstruct the data with an error of order 10^{-2} and save 62.5% in transmission costs.

VI. Conclusions

In this work, we have presented ARMA graph filters as well as different methods to perform the filter design on both directed and undirected graphs. The first two filter design approaches are inspired by Prony's method which focus on minimizing some modified errors. The third one iteratively minimizes the original error of the design problem. The iterative approach can be initialized with the solution from one of the previous methods, which suggests that its performance can be improved by the iterative approach. Our theoretical findings are surrogated by numerical results on both synthetic and real data. In a direct comparison with the FIR graph filters, ARMA filters have shown to be more suitable for filter approximation, data interpolation, data compression and linear prediction on graphs.

APPENDIX A

Since we assume that the shift operator $\mathbf{S}$ is real-valued and diagonalizable, the graph frequencies λ_n (eigenvalues) can be grouped into three sets: $1 \leq n \leq M$, $M+1 \leq n \leq 2M$ and $2M+1 \leq n \leq N$. The first and second groups are complex conjugate pairs while the last group consists of the real-valued frequencies. Note that this classification only changes the order of the frequencies, and has no influence on the results of the filter coefficients g_k.

Thus, we can split the Vandermonde matrix $\mathbf{\Psi}_{K+1}$ and write (5) as

$$\begin{aligned}&\min_{\mathbf{g}} \|\hat{\mathbf{h}} - \mathbf{\Psi}_{K+1}\mathbf{g}\|^2 \\ &= \min_{\mathbf{g}} \|[\hat{\mathbf{h}}_1^H, \hat{\mathbf{h}}_2^H, \hat{\mathbf{h}}_3^T]^T - [\mathbf{\Psi}_1^H, \mathbf{\Psi}_2^H, \mathbf{\Psi}_3^T]^T \mathbf{g}\|^2\end{aligned}$$

where the three blocks of matrices and vectors belong to the three different groups.

With $1 \leq n \leq M$, we use the nth and $(M+n)$th frequencies to represent a conjugate pair for the first and second groups of frequencies. Since $\lambda_n = \lambda_{M+n}^*$, the corresponding elements inside the Vandermonde matrix satisfy $[\mathbf{\Psi}_1]_{n,k} = [\mathbf{\Psi}_2]_{M+n,k}^*$, and thus we have $\mathbf{\Psi}_1 = \mathbf{\Psi}_2^*$.

According to Property 2, for the frequency pair $\lambda_n = \lambda_{M+n}^*$, the corresponding desired frequency response satisfies $\hat{h}_n = \hat{h}_{M+n}^*$ and thus, we also have $\hat{\mathbf{h}}_1 = \hat{\mathbf{h}}_2^*$. Meanwhile, for $2M+1 \leq n \leq N$, we have a real-valued $\mathbf{\Psi}_3$ and a real-valued $\hat{\mathbf{h}}_3$ since the corresponding frequencies λ_n inside this range are real-valued.

Now, we can rewrite the solution of (5) as

$$\begin{aligned}\hat{\mathbf{g}} &= \mathbf{\Psi}_{K+1}^{\dagger}\hat{\mathbf{h}} \\ &= (\mathbf{\Psi}_1^H\mathbf{\Psi}_1 + \mathbf{\Psi}_2^H\mathbf{\Psi}_2 + \mathbf{\Psi}_3^T\mathbf{\Psi}_3)^{-1}(\mathbf{\Psi}_1^H\hat{\mathbf{h}}_1 \\ &\quad + \mathbf{\Psi}_2^H\hat{\mathbf{h}}_2 + \mathbf{\Psi}_3^T\hat{\mathbf{h}}_3) \\ &= (\mathbf{\Psi}_1^H\mathbf{\Psi}_1 + \mathbf{\Psi}_1^T\mathbf{\Psi}_1^* + \mathbf{\Psi}_3^T\mathbf{\Psi}_3)^{-1}(\mathbf{\Psi}_1^H\hat{\mathbf{h}}_1 \\ &\quad + \mathbf{\Psi}_1^T\hat{\mathbf{h}}_1^* + \mathbf{\Psi}_3^T\hat{\mathbf{h}}_3).\end{aligned}$$

It is obvious that $\mathbf{\Psi}_1^H\mathbf{\Psi}_1 + \mathbf{\Psi}_1^T\mathbf{\Psi}_1^*$ and $\mathbf{\Psi}_1^H\hat{\mathbf{h}}_1 + \mathbf{\Psi}_1^T\hat{\mathbf{h}}_1^*$ are real-valued. Hence, solving (5) leads to a real-valued solution.

APPENDIX B

The error of the iterative approach on $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is given by

$$\mathbf{e}^{(i+1)} = \boldsymbol{\gamma}^{(i)} \circ (\hat{\mathbf{h}} \circ \boldsymbol{\alpha}) - \boldsymbol{\beta} \circ \boldsymbol{\gamma}^{(i)} \tag{40}$$

By extending $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we can rewrite (40) as

$$\mathbf{e}^{(i+1)} = \boldsymbol{\gamma}^{(i)} \circ \hat{\mathbf{h}} \circ (\mathbf{\Psi}_{P+1}\mathbf{a}) - (\mathbf{\Psi}_{Q+1}\mathbf{b}) \circ \boldsymbol{\gamma}^{(i)} \tag{41}$$

The first term in the right hand side of (41) can be expressed as

$$\begin{aligned}&\boldsymbol{\gamma}^{(i)} \circ \hat{\mathbf{h}} \circ (\mathbf{\Psi}_{P+1}\mathbf{a}) = \boldsymbol{\gamma}^{(i)} \circ [\hat{\mathbf{h}} \circ (\mathbf{\Psi}_{P+1}\mathbf{a})] \\ &= \boldsymbol{\gamma}^{(i)} \circ \left\{[\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a}\right\} \\ &= [(\boldsymbol{\gamma}^{(i)}\mathbf{1}_{P+1}^{\mathrm{T}}) \circ \mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a}.\end{aligned} \tag{42}$$

Similarly, the second term in the right hand side of (41) is rewritten as

$$(\mathbf{\Psi}_{Q+1}\mathbf{b}) \circ \boldsymbol{\gamma}^{(i)} = [(\boldsymbol{\gamma}^{(i)}\mathbf{1}_{Q+1}^{\mathrm{T}}) \circ \mathbf{\Psi}_{Q+1}]\mathbf{b}. \tag{43}$$

Finally, we define (42) and (43) as $\mathbf{H}^{(i)}\mathbf{a}$ and $\mathbf{B}^{(i)}\mathbf{b}$, respectively. This trivially leads to (29).

REFERENCES

[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[2] A. Sandryhaila and J. M. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, Sep. 2014.

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

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

[5] F. R. Chung, *Spectral Graph Theory*, no. 92. Providence, RI, USA: American Mathematical Soc., 1997.

[6] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2014, pp. 872–876.

[7] S. Deutsch, A. Ortega, and G. Medioni, "Manifold denoising based on spectral graph wavelets," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 4673–4677.

[8] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 2, no. 2, pp. 137–148, Jun. 2016.

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

[10] A. Sandryhaila and J. M. Moura, "Classification via regularization on graphs," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2013, pp. 495–498.

[11] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3775–3789, Jul. 2016.

[12] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovačević, "Signal recovery on graphs: Variation minimization," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4609–4624, Sep. 2015.

[13] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst, "Compressive spectral clustering," in *Proc. 33rd Int. Conf. Mach. Learn.*, Jun. 2016, pp. 20–22.

[14] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2786–2799, Jun. 2012.

[15] D. B. Tay and Z. Lin, "Design of near orthogonal graph filter banks," *IEEE Signal Process. Lett.*, vol. 22, no. 6, pp. 701–704, Jun. 2015.

[16] S. K. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4673–4685, Oct. 2013.

[17] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 129–150, 2011.

[18] A. Sakiyama, K. Watanabe, and Y. Tanaka, "Spectral graph wavelets and filter banks with low approximation error," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 2, no. 3, pp. 230–245, Sep. 2016.

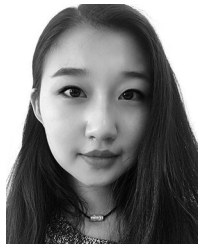
[19] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4223–4235, Aug. 2015.

[20] D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard, "Distributed signal processing via Chebyshev polynomial approximation," *IEEE Trans. Signal Inf. Process. Over Netw.*, to be published.

[21] S. Segarra, A. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, 2017.

[22] M. Coutino, E. Isufi, and G. Leus, "Distributed edge-variant graph filters," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adapt. Process.*, 2017, pp. 1080–1084.

[23] L. Goldsberry, W. Huang, N. F. Wymbs, S. T. Grafton, D. S. Bassett, and A. Ribeiro, "Brain signal analytics from graph signal processing perspective," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 851–855.
[24] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 274–288, Jan. 2017.
[25] 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.
[26] D. P. Bertsekas, *Convex Optimization Theory*. Belmont, MA, USA: Athena Scientific, 2009.
[27] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Carnegie Mellon Univ., Pittsburgh, PA, 1994.
[28] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*. Hoboken, NJ, USA: Wiley, 2009.
[29] C. H. Edwards, D. E. Penney, and D. T. Calvis, *Differential Equations and Boundary Value Problems*. Berlin, Germany: Springer, 2004.
[30] V. Sinswat and F. Fallside, "Eigenvalue/eigenvector assignment by state-feedback," *Int. J. Control*, vol. 26, no. 3, pp. 389–403, 1977.
[31] P. Erdos and A. Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.
[32] J. H. Wilkinson and J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, vol. 87. Oxford, U.K.: Clarendon, 1965.
[33] M. E. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.
[34] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *Proc. IEEE Int. Conf. Distrib. Comput. Sens. Syst. Workshops*, 2011, pp. 1–8.
[35] C.-I. Chang, "Orthogonal subspace projection (OSP) revisited: A comprehensive study and analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 502–518, Mar. 2005.
[36] Y. Hu and G. Leus, "On a unified framework for linear nuisance parameters," *EURASIP J. Adv. Signal Process.*, vol. 2017, no. 1, pp. 4–17, 2017.
[37] N. Perraudin *et al.*, "Gspbox: A toolbox for signal processing on graphs," arXiv:1408.5781.
[38] S. P. Chepuri, S. Liu, G. Leus, and A. O. Hero, "Learning sparse graphs under smoothness prior," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 6508–6512.
[39] S. K. Narang, A. Gadde, and A. Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. IEEE Acoust. Speech Signal Process.*, 2013, pp. 5445–5449.
[40] Y. Mao, G. Cheung, and Y. Ji, "Image interpolation for DIBR viewsynthesis using graph fourier transform," in *Proc. IEEE 3DTV-Conf.: True Vis.-Capture Transmiss. Disp. 3D Video*, 2014, pp. 1–4.
[41] S. P. Chepuri and G. Leus, "Graph sampling for covariance estimation," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 3, no. 3, pp. 451–466, Sep. 2017.
[42] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Separable autoregressive moving average graph-temporal filters," in *Proc. IEEE 24th Eur. Signal Process. Conf.*, 2016, pp. 200–204.
[43] E. Isufi, G. Leus, and P. Banelli, "2-dimensional finite impulse response graph-temporal filters," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2016, pp. 405–409.



Jiani Liu was born in Xi'an, China, in 1991. She received the M.Sc. degree in electrical engineering from the Northwestern Polytechnical University, Xi'an, China, in 2016. Since October 2015, she is working toward the Ph.D. degree with the Circuits and Systems Group, Department of Microelectronics, Delft University of Technology, Delft, The Netherlands.

Her research interests include signal processing on graphs and underwater acoustic.

Elvin Isufi was born in Albania, in 1989. He received the M.Sc. degree (cum laude) in electrical and telecommunication engineering from the University of Perugia, Perugia, Italy, in 2014. Since November 2014, he is 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 at Circuits and Systems Group, Delft University of Technology, where he worked on his Master Thesis. His research interests include graph signal processing, graph and time signal processing, network anomaly detection, and network coding for underwater routing.

Geert Leus received the M.Sc. and Ph.D. degrees in electrical engineering from the KU Leuven, Leuven, Belgium, in June 1996 and May 2000, respectively.

He is currently an "Antoni van Leeuwenhoek" Full Professor with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, The Netherlands. His research interests include the broad area of signal processing, with a specific focus on wireless communications, array processing, sensor networks, and graph signal processing.

Dr. Leus was the recepient of the 2002 IEEE Signal Processing Society Young Author Best Paper Award and the 2005 IEEE Signal Processing Society Best Paper Award. He was a Member-at-Large of the Board of Governors of the IEEE Signal Processing Society, the Chair of the IEEE Signal Processing for Communications and Networking Technical Committee, a Member of the IEEE Sensor Array and Multichannel Technical Committee, and the Editor-in-Chief of the *EURASIP Journal on Advances in Signal Processing*. He was also on the Editorial Boards 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*. Currently, he is the Vice-Chair of the EURASIP Special Area Team on Signal Processing for Multisensor Systems, an Associate Editor of Foundations and Trends in Signal Processing, and the Editor-in-Chief of EURASIP Signal Processing. He is a Fellow of EURASIP.