Autoregressive Moving Average Graph Filter Design

Jiani Liu, Elvin Isufi, and Geert Leus

***Abstract*—In graph signal processing, signals are processed by explicitly taking into account their underlying structure, which is generally characterized by a graph. In this field, graph filters play a major role to process such signals in the so-called graph frequency domain. In this paper, we focus on the design of autoregressive moving average (ARMA) graph filters and basically present two design approaches. The first approach is inspired by Prony's method, which considers a modified error between the modeled and the desired frequency response. The second approach is based on an iterative method, 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 design algorithms is evaluated and compared with finite impulse response (FIR) graph filters. The obtained results show that ARMA filters outperform FIR filters in terms of approximation accuracy even for the same computational cost.**

I. Introduction

In today's society, signals with irregular structure are abundant, e.g., data from social networks, brain signals, traffic information, and so on. Graph signal processing (GSP) is the tool to handle such signals, and it basically extends classical digital signal processing to signals that live on the vertices of irregular graphs [1], [2]. As for temporal signals, a Fourier-like transform for graph signals is defined, which decomposes a graph signal into the different harmonic modes of the graph [3] and which allows to analyze and process a graph signal in the so-called graph frequency domain [4]. Together with the graph Fourier transform (GFT), graph filters yield a key tool to process the graph spectrum, i.e., to amplify or attenuate different graph frequencies. Relevant applications of graph filters are graph signal denoising [5], smoothing [6], signal classification [7], signal recovery [8] and graph clustering [9].

Since graph filtering directly in the frequency domain is computationally expensive, finite impulse response (FIR) graph filters have been devised to implement this operation directly in the vertex domain. Such FIR graph filters are expressed as a polynomial in a so-called graph shift operator, e.g., the adjacency matrix [3], the discrete graph Laplacian matrix [1], or any modification of those matrices. Popular FIR graph filter design approaches are based on the linear least squares (LLS) method [4] and Chebyshev polynomials [10]. However, to accurately match a specified graph frequency response, FIR filters require a high filter order leading to a high implementation cost.

As an alternative to FIR graph filters, infinite impulse response (IIR) or autoregressive moving average (ARMA) graph filters have been proposed [11], [12]. These filters are characterized by a rational frequency response, which brings more degrees of freedom to the design. Existing works on ARMA graph filters mainly focus on a distributed implementation, which only leads to the modelled frequency response after an infinite number of iterations [11], [12] and limits the design space due to the required convergence constraints. To fully exploit the benefits of the rational frequency response, in this paper, we focus on a centralized ARMA filter implementation and hence the filter design is not hampered by the convergence issues of a distributed implementation. In a centralized fashion, the ARMA filter output can be simply computed by solving a linear system of equations, e.g., by using first order methods [13], or conjugate gradient (CG) [14].

Under the proposed centralized filter implementation, we propose in this paper two ARMA filter design strategies. The first approach is inspired by Prony's method [15], where a modified error between the modeled and the desired frequency response is minimized. The second approach, on the other hand, minimizes the true error iteratively following the Steigliz-McBride idea [15]. As initial condition, we could use the solution from the first approach, thereby potentially improving the approximation accuracy of that solution.

Numerical tests validate our findings. We show that the ARMA filters outperform FIR filters in terms of approximation accuracy not only for the same number of filter coefficients, but also for the same filter implementation cost.

II. Preliminaries

In this section we recall some basic concepts of signal processing on graphs. We start with the graph Fourier transform (GFT) and its relation to the graph shift operator. Then, we review FIR graph filters and present their design challenges in approximating a desired frequency response.

A. Graph Fourier Transform (GFT)

Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ the set of N nodes or vertices and $\mathcal{E}$ the set of edges between the nodes. The local structure of the graph is captured by the adjacency matrix $\mathbf{A}$ or the discrete graph Laplacian $\mathbf{L}_{\text{d}} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the degree matrix. Both $\mathbf{A}$ and $\mathbf{L}_{\text{d}}$ are candidates for the so-called graph shift operator $\mathbf{S}$, an operator that forms the basis for processing graph signals, as we will see later on. Other candidates for $\mathbf{S}$ can be the normalized Laplacian $\mathbf{L}_{\text{n}}{=}\mathbf{D}^{-1/2}\mathbf{L}_{\text{d}}\mathbf{D}^{-1/2}$, or any of their modifications like the translated normalized Laplacian $\mathbf{L}_{\text{n}} - \mathbf{I}$.

We will indicate with the vector $\mathbf{x}$ the graph signal, i.e., a signal living on the graph $\mathcal{G}$, where each value x_i is associated to the node v_i. Being a symmetric matrix, $\mathbf{S}$ enjoys an eigenvalue decomposition $\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\text{T}}$, with $\mathbf{U}$ the eigenvector matrix containing as columns the so-called graph modes $\mathbf{u}_1$ up to $\mathbf{u}_N$ and $\mathbf{\Lambda}$ a diagonal matrix containing as diagonal entries

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands. E-mails: {j.liu-1, e.isufi-1, g.j.t.leus}@tudelft.nl.

the so-called graph frequencies λ_1 up to λ_N. To obtain the graph frequency representation of $\mathbf{x}$, the eigenvector matrix $\mathbf{U}$ is used to transform the signal into the Fourier domain. Specifically, the GFT $\hat{\mathbf{x}}$ of $\mathbf{x}$ and its inverse are, respectively, $\hat{\mathbf{x}} = \mathbf{U}^{\mathrm{T}}\mathbf{x}$ and $\mathbf{x} = \mathbf{U}\hat{\mathbf{x}}$. For a deeper analysis of the GFT and graph signal processing in general, we refer the interested reader to [1] and [3].

B. FIR Graph Filters

Filtering a graph signal $\mathbf{x}$ can be carried out in the frequency domain as

$$\hat{\mathbf{y}} = h(\mathbf{\Lambda})\hat{\mathbf{x}},$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ represent the GFT of the input signal $\mathbf{x}$ and output signal $\mathbf{y}$, respectively, and $h(\mathbf{\Lambda})$ is a diagonal matrix containing the frequency response on its diagonal. In the vertex domain, this operation can be represented by

$$\mathbf{y} = \mathbf{H}\mathbf{x},$$

where $\mathbf{H} = \mathbf{U}h(\mathbf{\Lambda})\mathbf{U}^{\mathrm{T}}$. Hence, in the vertex domain, a graph filter is a linear operator that is diagonalizable by the graph Fourier transformation matrix $\mathbf{U}$. However, to carry out the graph filtering operation as above, an eigenvalue decomposition of the graph shift operator $\mathbf{S}$ is required, which complicates the filtering operation.

To alleviate this burden, the FIR graph filter has been proposed as a way to directly implement a graph filter in the vertex domain. An FIR graph filter $\mathbf{G}$ of order K can be expressed as a K-th order polynomial in the graph shift operator

$$\mathbf{G} = g(\mathbf{S}) = \textstyle\sum_{k=0}^{K} g_k \mathbf{S}^k, \tag{1}$$

with g_k the filter coefficients and $g(.)$ a K-th order polynomial function. Using the recursion $\mathbf{S}^k\mathbf{x} = \mathbf{S}(\mathbf{S}^{k-1}\mathbf{x})$, the implementation cost of the FIR graph filter is of order $O(KE)$, with $E = |\mathcal{E}|$ the number of edges. Clearly, the linear operator $\mathbf{G}$ is diagonalizable by $\mathbf{U}$ since $\mathbf{G} = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^{\mathrm{T}}$ and as such it is a valid graph filter. The n-th diagonal entry of $g(\mathbf{\Lambda})$ is the graph frequency response at eigenvalue λ_n and it is denoted in short as $\hat{g}_n$. As a result, the relation between the graph frequency reponse $\hat{g}_n$ and the filter coefficients g_k is given by

$$\hat{g}_n = \textstyle\sum_{k=0}^{K} g_k \lambda_n^k. \tag{2}$$

Assuming now that the desired frequency response at frequency λ_n is given by $\hat{h}_n$, we want to design g_n such that

$$\hat{h}_n \approx \hat{g}_n = \textstyle\sum_{k=0}^{K} g_k \lambda_n^k, \tag{3}$$

for every graph frequency λ_n. When the graph frequencies are known, the above problem can be solved using the LLS method, under the condition that all eigenvalues are different [4], [16]. However, since estimating the graph frequencies again entails some additional complexity, FIR filters are often designed without any explicit knowledge of the graph frequencies and given a desired frequency response $\hat{h}(\lambda)$ in a continuous range of frequencies $[\lambda_{\min}, \lambda_{\max}]$. Such a problem, named universal design, can again be solved using the LLS method by discretizing $[\lambda_{\min}, \lambda_{\max}]$ into a finite set of graph frequencies. Alternatively, we can fit Chebyshev polynomials to $\hat{h}(\lambda)$ leading to a closed-form expression for the coefficients g_k [10].

Algorithm1: Conjugate gradient

1	**Input**:	$\mathbf{y}^{(0)}$, $\mathbf{x}$, coefficients $\mathbf{a}$, $\mathbf{b}$
2		accuracy evaluation ε, maximum iteration T
3	**Compute**:	$\mathbf{x}_Q$, $\mathbf{P}\mathbf{y}^{(0)}$ (computed as $\mathbf{S}^k\mathbf{y}^{(0)} = \mathbf{S}(\mathbf{S}^{k-1}\mathbf{y}^{(0)})$)
4		$\mathbf{d}^{(0)} = \mathbf{r}^{(0)} = \mathbf{x}_Q - \mathbf{P}\mathbf{y}^{(0)}$, $\delta^{(0)} = \delta^{new} = \mathbf{r}^{(0)T}\mathbf{r}^{(0)}$
5	**Iteration**:	while $i < T$ and $\delta^{new} > \varepsilon^2\delta^{(0)}$
6		$\omega^{(i)} = \frac{\delta_{new}}{\mathbf{d}^{(i)T}\mathbf{P}\mathbf{d}^{(i)}}$
7		$\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \omega^{(i)}\mathbf{d}^{(i)}$, $\mathbf{r}^{(i+1)} = \mathbf{r}^{(i)} - \omega^{(i)}\mathbf{P}\mathbf{d}^{(i)}$
8		$\delta^{old} = \delta^{new}$, $\delta^{new} = \mathbf{r}^{(i+1)T}\mathbf{r}^{(i+1)}$
9		$\varphi^{(i+1)} = \frac{\delta^{new}}{\delta^{old}}$, $\mathbf{d}^{(i+1)} = \mathbf{r}^{(i+1)} + \varphi^{(i+1)}\mathbf{d}^{(i)}$
10		$i = i + 1$
11	**Return**:	$\mathbf{y}^{(i+1)}$

III. ARMA Graph Filters

To improve the approximation accuracy compared to the FIR graph filter, we now consider applying an ARMA filter to the signal $\mathbf{x}$. For an ARMA graph filter, the graph frequency response at frequency λ_n, can be written as [12]

$$\hat{g}_n = \frac{\sum_{q=0}^{Q} b_q \lambda_n^q}{1 + \sum_{p=1}^{P} a_p \lambda_n^p}, \tag{4}$$

where a_p are the coefficients of the autoregressive part and b_q are the coefficients of the moving average part. Stability is guaranteed if

$$1 + \textstyle\sum_{p=1}^{P} a_p \lambda_n^p \neq 0, \quad \forall\, n. \tag{5}$$

A. ARMA Implementation

Although many distributed ARMA graph filter implementations have been proposed [11], [12], they are all iterative in nature and require specific convergence constraints to be satisfied. In this paper, we want to fully exploit the power of the ARMA filter and avoid any additional design constraints. That is why we focus here on a centralized ARMA implementation.

Given the graph signal $\mathbf{x}$ and a set of (stable) filter coefficients a_p and b_q, the filter output $\mathbf{y}$ can be obtained in the frequency domain as

$$\hat{y}_n = \frac{\sum_{q=0}^{Q} b_q \lambda_n^q}{1 + \sum_{p=1}^{P} a_p \lambda_n^p} \hat{x}_n. \tag{6}$$

Bringing the denominator to the other side, grouping the equations $\forall\, n$, and applying the inverse GFT, we obtain the matrix-vector form

$$(\mathbf{I} + \textstyle\sum_{p=1}^{P} a_p \mathbf{S}^p)\mathbf{y} = (\sum_{q=0}^{Q} b_q \mathbf{S}^q)\mathbf{x}. \tag{7}$$

By defining the matrices $\mathbf{P} = \mathbf{I} + \sum_{p=1}^{P} a_p \mathbf{S}^p$ and $\mathbf{Q} = \sum_{q=0}^{Q} b_q \mathbf{S}^q$, we can express (7) in the compact form $\mathbf{P}\mathbf{y} = \mathbf{Q}\mathbf{x}$. Based on this expression, we can compute $\mathbf{y}$ by calculating the right hand-side denoted as $\mathbf{x}_Q = \mathbf{Q}\mathbf{x}$ (which corresponds to pre-filtering $\mathbf{x}$ with an FIR graph filter of order Q) and by solving the linear system $\mathbf{P}\mathbf{y} = \mathbf{x}_Q$. The latter can be done using techniques like first order methods [13] or conjugate gradient (CG) [14], which can be efficiently

implemented, especially when $\mathbf{S}$ is sparse, i.e., when the graph is sparse [17]. In this work, we consider the CG method to implement the ARMA graph filter in the vertex domain. With reference to Algorithm 1, the CG approach has a computational complexity that scales linearly with the number of edges E, as was the case for the FIR filter. If we assume that the CG is arrested after T iterations, the overall implementation cost of the ARMA graph filter is of order $O((PT+Q)E)$.

B. ARMA Design Problem

In the filter design phase, we would like to find the ARMA coefficients a_p and b_q such that a desired frequency response $\hat{h}_n$ is matched. The latter can be either a low-pass frequency response for clustering or bandlimiting graph signals, a heat kernel-like response which is encountered in Tikhonov denoising and interpolation, or a Wiener-like frequency response useful for processing stationary graph signals. Thus, given the desired frequency response $\hat{h}_n$, we would like to find the filter coefficients a_p and b_q that best approximate $\hat{h}_n$. Specifically, we would like to reduce the error

$$e_n = \hat{h}_n - \frac{\sum_{q=0}^{Q} b_q \lambda_n^q}{1+\sum_{p=1}^{P} a_p \lambda_n^p}, \tag{8}$$

for each frequency λ_n. Note that when solving this problem, we do not explicitly take the stability constraint (5) into account since most unconstrained solutions satisfy this constraint directly.

As for the FIR filter, a universal design is generally considered without explicit knowledge of the graph frequencies and given a desired frequency response $\hat{h}(\lambda)$ in a continuous range of frequencies $[\lambda_{\min}, \lambda_{\max}]$. In that case, the interval $[\lambda_{\min}, \lambda_{\max}]$ is simply discretized into a finite set of graph frequencies.

IV. ARMA Design Methods

This section contains our proposed ARMA graph filter design methods, starting with a method inspired by Prony's method, which is subsequently improved with an iterative approach.

A. Prony's Method

In Prony's method, instead of focussing on the error e_n, a modified error e'_n is considered, where

$$e'_n = \hat{h}_n \left(1+\sum_{p=1}^{P} a_p \lambda_n^p\right) - \sum_{q=0}^{Q} b_q \lambda_n^q. \tag{9}$$

This modified error is linear in the coefficients a_p and b_q and hence the LLS method can be used to reduce the modified error. This is explained in more detail next.

Stacking e'_n in the vector $\mathbf{e}' = [e'_1, \ldots, e'_N]^{\mathrm{T}}$ and $\hat{h}_n$ in the vector $\hat{\mathbf{h}} = [\hat{h}_1, \ldots, \hat{h}_N]^{\mathrm{T}}$ as well as defining the filter coefficient vectors $\mathbf{a} = [1, a_1, \ldots, a_P]^{\mathrm{T}}$ and $\mathbf{b} = [b_0, b_1, \ldots, b_Q]^{\mathrm{T}}$, (9) can be expressed as

$$\mathbf{e}' = \hat{\mathbf{h}} \circ (\mathbf{\Psi}_{P+1}\mathbf{a}) - \mathbf{\Psi}_{Q+1}\mathbf{b} \tag{10}$$

$$= [\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})]\mathbf{a} - \mathbf{\Psi}_{Q+1}\mathbf{b}, \tag{11}$$

where $\mathbf{\Psi}_{K+1}$ is the $N \times (K+1)$ Vandermonde matrix with entries $[\mathbf{\Psi}_{K+1}]_{n,k+1} = \lambda_n^k$, $\mathbf{1}_{P+1}$ is a $(P+1)\times 1$ all one vector, and $\circ$ represents the element-wise Hadamard product.

Minimizing the error in (10) can now be solved efficiently using the LLS method, which boils down to

$$\min_{\mathbf{a},\mathbf{b}} \|[\mathbf{\Psi}_{P+1} \circ (\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}}), -\mathbf{\Psi}_{Q+1}] \begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix} \|^2, \quad \text{s.t.} \quad a_0 = 1. \tag{12}$$

Note that instead of jointly solving (12) for $\mathbf{a}$ and $\mathbf{b}$, we could as well only solve it for $\mathbf{a}$ by projecting out $\mathbf{b}$ using the orthogonal projection matrix $\mathbf{P}^{\perp}_{\mathbf{\Psi}_{Q+1}} = \mathbf{I}_N - \mathbf{\Psi}_{Q+1}\mathbf{\Psi}^{\dagger}_{Q+1}$, where $\mathbf{A}^{\dagger}$ denotes the pseudo-inverse of $\mathbf{A}$. This would lead to the same solution for $\mathbf{a}$, but gives us the opportunity to use different methods to find $\mathbf{b}$, e.g., we could find $\mathbf{b}$ by minimizing the original error (8) instead of the modified error (9) since both errors are linear in $\mathbf{b}$. This last would resemble the well-known Shanks' method for temporal signals. However, since the performance of Shanks' method is close to the one of Prony's method, we will only simulate Prony's method in the simulations section.

B. Iterative Method

Prony's method focusses on minimizing the modified error (9), which is of course not equivalent to the original error (8). To alleviate this issue, in this section, we will devise an iterative method to directly minimize the original error (8).

To ease the notation, let us introduce the notations $\beta_n = \sum_{q=0}^{Q} b_q \lambda_n^q$ and $\alpha_n = 1+\sum_{p=1}^{P} a_p \lambda_n^p$, and rewrite the original error (8) as

$$e_n = \hat{h}_n - \frac{\beta_n}{\alpha_n}. \tag{13}$$

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

$$e_n = \hat{h}_n - \beta_n\gamma_n = [\hat{h}_n\alpha_n - \beta_n]\gamma_n, \tag{14}$$

which is linear in α_n, β_n and γ_n, if each of them is treated as a separate variable. Specifically, note that if γ_n is fixed, e_n becomes linear in α_n and β_n. This will be our starting point to minimize e_n iteratively.

Before we detail the algorithm, let us rewrite (14) in vector form as

$$\mathbf{e} = [\hat{\mathbf{h}} \circ \boldsymbol{\alpha} - \boldsymbol{\beta}] \circ \boldsymbol{\gamma}, \tag{15}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^{\mathrm{T}}$, $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_N]^{\mathrm{T}}$, $\boldsymbol{\gamma} = [\gamma_1, \cdots, \gamma_N]^{\mathrm{T}}$, and $\mathbf{e} = [e_1, \ldots, e_N]^{\mathrm{T}}$. Now, let $\boldsymbol{\alpha}^{(i)}$ and $\boldsymbol{\beta}^{(i)}$ denote the estimates of the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively, in the i-th iteration. A value for $\boldsymbol{\gamma}$ can then be found as an element-wise inversion of $\boldsymbol{\alpha}^{(i)}$, which we will label as $\boldsymbol{\gamma}^{(i)}$. Using this value for $\boldsymbol{\gamma}$ in (15), we obtain the updated error

$$\mathbf{e} = \boldsymbol{\gamma}^{(i)} \circ (\hat{\mathbf{h}} \circ \boldsymbol{\alpha}) - \boldsymbol{\gamma}^{(i)} \circ \boldsymbol{\beta}, \tag{16}$$

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

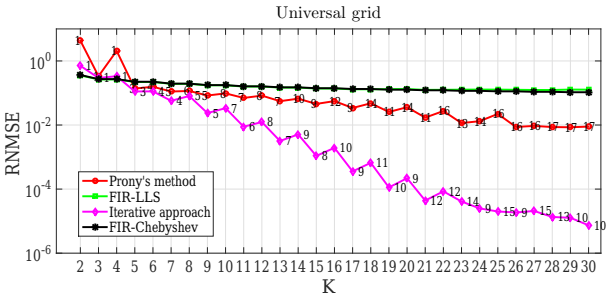

Fig. 1. RNMSE of the different filter design methods for different orders K (such that $P+Q=K$) in approximating an ideal low-pass frequency response. We consider a universal design by gridding the spectrum of $[0,2]$ in $N=100$ grid points. For the ARMA filter, the order Q is shown in the plot.

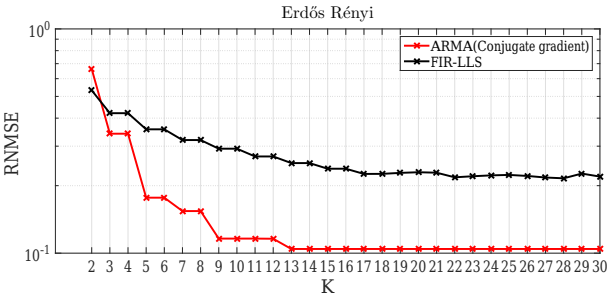

Fig. 2. RNMSE of the ARMA filter implementations on an Erdős Rényi graph with $N=100$, $p=0.1$. The ARMA filter is implemented using CG with a complexity that is smaller than or equal to the FIR filter implementation ($PT+Q\leq K$).

One thing that we ignored in the previous description is that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are directly related to $\mathbf{a}$ and $\mathbf{b}$, which means we need to express (16) as

$$\mathbf{e}=\mathbf{A}^{(i)}\mathbf{a}-\mathbf{B}^{(i)}\mathbf{b}, \tag{17}$$

where $\mathbf{A}^{(i)}=(\boldsymbol{\gamma}^{(i)}\mathbf{1}_{P+1}^{\mathrm{T}})\circ\boldsymbol{\Psi}_{P+1}\circ(\hat{\mathbf{h}}\mathbf{1}_{P+1}^{\mathrm{T}})$ and $\mathbf{B}^{(i)}=(\boldsymbol{\gamma}^{(i)}\mathbf{1}_{Q+1}^{\mathrm{T}})\circ\boldsymbol{\Psi}_{Q+1}$. Using this error, the filter coefficients at the $(i+1)$-th iteration, denoted as $\mathbf{a}^{(i+1)}$ and $\mathbf{b}^{(i+1)}$ are found by solving

$$\min_{\mathbf{a},\mathbf{b}}\|[\mathbf{A}^{(i)},-\mathbf{B}^{(i)}]\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix}\|^2 \quad \text{s.t.} \quad a_0=1. \tag{18}$$

Since the matrix $[\mathbf{A}^{(i)},-\mathbf{B}^{(i)}]$ has size $N\times(P+Q+2)$ a necessary condition for solving (18) is $P+Q+1\leq N$.

Note that for $\boldsymbol{\gamma}^{(0)}=\mathbf{1}$, the iterative approach leads to Prony's method (12), which will often be considered as an initialization of the iterative method.

V. Numerical Results

In this section, we compare our ARMA graph filters with FIR graph filters. In our simulations we make use of GSPBox [18]. We consider two different scenarios. First, we compare how well each graph filter approximates a desired frequency response under a universal setting (the graph frequencies are not known). Second, we compare the filter designs when implemented on an Erdős Rényi graph [19], where we consider the CG algorithm to implement the ARMA graph filter. For both cases we consider $N=100$ and use $\mathbf{S}=\mathbf{L}_{\mathrm{n}}$ as the graph shift operator. The desired frequency response is an ideal low-pass filter with cut-off $\lambda_c=1$. We measure the approximation accuracy with the root normalized mean squared error (RNMSE) between the desired frequency response $\hat{h}_n$ and the designed frequency response $\hat{g}_n$.

Filter design comparison. In this scenario, we assume the graph frequencies are not known and we discretize the interval $[0,2]$ into $N=100$ uniformly spaced grid points. In Fig. 1, we show the RNMSE for Prony's method and the iterative method. Specifically, the depicted RNMSE is related to the best combination of orders (P,Q) for each particular K such that $P+Q=K$. The iterative approach is initialized with the solution of Prony's method, to show its potential in improving the approximation accuracy. Additionally, as a benchmark, a FIR filter of order K is designed with LLS (FIR-LLS) and Chebyshev polynomials (FIR-Chebyshev). For both FIR designs, the RNMSE is higher, except for $K\leq 5$. Further, the FIR approximation accuracy does not improve when K increases. It is remarkable that the iterative approach outperforms the FIR design by several orders, where the latter has a comparable performance only for $K\leq 3$. This shows that ARMA graph filters are more suitable for applications demanding a high approximation accuracy, such as in filter-banks.

Implementation comparison. Here, we implement the universally designed ARMA filter using CG on an Erdős Rényi graph with link probability $p=0.1$ and we again use the universally designed FIR filter as a benchmark. We assume the ARMA filter has been designed using the iterative approach whereas the FIR filter has been designed using LLS. The filter is applied to a white input an the desired frequency response is compared to the division of the filter output and the input in the frequency domain. In Fig. 2, we show the performance of the ARMA filter 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. The results show again that the ARMA filter has a lower approximation error than the FIR filter. Because we here compare these two filters for a similar implementation complexity, the RNMSE gap is smaller as in the first scenario.

VI. Conclusions

In this work, we have presented two ARMA graph filter design approaches. The first approach is inspired by Prony's method which solves a modified error between the modeled and the desired frequency response. The second one minimizes iteratively the original error instead of the modified one and can also be initialized with the solution from the first method. Our theoretical findings are evaluated by numerical results. In a direct comparison with state of the art FIR graph filters, ARMA filters have shown to improve the approximation accuracy of FIR graph filters.

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 Processing Magazine*, vol. 30, no. 3, pp. 83–98, 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 Processing Magazine*, vol. 31, no. 5, pp. 80–90, 2014.

[3] ——, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[4] ——, "Discrete signal processing on graphs: Frequency analysis." *IEEE Trans. Signal Processing*, vol. 62, no. 12, pp. 3042–3054, 2014.

[5] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*. IEEE, 2014, pp. 872–876.

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

[7] A. Sandryhaila and J. M. Moura, "Classification via regularization on graphs." in *GlobalSIP*, 2013, pp. 495–498.

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

[9] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst, "Compressive spectral clustering," in *Machine Learning, Proceedings of the Thirty-third International Conference (ICML 2016), June*, 2016, pp. 20–22.

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

[11] X. Shi, H. Feng, M. Zhai, T. Yang, and B. Hu, "Infinite impulse response graph filters in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 22, no. 8, pp. 1113–1117, 2015.

[12] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 274–288, 2017.

[13] D. P. Bertsekas, *Convex optimization theory*. Athena Scientific Belmont, 2009.

[14] J. R. Shewchuk *et al.*, "An introduction to the conjugate gradient method without the agonizing pain," 1994.

[15] M. H. Hayes, *Statistical digital signal processing and modeling*. John Wiley & Sons, 2009.

[16] S. Segarra, A. G. Marques, and A. Ribeiro, "Distributed linear network operators using graph filters," *arXiv preprint arXiv:1510.03947*, 2015.

[17] M. E. Newman, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[18] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "Gspbox: A toolbox for signal processing on graphs," *arXiv preprint arXiv:1408.5781*, 2014.

[19] 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.