FINITE-PRECISION EFFECTS ON GRAPH FILTERS

Luiz F. O. Chamon and Alejandro Ribeiro

Department of Electrical and Systems Engineering
University of Pennsylvania
e-mail: luizf@seas.upenn.edu, aribeiro@seas.upenn.edu

ABSTRACT

Graph filters play a fundamental role in graph signal processing. In practice, however, the finite precision nature of digital computers introduces numerical errors that can hinder their performance and jeopardize their usefulness. To mitigate these effects, this work investigates the numerical behavior of graph filters in finite-precision arithmetic. It derives a closed-form expression for the variance of the quantization noise at the filter output and shows how the filter coefficients interact with the spectrum of the shift operator to affect the numerical performance of graph filters. Based on these results, the paper then provides an optimally weighted shrinkage regularizer that can be used to design filters robust to quantization errors. Bit-accurate experiments illustrate the performance of different designs and show the importance of considering numerical effects when designing graph filters.

1. INTRODUCTION

Graph signal processing (GSP) sets out to extend traditional signal processing techniques to irregular data structures [1, 2]. The cornerstone of the GSP generalization is the notion of graph shift operator, which is a generic way of denoting matrix representations of a graph—such as adjacency and Laplacian matrices—, that we interpret as a generalization of the time shift. In the same way as linear time invariant filters are defined in the time domain, linear shift invariant filters are defined in GSP as those that are invariant with respect to the application of the graph shift. The design and distributed implementation of polynomial, multirate, and ARMA graph filters is a fundamental problem in GSP which has found application in sensor networks and image processing among other domains [2–8].

As in classical signal processing, these filters are realized by finite-precision machines, such as general purpose processors, GPUs, or FPGAs. Regardless of the specific medium, finite precision arithmetic introduces quantization errors that affect the filters output and can lead to catastrophic results even when using high precision. These potential issues are well-known in classical signal processing [9–11]. To the extent that time-domain filters are particular cases of graph filters, these problems are at least as bad in GSP. In fact, it is not difficult to see that they are actually worse. This is because the effect of graph filters depend on powers of the eigenvalues of the shift operator, which may amplify quantization errors. This is different from the case of time invariant filters where the shift does not inherently alter the values of the signal.

An illustration of the numerical sensitivity of graph filters is shown in Figures 1 and 2. In this example, a "matched" low-pass filter is designed to detect a bandlimited signal (see Section 5 for details). The graph spectrum and filters responses are shown in Fig. 1. Note that the shift operator is well-conditioned ($\kappa < 5$)

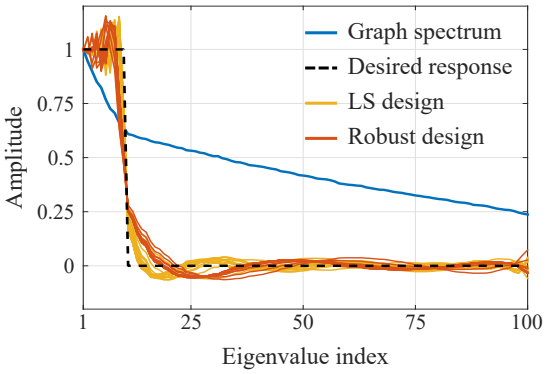

Fig. 1. LDA using graph filters: spectral responses.

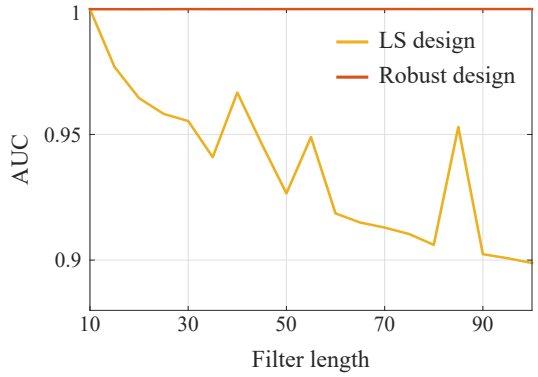

Fig. 2. LDA using graph filter: detection performance in single precision (32 bits floating-point).

and that the filters designed using simple least squares (LS) and our robust method match the desired graph frequency response equally well. In fact, when implemented in the spectral domain using eigenvalue decomposition (EVD), the area under the receiver operating curve (AUC) obtained using these filters is larger than 0.99. However, when implemented as polynomials numerical errors considerably degrade the detection performance for the LS design, as evidenced by Fig. 2. In fact, the AUC becomes as low as 0.9 with false positive rates as high as 20% when using the LS filter. Using the robust method from Section 4, however, yields false positive rates lower than 0.2%.

In view of these issues, this paper sets out to answer two questions: (a) how does finite precision affect graph filters? and (b) how to mitigate these effects? To answer (a), we use a quantization noise model to estimate the magnitude of the numerical errors at the out-

put of graph filters implemented in fixed-point arithmetic. We then quantify the effects of the filter coefficients and the shift operator spectrum on the numerical performance of graph filters (Section 3). Based on these results, we formulate the problem of designing numerically robust filters as a regularized convex optimization problem that explicitly minimizes the output quantization errors and provide an answer to question (b) by solving this problem in a numerically stable fashion (Section 4). Finally, we illustrate the advantages of this design method in bit-accurate experiments.

2. GRAPH FILTERS AND QUANTIZATION NOISE

A *graph signal* is a signal that comes with a graph. Formally, a graph signal is a pair $(\mathbb{G}, \boldsymbol{x})$, where $\mathbb{G}$ is a weighted graph with node set $\mathcal{V}$ of cardinality $|\mathcal{V}| = n$ and $\boldsymbol{x} = [x_i]_{i\in[n]}$ is an $n \times 1$ vector that collects the samples of the signal. Underlying this pair is a bijection $\rho : \mathcal{V} \to [n]$ (i.e., an ordering) that maps the nodes in $\mathcal{V}$ to samples in $\boldsymbol{x}$. Thus, the value of the signal $\boldsymbol{x}$ on node $u \in \mathcal{V}$ is $x_{\rho(u)}$. We assume throughout that ρ is fixed.

Instead of the graph itself, GSP typically studies its matrix representation $\boldsymbol{S} \in \mathbb{R}^{n\times n}$, called the graph *shift operator* to mirror classical signal processing. Common choices include the adjacency matrix or one of the discrete Laplacians [1, 2]. We assume that $\boldsymbol{S}$ is consistent with the signal vector $\boldsymbol{x}$ in the sense that they employ the same permutation ρ of the nodes in $\mathcal{V}$. For $\boldsymbol{S}$ is diagonalizable, the graph Fourier transform of a graph signal $\boldsymbol{x}$ is defined as $\tilde{\boldsymbol{x}} = \boldsymbol{V}^{-1}\boldsymbol{x}$, where $\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$ is the EVD of $\boldsymbol{S}$. To make the exposition more concise, we assume the eigenvalues of $\boldsymbol{S}$ are real-valued.

The shift $\boldsymbol{S}$ induces a class of operators called filters. In this work, we focus on *linear shift-invariant* filters which are the counterpart of linear time-invariant filters in traditional signal processing. This class of filters is isomorphic to matrix polynomials in $\boldsymbol{S}$ [2], i.e., the output $\boldsymbol{y}$ of such filter for an input $\boldsymbol{x}$ can be written as

$$\boldsymbol{y} = \left(\sum_{k=0}^{L-1} h_k \boldsymbol{S}^k\right)\boldsymbol{x}, \tag{1}$$

where $\{h_k\}$ are the *filter coefficients* and L is the *filter length*. It is sometimes convenient to collect the filter coefficients into the $L \times 1$ vector $\boldsymbol{h} = [h_0 \ \dots \ h_{L-1}]^T$. We assume without loss of generality that $L < n$. Indeed, due to the Caley-Hamilton theorem, there exists a polynomial of order strictly less than n equivalent to any analytical function of the matrix $\boldsymbol{S}$ [12]. In practice, it may be advantageous to implement (1) recursively, in which case it becomes $\boldsymbol{y} = c_0 \prod_{k=1}^{L-1}(\boldsymbol{I} + c_k\boldsymbol{S})\boldsymbol{x}$, with $h_0 = c_0$ and $h_k = c_0 \cdot \sum \binom{\mathcal{C}}{k}$, $1 \le k \le L-1$, where $\mathcal{C} = \{c_n\}$ and $\binom{\mathcal{C}}{k}$ is the set of all products of k elements of $\mathcal{C}$.

Having specified that graph filters are of the form (1), the coefficients $\boldsymbol{h}$ uniquely determine the filter response. Hence, designing a filter reduces to determining the coefficients that yield a desired spectral response $\boldsymbol{d}$. This can be written as the problem of finding the vector $\boldsymbol{h}$ that satisfies

$$\boldsymbol{d} = \boldsymbol{\Psi}\boldsymbol{h}, \tag{2}$$

where $\boldsymbol{\Psi}$ is a Vandermonde matrix whose nodes are the eigenvalues of the shift operator. Explicitly, for λ_i denoting the i-th eigenvalue of $\boldsymbol{S}$, we have $[\boldsymbol{\Psi}]_{ij} = \lambda_i^{j-1}$. When the eigenvalues of $\boldsymbol{S}$ are distinct, $\boldsymbol{\Psi}$ is full rank and there always exists a unique solution to (2). Otherwise, approximate designs can be obtained by minimizing $\|\boldsymbol{\Psi}\boldsymbol{h} - \boldsymbol{d}\|_2$ or some alternative fit measure [6]. In some cases, the coefficients can also be designed using Chebyshev polynomials as in [3].

Remark 1. The system of equations in (2) is extremely ill-conditioned. Indeed, the condition number of a generic Vandermonde matrix grows exponentially with its dimension. Thus, designing graph filters using (2) is prone to errors even for moderately sized n. Interestingly, this is not the case of filter design in the time domain because $\boldsymbol{\Psi}$ is a Fourier matrix. Fourier matrices belong to the reduced set of Vandermonde matrices that are well-conditioned [13].

2.1. Fixed-point arithmetic and quantization noise

In pratice, the graph filter (1) is realized by a finite-precision machine, i.e., one that can only represent a finite set of numbers $\mathcal{Q}$. When an input or the result of an operation z is not in $\mathcal{Q}$, the machine replaces it by the closest representable number $\bar{z} \in \mathcal{Q}$. This process, called *quantization*, introduces a round-off error q to almost every computation. Formally, we write

$$\bar{z} = z + q. \tag{3}$$

Although quantization is typically a deterministic map, q can be modeled as a random variable independent of z under mild conditions [14]. For this reason, q is sometimes called the *quantization noise*.

The statistical characteristics of q depend on the set $\mathcal{Q}$, i.e., the number representation used by the machine. In this work, we focus on fixed-point representations and leave the study of floating-point ones for future work. Also, we assume in what follows that all machines are binary and employ the usual two's complement with zero number format [10]. Fixed-point numbers are specified by the length of their integer (B) and fractional (K) parts. For conciseness, we write $QB.K$ to denote a fixed-point number represented by $B + K + 1$ bits with one sign bit, B bits for the integer part, and K bits for the fractional part. Since K is constant, the binary decimal point is "fixed". Then, $\mathcal{Q} = \{k2^{-K} : k \in [-2^{B+K}, 2^{B+K} - 1]\}$ and q can be modeled as a random variable uniformly distributed in $\left[-2^{-K-1}, 2^{-K-1}\right]$, assuming B is large enough to avoid saturation. In other words, for $-2^B \le z \le 2^B - 2^{-K}$ in (3). The realizations of q are assumed across quantizations [14].

3. GRAPH FILTERS IN FIXED-POINT ARITHMETIC

In this section, we study the effects of quantization on graph filters. To reflect how calculations are typically carried out by processors, each multiplication and addition pair is grouped into a single multiply-and-accumulate (MAC) operation. Hence, we quantize after every MAC instead of each individual arithmetic operation. This mimics the common process of using a double precision register for MACs and quantizing the result only for memory storage [10, 14].

To proceed, note that (1) can be carried out in two steps: the shift operator application ($\boldsymbol{x}_k = \boldsymbol{S}\boldsymbol{x}_{k-1}$) and the filter MACs ($\sum h_k\boldsymbol{x}_k$). This suggest that graph filters can be written as a linear dynamical system, so that we can account for quantization in (1) by writing

$$\bar{\boldsymbol{x}}_{k+1} = \boldsymbol{S}\bar{\boldsymbol{x}}_k + \boldsymbol{v}_k \tag{4a}$$

$$\bar{\boldsymbol{y}} = \sum_{k=0}^{L-1} h_k\bar{\boldsymbol{x}}_k + \boldsymbol{w} \tag{4b}$$

where $\bar{\boldsymbol{x}}_k$ and $\bar{\boldsymbol{y}}$ are the quantized versions of $\boldsymbol{S}^k\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, $\bar{\boldsymbol{x}}_0 = \boldsymbol{x}$, and $\{\boldsymbol{v}_k, \boldsymbol{w}\}$ are $n \times 1$ random vectors representing the overall quantized arithmetic effect of each step. Note that $\boldsymbol{v}_k, \boldsymbol{w}$ are sums of independent random variables (the quantization noises q)

and are therefore not uniformly distributed. Their statistics depend on the number of operations that occurred in each step. From (4) and the quantization noise model from Section 2.1, we obtain the following result:

Proposition 1. *Consider a graph filter as in* (1) *of length* L *and coefficients* $\{h_k\}$ *whose underlying graph shift operator* $\boldsymbol{S}$ *is diagonalizable with eigenvalues* $\{\lambda_p\}$*. In* $QB.K$ *arithmetic with* B *large enough to avoid overflow, the quantization noise at the output filter is a zero-mean random variable* $\boldsymbol{q_y}$ *with variance*

$$\mathbb{E}\,\|\boldsymbol{q_y}\|_2^2 = \left(L + \|\boldsymbol{P}\boldsymbol{H}\|_F^2\right) n\sigma^2, \tag{5}$$

where $\sigma^2 = 2^{-2K}/12$*,* $\boldsymbol{P}$ *contains the first* $L-1$ *columns of* $\boldsymbol{\Psi}$*, and* $\boldsymbol{H}$ *is the* $(L-1)\times(L-1)$ *Hankel matrix of the filter coefficients* $\{h_k\}_{k\geq 1}$*. Explicitly,*

$$\boldsymbol{P} = \begin{bmatrix} 1 & \lambda_1 & \dots & \lambda_1^{L-2} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_n & \dots & \lambda_n^{L-2} \end{bmatrix}, \quad \boldsymbol{H} = \begin{bmatrix} h_1 & \dots & h_{L-1} \\ \vdots & \ddots & \vdots \\ h_{L-1} & \dots & 0 \end{bmatrix}.$$

Proof. The solution to the dynamical system (4a) is given by [15]

$$\bar{\boldsymbol{x}}_k = \boldsymbol{S}^k\boldsymbol{x} + \sum_{\ell=1}^{k} \boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1},$$

so that (4b) becomes

$$\begin{aligned} \bar{\boldsymbol{y}} &= \sum_{k=0}^{L-1} h_k(\boldsymbol{S}^k\boldsymbol{x} + \sum_{\ell=1}^{k} \boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1}) + \boldsymbol{w} \\ &= \boldsymbol{y} + \sum_{k=1}^{L-1}\sum_{\ell=1}^{k} h_k\boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1} + \boldsymbol{w}. \end{aligned} \tag{6}$$

According to (6), the quantized filter output can be written as the full precision output corrupted by a filtered noise $\boldsymbol{q_y} = \boldsymbol{y} - \bar{\boldsymbol{y}}$. Immediately, $\mathbb{E}\,\boldsymbol{q_y} = 0$, since $\boldsymbol{v}_k$ and $\boldsymbol{w}$ are sums of zero-mean random variables, namely q from (3).

To obtain (5), we evaluate the variance of $\boldsymbol{q_y}$. Explicitly,

$$\begin{aligned} \mathbb{E}\,\|\boldsymbol{q_y}\|^2 = \mathbb{E}\left\|\sum_{k=1}^{L-1}\sum_{\ell=1}^{k} h_k\boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1}\right\|^2 \\ + 2\sum_{k=1}^{L-1}\sum_{\ell=1}^{k} h_k\,\mathbb{E}\,\boldsymbol{w}^T\boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1} + \mathbb{E}\,\|\boldsymbol{w}\|^2. \end{aligned} \tag{7}$$

To simplify (7), start by recalling that the effects of different quantizations are i.i.d. Since $\boldsymbol{v}_k$ and $\boldsymbol{w}$ are zero-mean it holds that $\mathbb{E}\,\boldsymbol{w}\boldsymbol{v}_k^T = \mathbb{E}\,\boldsymbol{v}_k\boldsymbol{v}_\ell^T = 0$, for $k \neq \ell$. Hence, after inverting the order of the summations in the first term of (7) we obtain

$$\begin{aligned} \mathbb{E}\,\|\boldsymbol{q_y}\|^2 &= \sum_{\ell=1}^{L-1}\mathbb{E}\left\|\sum_{k=\ell}^{L-1} h_k\boldsymbol{S}^{k-\ell}\boldsymbol{v}_{\ell-1}\right\|^2 + \mathbb{E}\,\|\boldsymbol{w}\|^2 \\ &= \sum_{\ell=1}^{L-1}\mathrm{Tr}\left(\boldsymbol{T}_\ell\boldsymbol{C}_{v,\ell}\boldsymbol{T}_\ell^T\right) + \mathrm{Tr}\left(\boldsymbol{C}_w\right), \end{aligned} \tag{8}$$

where $\boldsymbol{T}_\ell = \sum_{k=\ell}^{L-1} h_k\boldsymbol{S}^{k-\ell}$, $\boldsymbol{C}_{v,\ell} = \mathbb{E}\,\boldsymbol{v}_\ell\boldsymbol{v}_\ell^T$, $\boldsymbol{C}_w = \mathbb{E}\,\boldsymbol{w}\boldsymbol{w}^T$, and we used the fact that $\|\boldsymbol{x}\|^2 = \mathrm{Tr}(\boldsymbol{x}\boldsymbol{x}^T)$.

We now proceed by evaluating the covariances of $\boldsymbol{v}_k$ and $\boldsymbol{w}$. First, notice that since the quantization noise is independent across quantizations, both $\boldsymbol{C}_{v,\ell}$ and $\boldsymbol{C}_w$ are diagonal matrices and their variance depends only on the number of operations at each step. From (4b), it is ready that it takes L MACs to calculate each element of $\bar{\boldsymbol{y}}$. Assuming $\boldsymbol{S}$ and $\bar{\boldsymbol{x}}$ are full matrices—the worst case scenario—, each element of $\bar{\boldsymbol{x}}_{k+1}$ in (4a) requires n MACs to be evaluated. It therefore holds that $\boldsymbol{C}_{v,\ell} = n\sigma^2\boldsymbol{I}$ and $\boldsymbol{C}_w = L\sigma^2\boldsymbol{I}$ with $\sigma^2 = 2^{-2K}/12$. Using these values in (8) gives

$$\mathbb{E}\,\|\boldsymbol{q_y}\|^2 = \left(L + \sum_{\ell=1}^{L-1}\left\|\sum_{k=\ell}^{L-1} h_k\boldsymbol{S}^{k-\ell}\right\|_F^2\right) n\sigma^2. \tag{9}$$

Finally, let $\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$ be the EVD of $\boldsymbol{S}$ where $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_p)$ is its eigenvalue matrix. Since the Frobenius norm is invariant under similarity transformations [12], we can write (9) as

$$\mathbb{E}\,\|\boldsymbol{q_y}\|^2 = \left(L + \sum_{\ell=1}^{L-1}\left\|\sum_{k=\ell}^{L-1} h_k\boldsymbol{\Lambda}^{k-\ell}\right\|_F^2\right) n\sigma^2. \tag{10}$$

Notice that the summation inside the norm in (10) is the convolution between the sequences $\{h_k\}$ and $\{\lambda_p^k\}$. Using the filter Hankel form, the second term of (10) can be written in matrix form to yield (5). ∎

Using Proposition 1, we can write the output of a graph filter in fixed-point arithmetic as $\bar{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{q_y}$, in terms of its full precision output $\boldsymbol{y}$ and a zero-mean random variable $\boldsymbol{q_y}$ whose variance is given in (5). Equivalently, (5) establishes the MSE incurred from using fixed-point arithmetic instead of full precision. In contrast to classical signal processing, the quantization error is affected by both the filter coefficients ($\boldsymbol{H}$) and shift operator ($\boldsymbol{P}$). This is due to the fact that the "classical shift" (directed cycle) is *lossless*, in the sense that its applications do not change the value of the signal. When $\boldsymbol{S}$ is the directed cycle, $\boldsymbol{P}$ is a Fourier matrix and (5) depends only on $\boldsymbol{H}$ because $\boldsymbol{P}^H\boldsymbol{P} = n\boldsymbol{I}$.

We can identify two sources of quantization error in (5). The first one is related to the filtering operation in (4b) and depends only on the filter length L. Hence, shorter filters are less prone to finite-precision effects. The second one is related to the successive applications of the shift operator in (4a) and depends on an interaction between the filter coefficients and the spectrum of $\boldsymbol{S}$. Thus, reducing the magnitude of the filter coefficients and eigenvalues of $\boldsymbol{S}$ also reduces the quantization errors. Note that, as expected, the overall MSE grows linearly with the size of the graph signal n.

These observations suggest three situations in which graph filters are susceptible to numerical issue: (i) short transition band; (ii) large spectral gains; and (iii) large shift operator spectral radius. The first situation also occurs in classical signal processing: abrupt transitions of the filter response require higher-order polynomials (large L) which increase the output quantization error. In GSP, however, the frequencies are no longer equally spaced. Hence, shift operators with dense spectra are more prone to this issue. In contrast to (i), (ii) and (iii) are issues exclusive to GSP. By (ii) we mean the amplification of small eigenvalues and/or attenuation large eigenvalues of the shift operator. Using the total variation orderings from [1, 16, 17], these are low-pass filters on graph Laplacians and high-pass filters on adjacency matrices. Such filters require larger coefficients, which leads to an increase in round-off errors. As for (iii), graph shift operators with large eigenvalues contribute considerably to increasing the output quantization noise, especially when their spectral radii are larger than one. These situations do

not arise in the time domain because the shift is isometric, i.e., all eigenvalues have magnitude one.

In the sequel, we put forward a design solution that mitigates these issues and explicitly reduces the output quantization error.

4. ROBUST GRAPH FILTER DESIGN

Given the central role of the shift operator spectrum on the numerical properties of graph filters, it is worth noting that appropriately choosing the underlying graph is a crucial step towards robustness to finite precision. Since this is not always possible, this section describes a coefficients design method to mitigate the quantization effects without changing $\boldsymbol{S}$.

To do so, note that the second term in (5) is convex with respect to $\boldsymbol{h}$. Therefore, we can explicitly minimize the output quantization noise by solving the penalized convex problem

$$\text{minimize} \quad \|\boldsymbol{d} - \boldsymbol{\Psi h}\|_2^2 + \eta^2 \left(h_0^2 + \|\boldsymbol{PH}\|_F^2\right). \qquad (11)$$

Note that (11) includes regularization for coefficient h_0 even though it does not directly affect (5). This improves the stability of the design solution and accounts for the fact that a large zero-th order term would require additional integer bits to avoid overflow.

For any given filter length L, (11) allows us to explicitly minimize the output quantization error and design graph filters more robust to finite-precision arithmetic. Although it would seem that we should choose the shortest filter possible, the filter length and coefficients magnitude can interact in non-trivial ways: shorter filters may require larger coefficients to achieve the same design accuracy leading to numerically inaccurate filters. The best way to deal with this issue is therefore to design filters of different lengths and use the design with the smallest (5). This is typically not an issue since the complexity of solving (12) is $\mathcal{O}(n^3)$.

For this robust design method to be effective, (11) must be solved in a numerically stable manner. Otherwise, we may need to increase η^2 and trade-off fit to deal with large coefficients. This is particularly challenging in this case due to the fact that $\boldsymbol{\Psi}$ is generally ill-conditioned (see Remark 1). Moreover, the smallest-to-largest entry ratio of $\boldsymbol{\Psi}$ often exceed the machine precision by several orders of magnitude, especially for large L or ill-conditioned $\boldsymbol{S}$. Solving (11) using a generic quadratic solver can therefore be unstable even for moderately sized n [13, 18, 19]. Hence, we propose using a stable iterative solver such as LSQR, a standard conjugate gradient-based regularized solver for $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2$ [20]. The following proposition writes problem (11) in this form.

Proposition 2. *Problem* (11) *is equivalent to*

$$\text{minimize} \quad \left\| \begin{bmatrix} \boldsymbol{\Psi} \\ \eta \hat{\boldsymbol{Q}} \end{bmatrix} \boldsymbol{h} - \begin{bmatrix} \boldsymbol{d} \\ \boldsymbol{0} \end{bmatrix} \right\|_2^2, \qquad (12)$$

where $\hat{\boldsymbol{Q}}$ is the Cholesky decomposition of $\boldsymbol{Q}$ given by

$$\boldsymbol{Q} = \begin{bmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \bar{\boldsymbol{Q}} \end{bmatrix} \quad \textit{with} \quad \bar{\boldsymbol{Q}} = \left[\sum_{p=1}^{n} \min(i,j) \lambda_p^{i+j} \right].$$

Proof. The Frobenius norm in (11) expands to

$$\begin{aligned} \|\boldsymbol{PH}\|_F^2 &= \sum_{p=1}^{n} \sum_{\ell=1}^{L-1} \left(\sum_{k=\ell}^{L-1} h_k \lambda_p^k \right)^2 \\ &= \sum_{p=1}^{n} \left[\sum_{\ell=1}^{L-1} \sum_{i=\ell}^{L-1} \sum_{j=\ell}^{L-1} h_i h_j \lambda_p^{i+j} \right]. \end{aligned}$$

Exchanging the order of the summation in the brackets yields for each p

$$\sum_{\ell=1}^{L-1} \sum_{i=\ell}^{L-1} \sum_{j=\ell}^{L-1} h_i h_j \lambda_p^{i+j} = \sum_{i=1}^{L-1} \sum_{j=1}^{L-1} \min(i,j) h_i h_j \lambda_p^{i+j}. \qquad (13)$$

Using (13), the objective of (11) can be written as

$$\|\boldsymbol{d} - \boldsymbol{\Psi h}\|_2^2 + \eta^2 \boldsymbol{h}^T \boldsymbol{Q h}$$

for $\boldsymbol{Q}$ as in (12). It is straightforward that this expression is equal to the objective of (12). ■

5. EXPERIMENTS

Before illustrating the fixed-point arithmetic results from the previous sections, we provide more details about the introductory example from Figs. 1 and 2. The graph with $n = 100$ nodes was obtained by simulating a typical covariance matrix by randomly choosing an orthogonal matrix $\boldsymbol{V}$ and selecting a spectrum with two rates of decay: $\lambda_1 = 1$, $\lambda_p = 0.95\lambda_{p-1}$ for $p \in [2, 10]$, and $\lambda_p = 0.99\lambda_{p-1}$ for $p \geq 11$. We then sparsify the resulting matrix by setting to zero all elements with magnitude lower than 10^{-2} and normalize the result so that the largest eigenvalue is one. Using this graph, we set out to detect a bandlimited signal whose spectrum is given by the filter response in Fig. 1. This signal is embedded in a white Gaussian noise with covariance matrix $\sigma_n^2 \boldsymbol{I}$, $\sigma_n^2 = 10^{-1}$. The detector is based on thresholding the output power of a matched graph filter, which is equivalent to performing linear discriminant analysis (LDA). The method is implemented in 32 bits floating-point arithmetic (IEEE 754 *single* or *binary32* [21]), ubiquitous in general purpose processors and GPUs. Results shown in Fig. 2 are obtained from 2000 signal realizations, half of which contain only noise, and for filters designed by solving (2) (*LS design*) and using (12) with $\eta = 10^{-6}$ (*robust design*). All designs are obtained using LSQR.

We now turn to our fixed-point results. Since bit-accurate simulation is slow and computationally intensive, we use graphs with $n = 20$ nodes. The graph signals $\boldsymbol{x}$ are taken to be realizations of an i.i.d. zero-mean Gaussian vector with unit variance so as to excite all the graph modes. First, we choose a random orthogonal $\boldsymbol{V}$ with the spectrum shown in Fig. 3a. Besides the LS and robust designs, we also show results for a simple *shrinkage design* that uses $\hat{\boldsymbol{Q}} = \boldsymbol{I}$ in (12). The variance of $\boldsymbol{q_y}$ is estimated based on 100 realizations of graph signals and displayed in Fig. 3b. Note that due to the large gap between the eigenvalues in the passband and those in the stopband of the filter, this is a simpler design scenario. Still, using the weighting from (12) reduces the output quantization noise by up to 8 dB compared to the shrinkage design.

We conclude by illustrating some limitations of the quantization noise model from Section 2.1. To do so, we build a geometric graph by positioning the nodes uniformly at random in a unit square and connecting two nodes if their Euclidian distance is less 0.3. We normalize the resulting shift operator (adjacency matrix) so that its maximum eigenvalue is 1.2 (Fig. 4a) and design filters that attenuate all except the five frequency bands with largest magnitude ($\eta^2 = 10^{-11}$). Since $\|\boldsymbol{S}\| > 1$, the elements of $\boldsymbol{S}^k$ grow considerably as k increases and i.i.d. uniform quantization noise is no longer a suitable error model. Moreover, the wordlength must be doubled simply to avoid overflow saturation. Still, the design technique from Section 4 can be used to reduce the overall numerical errors of the graph filter (Fig. 4b). Once again, though simple shrinkage improves the filter numerical performance, the optimal weights from (12) yield from 2 to 20 dB reductions in the output quantization noise variance.

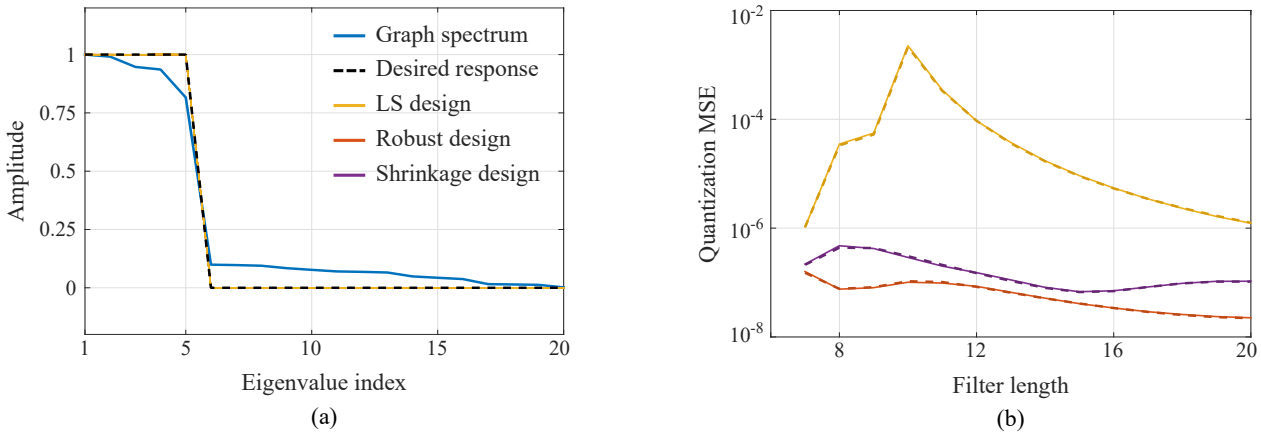

Fig. 3. Design example in signed $Q13.18$: (a) graph spectrum and filter responses; (b) output quantization MSE [(5) in dashed curves].

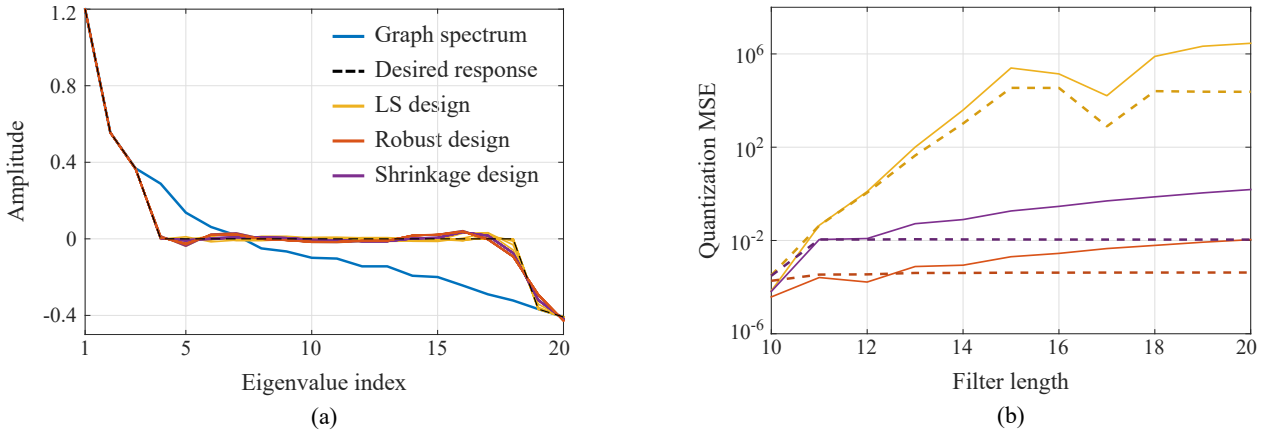

Fig. 4. Design example in signed $Q43.20$: (a) graph spectrum and filter responses; (b) output quantization MSE [(5) in dashed curves].

6. CONCLUSION

As graph filters become central pieces of GSP applications, it becomes increasingly important to understand and improve their numerical properties. In this work, we have shown that graph filters are sensitive to numerical errors by quantifying the effects of the filter coefficients and the shift operator spectrum on the output quantization noise for fixed-point implementations. Using these results, we then put forward a robust design method that explicitly minimizes these round-off errors and illustrated its performance. In the future, it is worth extending these results to floating-point arithmetic and investigate methods to design numerically robust graph shift operator.

7. REFERENCES

[1] D. Shuman, S. 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[3], pp. 83–98, 2013.

[2] A. Sandryhaila and J. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61[7], pp. 1644–1656, 2013.

[3] D. Shuman, P. Vandergheynst, and P. Frossard, "Distributed signal processing via Chebyshev polynomial approximation," 2011, arXiv:1111.5239.

[4] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Int. Conf. on Acoust., Speech and Signal Process.*, 2012, pp. 3921–3924.

[5] S. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60[6], pp. 2786–2799, 2012.

[6] S. Segarra, A. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65[15], pp. 4117–4131, 2017.

[7] O. Teke and P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs—Part I: Fundamentals," *IEEE Trans. Signal Process.*, vol. 65[2], pp. 409–422, 2017.

[8] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Trans. Signal Process.*, vol. 65[2], pp. 274–288, 2017.

[9] P. Diniz, E. da Silva, and S. Netto, *Digital Signal Processing: System Analysis and Design*. Cambridge, 2002.

[10] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*. Springer, 2014.

[11] R. Lyons, *Understanding Digital Signal Processing*. Prentice Hall, 2010.

[12] R. Horn and C. Johnson, *Matrix analysis*. Cambridge University Press, 2013.

[13] V. Pan, "How bad are Vandermonde matrices?" *SIAM J. on Matrix Anal. and Applicat.*, vol. 37[2], pp. 676–694, 2016.

[14] B. Widrow and I. Kollár, *Quantization noise: Roundoff Error in Digital Computation, Signal Processing, Control, and Communications*. Cambridge, 2008.

[15] T. Kailath, *Linear systems*. Prentice-Hall, 1980.

[16] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15[6], pp. 1373–1396, 2003.

[17] A. Sandryhaila and J. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62[12], pp. 3042–3054, 2014.

[18] G. Golub and C. van Loan, *Matrix Computations*. Johns Hopkins University Press, 2012.

[19] N. Higham, *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[20] C. Paige and M. Saunders, "LSQR: An algorithm for sparse linear equations and sparse least squares," *ACM Trans. Math. Softw.*, vol. 8[1], pp. 43–71, 1982.

[21] IEEE 754-2008, "IEEE standard for floating-point arithmetic," 2008.