Porting Signal Processing from Undirected to Directed Graphs: Case Study Signal Denoising with Unrolling Networks

Vedran Mihal
Department of Computer Science
ETH Zürich
Zürich, Switzerland
vedran.mihal@inf.ethz.ch

Bastian Seifert
Department of Computer Science
ETH Zürich
Zürich, Switzerland
bastian.seifert@inf.ethz.ch

Markus Püschel
Department of Computer Science
ETH Zürich
Zürich, Switzerland
pueschel@inf.ethz.ch

***Abstract*—Directionality is an essential feature of many real-world networks, but problematic in graph signal processing (GSP) because there is no obvious choice of Fourier basis. In this work we investigate how to port GSP methods from undirected to directed graphs using recent work on graph signal denoising using trainable networks as a case study. We consider five notions of directed Fourier bases from the literature and different approaches for porting, from ad-hoc to conceptual. Our experimental results show that directionality does matter, the importance of a shift operator related to the chosen basis, and which directed Fourier basis may be best suited for applications. The best variant also provides a promising method for denoising signals on directed graphs.**

***Index Terms*—Graph signal processing, directed graph, graph Fourier transform, graph Fourier basis, neural net**

I. Introduction

Graph signal processing (GSP) is an evolving field of research that generalizes classical signal processing to data, or signals, associated with the nodes of a graph [1]. Two basic variants exist, associated with using the adjacency matrix [2] or Laplacian [3] as shift operator. The vast majority of GSP has focused on undirected graphs. The reason is both conceptual (no direction is needed to model relationships between nodes) and mathematical: if the shift is symmetric, it can be diagonalized. This means that Fourier basis and associated transform are well-defined and the latter is even orthogonal.

However, direction can matter but in this case the shift offers (in general) no well-defined Fourier basis and the more general Jordan basis is not computable (except for very small sizes) [4]. Various works have addressed this problem and proposed Fourier bases and associated transforms for directed graphs by relaxing the requirement to be an eigenbasis in different ways [5]–[8], but have not yet been used much in applications [9].

Many GSP methods for undirected graphs rely on the Fourier basis and transform. So a fundamental question is: can one port such a method to work on directed graphs, and how to do so by leveraging the above-mentioned proposed substitute bases and transforms.

Contribution. In this paper, we build on the recent work [10], which proposes methods for denoising signals on undirected graphs by unrolling an iterative denoising algorithm into a trainable neural network. Internally the approach uses the adjacency matrix as shift and the associated Fourier basis. We consider different ways of porting the method to the five different types of Fourier bases proposed in [5]–[8], from simply ignoring directions, to conceptual porting (i.e., replacing both the shift operator and the Fourier basis). Our results show that ignoring directions yields inferior results, the importance of having a shift related to the basis, and hint at which basis may be best for applications. The best variant also provides a method for denoising signals on directed graphs.

II. Signal processing on Graphs

We provide a brief background on GSP and describe challenges specific to directed graphs.

Graphs. A graph $G = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$. An edge is an ordered pair of vertices $(x_i, x_j) \in \mathcal{V} \times \mathcal{V}$. For undirected graphs, $(x_i, x_j) \in \mathcal{E}$ implies $(x_j, x_i) \in \mathcal{E}$.

Graph signals. A graph signal s on a graph $G = (\mathcal{V}, \mathcal{E})$ maps nodes to values: $s\colon \mathcal{V} \to \mathbb{C}$. We fix a vertex ordering $\mathcal{V} = \{x_1, \ldots, x_n\}$, and represent a graph signal by the vector $\mathbf{s} = (s_i)_{i=1,\ldots,n} \in \mathbb{R}^n$, where $s_i = s(x_i)$.

Graph shifts. Two basic GSP versions exist, depending on the chosen linear shift (or variation) operator S [2], [3], which is the defining concept for any linear SP framework [11], [12].

In [2], S is the adjacency matrix, i.e., the matrix $A \in \mathbb{R}^{n \times n}$ with entries $A_{i,j} = 1$ if $(x_i, x_j) \in \mathcal{E}$ and $= 0$ else. In [3], S is the Laplacian matrix $L = D - A$, where $D = \mathrm{diag}(\deg(x_1), \ldots, \deg(x_n))$ collects the degrees $\deg(x) = |\{(x, y) : (x, y) \in \mathcal{E}\}|$, where $|\cdot|$ denotes set cardinality.

Filters are then just polynomials in the shift operator.

Graph Fourier basis: undirected graph. In this case, S is symmetric and can be diagonalized by an orthogonal matrix V: $S = V \Lambda V^{-1}$. The columns $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of V form the (orthonormal) Fourier basis and $\widehat{\mathbf{s}} = \mathcal{F}\mathbf{s} = V^{-1}\mathbf{s}$ is the Fourier transform. Λ is a diagonal matrix containing the real eigenvalues, typically ordered by magnitude.

Challenges for directed graphs. For directed graphs GSP becomes problematic since S is in general not diagonalizable, and if so, the basis is not orthogonal [1, Sec. III.A]. [2] proposed the Jordan decomposition instead, but it is not computable for large sizes [4]. The challenge is how to define a suitable substitute for a Fourier basis of a directed graph.

III. Fourier bases for directed graphs

Several notions of Fourier bases for directed graphs have been proposed in the literature, each offering advantages and disadvantages. We use prior notation and review the most important details that we use later in our work.

Spread frequencies (SprFreq). In [6], extending [13], [14], the authors propose a Fourier basis that is orthogonal and whose frequencies are, in a sense, evenly distributed in the spectral domain. The frequency of a basis vector $\mathbf{u} = (u_1, \dots, u_n)$ in the spectral domain is determined by the directed variation

$$\mathrm{DV}(\mathbf{u}) = \sum_{i,j=1}^{n} A_{i,j}[u_i - u_j]_+^2, \tag{1}$$

where $[x]_+ = \max(x, 0)$. The directed variation of a constant vector $\mathbf{v}_1$ is $\mathrm{DV}(\mathbf{v}_1) = 0$, while the basis vector $\mathbf{v}_n$ with maximal directed variation is found by solving a maximization problem with $\mathrm{DV}(\mathbf{u})$ as objective function. The remaining basis vectors are found by minimizing the spectral dispersion function $\delta(V) = \sum_{i=1}^{n-1}(\mathrm{DV}(\mathbf{v}_{i+1}) - \mathrm{DV}(\mathbf{v}_i))^2$, which measures how evenly spread the frequencies are.

The resulting basis is real, orthogonal, but becomes very expensive to compute for $n > 1000$ nodes. There is no obvious associated sparse shift operator diagonalized by the basis.

Hermitian Laplacian (HermL). The work in [5] changes the notion of Laplacian to

$$L_q = D - \Gamma_q \odot \overline{A}. \tag{2}$$

Here, $\overline{A}$ is the *undirected* adjacency matrix, $\odot$ the pointwise (Hadamard) product, and $(\Gamma_q)_{i,j} = \exp(2\pi q \mathrm{j}(A_{i,j} - A_{j,i}))$, for $q \in [0, 1)$, encodes the directionality of the graph. The authors propose $q = 0.02$, but it is stated that the conducted experiments are insensitive to small changes in q.

Since L_q is a Hermitian matrix, all its eigenvalues are real and it is diagonalized by a unitary matrix V which defines the Fourier basis. The basis vectors are ordered with respect to total variation, defined as [3]

$$\mathrm{TV}_2(\mathbf{u}) = \sum_{(x_i,x_j)\in\mathcal{E}} |u_i - u_j|^2.$$

For small $q \ll 1$ this order corresponds to the order of the (real) eigenvalues. The basis is complex, orthogonal, and has L_q as the associated shift operator.

Stable approximation (StabApp). The work in [7] uses an iterative algorithm to construct a numerically stable, approximate diagonalization of the adjacency matrix A. Namely, for $\epsilon > 0$,

$$A = V_\epsilon \Lambda_\epsilon V_\epsilon^{-1}, \tag{3}$$

where Λ_ϵ is upper triangular with all off-diagonal entries $< \epsilon$ and maintains the eigenstructure of A. The frequency ordering is w.r.t. [2]:

$$\mathrm{TV}_1^A(\mathbf{u}) = \|\mathbf{u} - \tfrac{1}{|\lambda_{\max}|} A\mathbf{u}\|_1, \tag{4}$$

where $|\lambda_{\max}|$ is the maximal eigenvalue magnitude of A.

The columns of V_ϵ form the Fourier basis, which is complex, not orthogonal, and there is no associated sparse diagonalized shift operator.

Generalized boundary conditions (GenBC). The classical 1D-time DFT assumes periodicity, which is equivalent to adding a backwards edge to a directed path graph, making it a circle. This motivates the work in [8], which adds a small number of edges to the graph to make A diagonalizable:

$$A + B = V \Lambda V^{-1}, \tag{5}$$

where the sparse B collects the added edges (GenBC1). The columns of V then serve as Fourier basis for A. A variant (GenBC2) computes instead a B' which also removes all zero eigenvalues to make $A+B$ invertible. Both methods also work with L instead of A. The frequencies are ordered by TV_1^A.

The obtained Fourier basis is complex, not orthogonal, and has an associated shift operator ($A + B$ or $A + B'$).

IV. Unrolling Networks on Directed Graphs for Denoising

The idea of unrolling networks [15] is to unroll an iterative optimization method into a neural network, making it possible to learn its parameters by end-to-end training. Here we first recall a recently proposed algorithm [10, Alg. 1], which applies this technique to denoising signals on undirected graphs. Then we propose several approaches to port this method to directed graphs, which are evaluated in the next section.

A. Graph Unrolling Networks: Signal Denoising via Sparse Coding

Given are k measurements $T = [\mathbf{t}^{(1)}, \dots, \mathbf{t}^{(k)}]$ of noisy signals on an undirected graph G of the form

$$\mathbf{t}^{(i)} = \mathbf{x}^{(i)} + \mathbf{e}^{(i)}, \quad \mathbf{x}^{(i)} = H\mathbf{s}^{(i)}, \tag{6}$$

where the $\mathbf{e}^{(i)}$ are noise, and the $\mathbf{x}^{(i)}$ are noiseless graph signals assumed to be generated from sparse graph signals $\mathbf{s}^{(i)}$ by convolution with a fixed, unknown filter H. The goal of [10, Alg. 1] is to denoise, i.e., recover $\mathbf{x}^{(i)}$ from the $\mathbf{t}^{(i)}$.

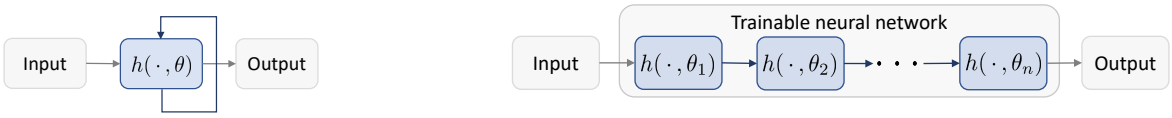
Graph sparse coding. If the filter H in (6) were known, $\mathbf{x}$ could be recovered by solving the optimization problem

$$\min_{\mathbf{s}\in\mathbb{R}^n} \tfrac{1}{2}\|\mathbf{t} - \mathbf{x}\|_2^2 + \alpha\|\mathbf{s}\|_1, \quad \text{s.t. } \mathbf{x} = H\mathbf{s}, \tag{7}$$

where $\|\mathbf{s}\|_1$ is a regularization term to promote sparsity of $\mathbf{s}$ and α is a hyperparameter. Classically, such an optimization problem is solved using an iterative algorithm.

Iterative algorithm. As iterative algorithm, which will then be unrolled into a neural network, [10] uses the half-quadratic splitting algorithm. It solves (7) by variable-splitting followed by alternating minimization. For this the following penalty function (μ_1, μ_2 are step sizes) is used:

$$\begin{aligned} L(\mathbf{x}, \mathbf{s}, \mathbf{z}) =& \tfrac{1}{2}\|\mathbf{t} - \mathbf{x}\|_2^2 + \alpha\|\mathbf{z}\|_1 \\ &+ \tfrac{\mu_1}{2}\|\mathbf{x} - H\mathbf{s}\|_2^2 + \tfrac{\mu_2}{2}\|\mathbf{z} - \mathbf{s}\|_2^2. \end{aligned} \tag{8}$$



(a) Iterative algorithm.

(b) Unrolled algorithm.

Fig. 1: The high-level idea of algorithm unrolling. An iterative algorithm (a) is converted into a neural network (b) by unrolling its iterations h. The parameter set θ can now be learned from training data, and even separately for each iteration (θ_i) for an additional degree of freedom. Adapted from [16].

The variable $\mathbf{z}$ is introduced to separate the non-linearity from the linear parts. Minimizing the penalty function for each variable leads to the updates

$$\mathbf{x} \leftarrow \tfrac{1}{1+\mu_1}(\mathbf{t} + \mu_1 H\mathbf{s}), \tag{9}$$

$$\mathbf{s} \leftarrow (\mu_1 H^T H + \mu_2 I)^{-1}(\mu_1 H^T \mathbf{x} + \mu_2 \mathbf{z}), \tag{10}$$

$$\mathbf{z} \leftarrow \operatorname{argmin}_{\mathbf{z}} \|\mathbf{z}\|_1, \tag{11}$$

in each step of the iteration.

Unrolled graph sparse coding. The basic idea of unrolling networks [15], [16] is to take the linear operations followed by a non-linearity in an iterative algorithm (e.g., the one described above) and map each iteration to a network layer. The procedure is illustrated in Fig. 1. This enables learning of the parameters μ_1, μ_2, *and the filter coefficients* H from training data.

Learning the filter H and the parameters μ_1, μ_2 is done in [10] with a trainable graph convolution. The trainable filter coefficients are collected in the five-mode tensor $\mathbb{H} \in \mathbb{R}^{L\times K_1\times K_2\times n\times n}$, where as input the convolution takes a batch of K_1 input signals $X = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(K_1)}] \in \mathbb{R}^{n\times K_1}$ and outputs a batch of K_2 output signals $Y = [\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(K_2)}]$. The k_2th channel output of the trainable graph convolution $\mathbb{H} *_a X$ is defined as

$$\mathbf{y}^{(k_2)} = \sum_{\ell=1}^{L}\sum_{k_1=1}^{K_1} (\mathbb{H}_{(\ell,k_1,k_2)} \odot A^\ell)\mathbf{x}^{(k_1)}. \tag{12}$$

To keep the number of training parameters low, the filter coefficients $\mathbb{H}$ are parameterized by a learnable kernel function, which uses a truncated version $\{\mathbf{v}_1, \dots, \mathbf{v}_p\}$ of the graph Fourier basis V as feature input:

$$\Psi_{i,j} = \psi_w(\mathbf{v}_j - \mathbf{v}_i) \in \mathbb{R}, \tag{13}$$

where ψ_w is implemented with a multilayer perceptron. The filter coefficients $\mathbb{H}_{\ell,k,h,i,j}$ are trained with (13), implementing an edge-weight sharing mechanism between the layers.

Using $\Sigma = [\mathbf{s}^{(1)}, \dots, \mathbf{s}^{(k)}]$ and unrolling the steps (9)–(11) with a trainable graph convolution yields layers of the form

$$X^{(b)} \leftarrow \mathbb{A} *_a T + \mathbb{B} *_a \Sigma^{(b-1)},$$
$$\Sigma^{(b)} \leftarrow \mathbb{D} *_a X^{(b)} + \mathbb{E} *_a Z^{(b-1)},$$
$$Z^{(b)} \leftarrow \operatorname{thresh}_\alpha(\Sigma^{(b)}),$$

with b indicating the layer and $\mathbb{A}, \mathbb{B}, \mathbb{D}, \mathbb{E}$ are trainable filter coefficients. Here, $\operatorname{thresh}_\alpha$ is the soft-thresholding function, the analytical solution of (11).

B. Porting to Directed Graphs

Our goal is to port the graph signal denoising in Section IV-A from undirected to directed graphs with adjacency matrix A. To do so, we first identify the algorithmic components specific to the graph structure. These are the adjacency matrix A as the shift in (12) and the associated Fourier basis V used in (13). The challenges thus are (a) the choice of shift A, since (12) shows that zero rows (i.e, sinks in G) are a problem as they fix the corresponding (to be estimated) signal value to zero; and (b) the choice of V as several options exist (Section III), and not all have an associated shift.

We consider the following variants:

Baseline: Forget directions (ForgetDir). Our baseline considers the hypothesis that directions do not matter. We choose the undirected version of A: $\overline{A} = \max(A, A^T)$ (elementwise), and the associated diagonalizing basis $\overline{V}$.

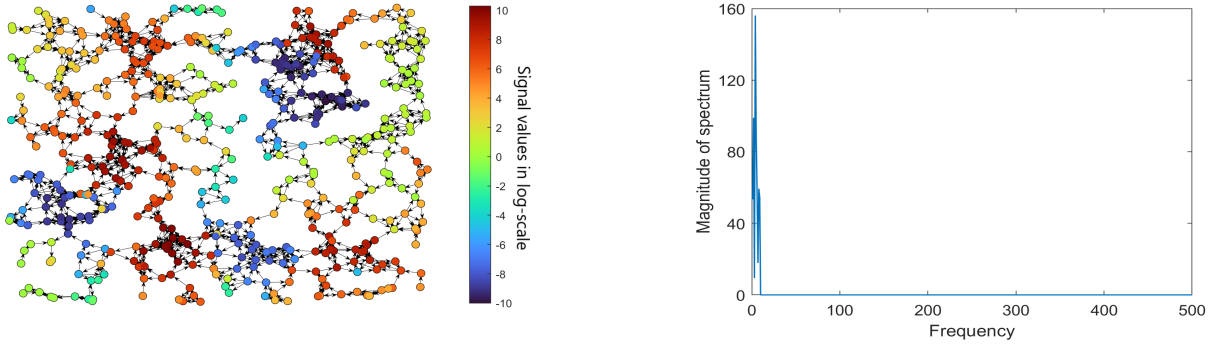
Ad-hoc (AdHoc). We keep $\overline{V}$ but choose (the directed) A as shift, removing sinks by adding self-loops to them to obtain A'. This way the information contained in the measured signal at the sink is taken into account.

Conceptual porting. We use one of the five Fourier bases (SprFreq, HermL, StabApp, GenBC1, GenBC2) from Section III as V to encode the direction information in the weight-sharing mechanism (13). As associated shift we consider three choices: (a) *Self-loops (SelfLoops)* uses the ad-hoc shift A' as above; (b) *Undirected shift (UndShift)* uses the associated undirected shift $\overline{A}$ from above; and, as conceptually "cleanest" porting, (c) *Associated shift (AssocShift)* uses the shift associated with the chosen basis if possible. This means L_q in (2) for HermL, and $A + B$, $A + B'$ in (5) for GenBC1 and GenBC2, respectively. SprFreq and StabApp have no obvious sparse shift operator.

V. Experimental Results

In this section we numerically evaluate our proposed variants for graph signal denoising on directed graphs. We do this by closely replicating a synthetic task used in [10, Table I] on noisy low frequency signals, but adapted to directed graphs.

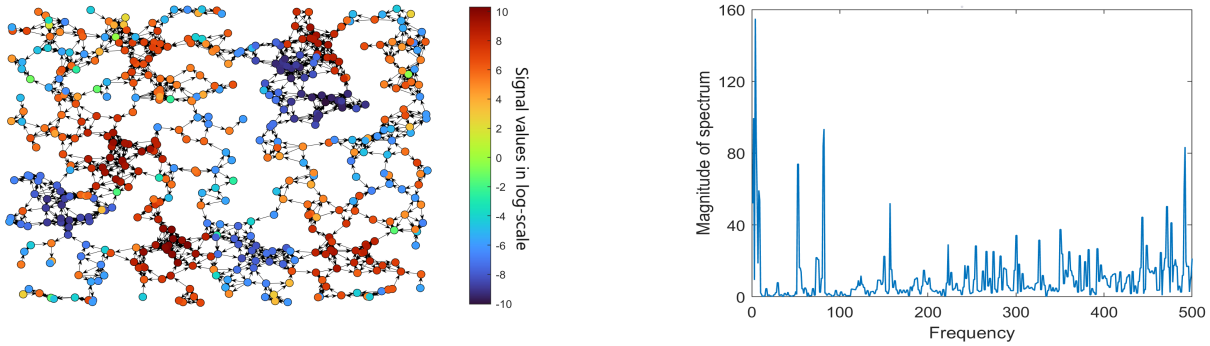
Graphs. We generate graphs using a random geometric graph model as in [10]. For this, we sample 500 points in the unit square and connect them if their distance is less than 0.065. To obtain a directed graph, we randomly fix an order on the vertices $\{x_1, \dots, x_n\}$. Then we set the direction of each edge (x_i, x_j), for $i < j$, with probability p to $x_i \to x_j$ and with probability $1 - p$ to $x_j \to x_i$. $p = 1$ or $= 0$ would yield



(a) Original signal

(b) Spectrum (GenBC2) of the original signal

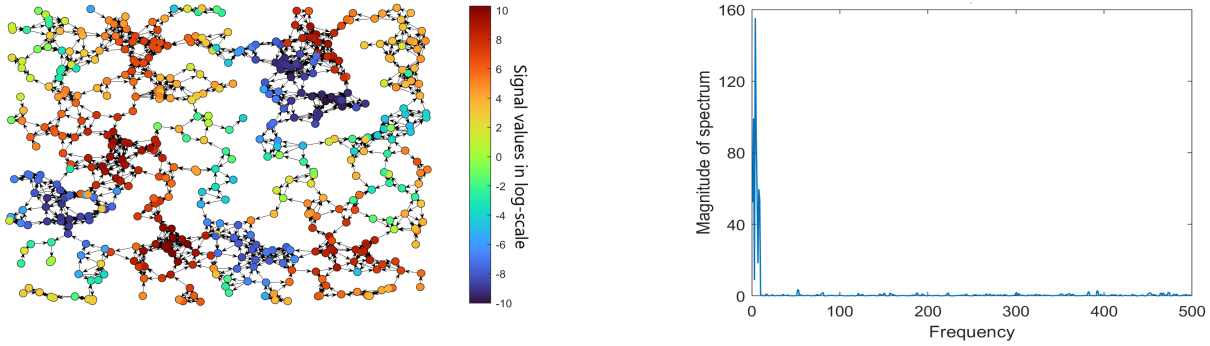
Fig. 2: Original signal (a) on graph, and (b) its spectrum.



(a) Noisy signal

(b) Spectrum (GenBC2) of the noisy signal

Fig. 3: Noisy version of the signal in Figure 2 with an SNR ≈ 2.



(a) Denoised signal

(b) Spectrum (GenBC2) of the denoised signal

Fig. 4: The signal from Figure 3 after denoising.

a directed acyclic graph, in a sense a worst case for Fourier analysis, since the adjacency matrix A has only one eigenvalue zero. We consider $p \in \{0.5, 0.9\}$.

Signals. We consider low frequency signals $\mathbf{x}$ using the proposed models in Section III. For each directed Fourier basis (SprFreq, HermL, StabApp, GenBC1, GenBC2), we use random linear combinations of the ten Fourier basis vectors with lowest frequencies. This means that for a given graph we have 25 experiments overall: five methods of porting (Section IV-B) combined with five notions of low frequency signals. For each basis we generate six sets of 100 signals.

Experimental setup. As a noise model we add Gaussian noise $\mathbf{e}$ to the low frequency signals $\mathbf{x}$, resulting in a normalized mean square error (NMSE) of $\|\mathbf{t}-\mathbf{x}\|_2^2/\|\mathbf{x}\|_2^2 \approx 0.5$, i.e., a very low SNR of ≈ 2 for the noisy signals $\mathbf{t} = \mathbf{x} + \mathbf{e}$.

Hyperparameters. We set the multilayer perceptron (13) to have one hidden layer of length 64, and use in (13) a truncated graph Fourier basis (whose form depends on the used porting method from Section IV-B) with 100 vectors. For the unrolling network we use one layer and train the model with 2000 epochs and a learning rate of 0.005 using the Adam optimizer [17]. The training for SprFreq (real basis)

	ForgetDir	AdHoc	SelfLoops	UndShift	AssocShift
$p = 0.5$					
SprFreq	0.43 ± 0.01	0.46 ± 0.01	0.35 ± 0.01	$\mathbf{0.19} \pm 0.00$	-
HermL	0.28 ± 0.01	0.31 ± 0.01	0.19 ± 0.00	$\mathbf{0.07} \pm 0.01$	0.19 ± 0.04
StabApp	0.13 ± 0.00	$\mathbf{0.08} \pm 0.00$	0.09 ± 0.00	0.10 ± 0.00	-
GenBC1	0.10 ± 0.00	$\mathbf{0.01} \pm 0.00$	$\mathbf{0.01} \pm 0.00$	0.03 ± 0.00	$\mathbf{0.01} \pm 0.00$
GenBC2	0.10 ± 0.00	$\mathbf{0.01} \pm 0.00$	$\mathbf{0.01} \pm 0.00$	0.03 ± 0.00	$\mathbf{0.01} \pm 0.00$
$p = 0.9$					
SprFreq	0.19 ± 0.01	0.19 ± 0.01	0.12 ± 0.00	$\mathbf{0.06} \pm 0.00$	-
HermL	0.21 ± 0.01	0.28 ± 0.01	0.16 ± 0.01	$\mathbf{0.07} \pm 0.00$	0.19 ± 0.01
StabApp	0.24 ± 0.00	$\mathbf{0.10} \pm 0.00$	0.11 ± 0.00	0.11 ± 0.00	-
GenBC1	0.15 ± 0.01	$\mathbf{0.03} \pm 0.00$	$\mathbf{0.03} \pm 0.01$	0.06 ± 0.01	0.05 ± 0.00
GenBC2	0.15 ± 0.00	0.06 ± 0.00	0.06 ± 0.00	0.05 ± 0.00	$\mathbf{0.01} \pm 0.00$

TABLE I: NMSE results of denoising low-frequency signals.

and GenBC is over the real numbers; for the latter by choosing conjugate coefficients for conjugate base vectors. For HermL and StabApp this is not possible. The hyperparameter *weight decay* of the Adam optimizer is obtained for each method by a search using the first of the six sets of 100 signals.

After obtaining the weight decay hyperparameter, we train the unrolling network on the five remaining sets of 100 signals and report the NMSE of the denoised signals.

Implementation. We implemented the method from scratch and first confirmed that we could reproduce the results from [10] for undirected graphs.

Results. In Table I we show the mean NMSE, with standard deviation over the remaining five sets, after denoising a random graph with $p = 0.5$ and $p = 0.9$.

In Figures 2 to 4, we show one experiment as example. We plot the original, the noisy, and the denoised version of the same signal. In this case we use the Fourier basis GenBC2 and the associated shift (AssocShift), which, by Table I, achieves an NMSE of 0.01. Thus, the method successfully learns a low pass-filter without prior information on the width (here ten) of the signals' frequency band.

Discussion. The neural network approach makes the denoising method highly adaptable, so it is not surprising that in each experiment some denoising is achieved. However, the achieved NMSEs differ substantially by up to 50 times.

The first main finding is that ForgetDir never performs best, i.e., the original algorithm is not a universal denoiser for all notions of directed low frequency signals.

The second main finding is that the best results are achieved with the low-frequency signals using GenBC and the associated shift and basis (AssocShift), which is the cleanest porting. AdHoc and SelfLoops also perform well, as their shift A' is close to the associated shift $A + B$, respectively $A + B'$.

The pointwise multiplication of A by learnable coefficients in (12) makes directions in principle learnable, which may explain why the undirected $\overline{A}$ (UndShift) performs reasonably well with most directed bases.

All methods perform best on the low frequency GenBC signals and worst on the low frequency SprFreq signals, even though it is based on the only real Fourier basis.

VI. Conclusion

At the high level, the paper is concerned with the challenge of making GSP methods on undirected graphs (and thus with orthogonal Fourier basis) work on directed graphs. The considered prior denoising method based on unrolling an optimization algorithm into a trainable network is well-suited as a case study since the training makes it highly adaptive and thus may better reveal inherent limitations. Our results show that direction matters, the importance of a shift related to the Fourier basis, and that the basis GenBC may be best suited for applications. The best variant is the practical contribution of this paper: a method for denoising low-frequency signals on directed graphs.

Acknowledgements

We would like to thank the authors of [6], [7], and [10] for providing us with the software for the corresponding papers.

References

[1] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph Signal Processing: Overview, Challenges, and Applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

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

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

[4] T. Beelen and P. Van Dooren, "Computational aspects of the Jordan canonical form," in *Reliable Numerical Computation*, Cox and Hammarling, Eds. Oxford: Clarendon, 1990, pp. 57–72.

[5] S. Furutani, T. Shibahara, M. Akiyama, K. Hato, and M. Aida, "Graph Signal Processing for Directed Graphs based on the Hermitian Laplacian," in *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2019, pp. 447–463.

[6] R. Shafipour, A. Khodabakhsh, G. Mateos, and E. Nikolova, "A Directed Graph Fourier Transform with Spread Frequency Components," *IEEE Trans. Signal Process.*, vol. 67, no. 4, pp. 946–960, 2019.

[7] J. Domingos and J. M. F. Moura, "Graph Fourier Transform: A Stable Approximation," *IEEE Trans. Signal Process.*, vol. 68, pp. 4422–4437, 2020.

[8] B. Seifert and M. Püschel, "Digraph signal processing with generalized boundary conditions," *IEEE Trans. Signal Process.*, vol. 69, pp. 1422–1437, 2021.

[9] A. G. Marques, S. Segarra, and G. Mateos, "Signal Processing on Directed Graphs: The Role of Edge Directionality When Processing and Learning From Network Data," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 99–116, 2020.

[10] S. Chen, Y. C. Eldar, and L. Zhao, "Graph Unrolling Networks: Interpretable Neural Networks for Graph Signal Denoising," *IEEE Trans. Signal Process.*, vol. 69, pp. 3699–3713, 2021.

[11] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory: Foundation and 1-D time," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3572–3585, 2008.

[12] ——, "Algebraic signal processing theory," *CoRR*, vol. abs/cs/0612077, 2006. [Online]. Available: http://arxiv.org/abs/cs/0612077

[13] S. Sardellitti, S. Barbarossa, and P. di Lorenzo, "On the Graph Fourier Transform for Directed Graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 796–811, 2017.

[14] R. Shafipour, A. Khodabakhsh, G. Mateos, and E. Nikolova, "Digraph Fourier Transform via Spectral Dispersion Minimization," in *Proc. Int. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2018, pp. 6284–6288.

[15] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. Int. Conf. Machine Learning (ICML)*, 2010, pp. 399–406.

[16] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, 2021.

[17] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent., (ICLR)*, 2015.