Learning to Identify Sources of Network Diffusion

Chang Ye and Gonzalo Mateos
Dept. of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA

***Abstract*—We propose a deep learning solution to the inverse problem of localizing sources of network diffusion. Invoking graph signal processing (GSP) fundamentals, the problem boils down to blind estimation of a diffusion filter and its sparse input signal encoding the source locations. While the observations are bilinear functions of the unknowns, a mild requirement on invertibility of the graph filter enables a convex reformulation that we solve via the alternating-direction method of multipliers (ADMM). We unroll and truncate the novel ADMM iterations, to arrive at a parameterized neural network architecture for Source Localization on Graphs (SLoG-Net), that we train in an end-to-end fashion using labeled data. This way we leverage inductive biases of a GSP model-based solution in a data-driven trainable parametric architecture, which is interpretable, parameter efficient, and offers controllable complexity during inference. Experiments with simulated data corroborate that SLoG-Net exhibits performance in par with the iterative ADMM baseline, while attaining significant (post-training) speedups.**

***Index Terms*—Graph signal processing, network diffusion, deep learning, blind deconvolution, algorithm unrolling.**

I. Introduction

We study the problem of localizing sources of network diffusion, which can be cast as one of *blind* graph filter identification [23], [26]. To fix ideas, suppose we observe P graph signals $\{\mathbf{y}_i\}_{i=1}^P$ that we model as outputs of some diffusion graph filter, i.e., a polynomial in the graph-shift operator of a known graph $\mathcal{G}$ [7], [15], [20]. The goal is to jointly identify the filter coefficients $\mathbf{h}$ and the input signals $\{\mathbf{x}_i\}_{i=1}^P$ that generated the network observations. This inverse problem broadens blind deconvolution of temporal or spatial signals to graph domains [1], [12], [24]. Since the resulting bilinear inverse problem is ill-posed, we assume that the inputs are sparse – a natural setting when few source nodes inject a signal that spreads through the network [23]. Applications of source localization on graphs include sensor-based environmental monitoring (where is the epicenter?), opinion formation in social networks (who started the rumor?), neural signal processing (which brain regions were activated post stimuli?), epidemiology (who is patient zero for the disease outbreak?), or disinformation campaigns (which accounts instilled fake news?). In this paper, we propose a novel *data-driven* deep learning (DL) solution to this source localization problem.

Prior art, proposed approach and contributions. Unlike most existing works dealing with source localization on graphs, e.g., [17], [21], [28], similar to [16], [23], [26] the advocated approach brings to bear the graph signal processing (GSP) toolbox [15]. A noteworthy GSP method was put forth in [23], which casts the (bilinear) blind graph filter identification task as a linear inverse problem in the "lifted" rank-one, row-sparse matrix $\mathbf{x}\mathbf{h}^\top$; see also [1], [13] for seminal blind deconvolution work via convex programming. While the rank and sparsity minimization algorithms in [19], [23] can successfully recover sparse inputs along with low-order graph filters, reliance on matrix lifting can hinder applicability to large graphs. Beyond this computational consideration, the overarching assumption of [23] is that the inputs $\{\mathbf{x}_i\}_{i=1}^P$ share a common support. Other works adopt probabilistic models of network diffusion, and resulting maximum-likelihood source estimators can only be optimal for particular (e.g., tree) graphs [17], or rendered scalable under restrictive dependency assumptions [6]. Relative to [10], [16], the proposed framework can accommodate signals defined on general undirected graphs and relies on a convex estimator of the sparse sources of diffusion, which here we favorably exploit to design a DL architecture as well as to generate training examples.

In this context, our starting point is the *model-based* blind graph filter identification formulation in [26]. A mild requirement on invertibility of the graph filter facilitates an efficient convex formulation for the multi-signal case with arbitrary supports (Section III); see also [24] for a time-domain precursor. While [26] focused on fundamental identifiability conditions and exact recovery guarantees, here we shift gears to algorithmic issues and develop a solver based on the alternating-directions method of multipliers (ADMM) [2, Ch. 3.4.4]. In Section IV we unroll and truntate the novel ADMM iterations [14], [25], to arrive at a parameterized neural network architecture for Source Localization on Graphs (SLoG-Net), that we train in an end-to-end fashion using labeled data. This way we leverage inductive biases of a GSP model-based solution in a data-driven trainable parametric architecture, which is interpretable, parameter efficient, and offers controllable complexity during inference. Experiments with simulated data corroborate that SLoG-Net exhibits performance in par with the iterative ADMM baseline, while attaining significant (post-training) speedups (Section V). These preliminary tests show promise and support the prospect of algorithm unrolling for learning from network data; see also [4], [18]. Concluding remarks are given in Section VI, which includes a discussion about future research direction in this space.

II. Preliminaries and Problem Statement

Consider a weighted and undirected network graph $\mathcal{G}(\mathcal{V}, \mathbf{A})$, where $\mathcal{V}$ is the set of vertices of cardinality $|\mathcal{V}| = N$, and $\mathbf{A} \in \mathbb{R}_+^{N\times N}$ is the symmetric adjacency matrix. Entries $A_{ij} = A_{ij} \geq 0$ denote the edge weight between nodes i and j. As a more general algebraic descriptor of network

This work was supported in part by the NSF Awards under Grants CCF-1750428, CCF-1934962, and ECCS-1809356. Author emails: {cye7,gmateosb}@ur.rochester.edu.

structure, one can define a *graph-shift operator* $\mathbf{S} \in \mathbb{R}^{N\times N}$ as a matrix with the same sparsity pattern as $\mathcal{G}$ [20]. Accordingly, $\mathbf{S}$ can be viewed as a local, meaning one-hop, diffusion (or averaging) operator. See [8], [15] for typical choices including normalized variations of adjacency and Laplacian matrices. Since $\mathbf{S}$ is symmetric, it is diagonalizable as $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, with $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$. Lastly, a graph signal $\mathbf{x} : \mathcal{V} \mapsto \mathbb{R}^N$ is an N-dimensional vector, where entry x_i represents the signal value at node $i \in \mathcal{V}$; see [15] for examples.

A. Graph filter models of network diffusion

Let $\mathbf{y}$ be a graph signal supported on $\mathcal{G}$, which is generated from an input state $\mathbf{x}$ via linear network diffusion, namely

$$\mathbf{y} \;=\; \alpha_0 \textstyle\prod_{l=1}^{\infty}(\mathbf{I}_N - \alpha_l\mathbf{S})\mathbf{x} \;=\; \sum_{l=0}^{\infty}\beta_l\mathbf{S}^l\mathbf{x}. \tag{1}$$

While $\mathbf{S}$ encodes only one-hop interactions, each successive application of the shift in (1) diffuses $\mathbf{x}$ over $\mathcal{G}$. This model is quite general and subsumes heat diffusion, consensus and the classic DeGroot model of opinion dynamics [5].

The diffusion expressions in (1) are polynomials on $\mathbf{S}$ of possibly infinite degree, yet the Cayley-Hamilton theorem asserts they are equivalent to polynomials of degree smaller than N. Upon defining the vector of coefficients $\mathbf{h} := [h_0, \ldots, h_{L-1}]^\top$ and the (convolutional) graph filter

$$\mathbf{H} := \sum_{l=0}^{L-1} h_l\mathbf{S}^l, \tag{2}$$

the signal model in (1) becomes $\mathbf{y} = \big(\sum_{l=0}^{L-1} h_l\mathbf{S}^l\big)\,\mathbf{x} := \mathbf{H}\mathbf{x}$, for some particular $\mathbf{h}$ and $L \leq N$. Due to the local structure of $\mathbf{S}$, graph filters represent linear transformations that can be implemented in a distributed fashion [22], e.g., via $L-1$ successive exchanges of information among neighbors in $\mathcal{G}$.

Leveraging the spectral decomposition of $\mathbf{S}$, graph filters and signals can be represented in the frequency domain. Specifically, let us use the eigenvalues of $\mathbf{S}$ to define the $N \times L$ Vandermonde matrix $\mathbf{\Psi}_L$, where $\Psi_{ij} := \lambda_i^{j-1}$. The frequency representations of a signal $\mathbf{x}$ and filter $\mathbf{h}$ are defined as $\tilde{\mathbf{x}} := \mathbf{V}^\top\mathbf{x}$ and $\tilde{\mathbf{h}} := \mathbf{\Psi}_L\mathbf{h}$, respectively. The latter follows since the output $\mathbf{y} = \mathbf{H}\mathbf{x}$ in the frequency domain is given by

$$\tilde{\mathbf{y}} = \text{diag}\big(\mathbf{\Psi}_L\mathbf{h}\big)\mathbf{V}^\top\mathbf{x} = \text{diag}\big(\tilde{\mathbf{h}}\big)\tilde{\mathbf{x}} = \tilde{\mathbf{h}} \circ \tilde{\mathbf{x}}. \tag{3}$$

This identity can be seen as a counterpart of the convolution theorem for temporal signals, where $\tilde{\mathbf{y}}$ is the elementwise product ($\circ$) of $\tilde{\mathbf{x}}$ and the filter's frequency response $\tilde{\mathbf{h}}$.

B. Problem statement

For given shift operator $\mathbf{S}$ and filter order L, suppose we observe P output signals collected in a matrix $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_P] \in \mathbb{R}^{N\times P}$ such that $\mathbf{Y} = \mathbf{H}\mathbf{X}$, where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{N\times P}$ is *sparse* having at most $S \ll N$ non-zero entries per column. The goal is to perform blind identification of the graph filter (and its input signals), which amounts to estimating sparse $\mathbf{X}$ and the filter coefficients $\mathbf{h}$ up to scaling and (possibly) permutation ambiguities [26]. Sparsity is well motivated when the signals in $\mathbf{Y}$ represent diffused versions of a few localized sources in $\mathcal{G}$, here indexed by $\text{supp}(\mathbf{X}) := \{(i,j) \mid X_{ij} \neq 0\}$. Moreover, the non-sparse formulation is ill-posed, since the number of unknowns $NP + L$ in $\{\mathbf{X}, \mathbf{h}\}$ exceeds the NP observations in $\mathbf{Y}$.

All in all, using (3) the diffused source localization task can be stated as a feasibility problem of the form

$$\text{find } \{\mathbf{X}, \mathbf{h}\} \text{ s. to } \mathbf{Y} = \mathbf{V}\text{diag}\big(\mathbf{\Psi}_L\mathbf{h}\big)\mathbf{V}^\top\mathbf{X},\ \|\mathbf{X}\|_0 \leq PS, \tag{4}$$

where the ℓ_0-(pseudo) norm $\|\mathbf{X}\|_0 := |\text{supp}(\mathbf{X})|$ counts the non-zero entries in $\mathbf{X}$. In words, we are after the solution to a system of bilinear equations subject to a sparsity constraint in $\mathbf{X}$; a hard problem due to the non-convex ℓ_0-norm as well as the bilinear constraints. To deal with the latter, similar to [24], [26] we will henceforth assume that the filter $\mathbf{H}$ is invertible.

Suppose that $\mathbf{X}$ is a realization drawn from some distribution of sparse matrices, say the Bernoulli-Gaussian model for which one can establish (4) is identifiable [26, Remark 1]. Likewise, suppose the filter taps $\mathbf{h}$ are drawn from a distribution such that $\mathbf{H}$ is invertible with high probability. Then given independent training samples $\mathcal{T} := \{\mathbf{X}_i, \mathbf{Y}_i\}_{i=1}^{|\mathcal{T}|}$ adhering to (1), our goal in this paper is to learn a judicious parametric mapping that predicts $\hat{\mathbf{X}} = \Phi(\mathbf{Y}; \mathbf{\Theta})$ by minimizing a loss function

$$L(\mathbf{\Theta}) := \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}} \ell(\mathbf{X}_i, \Phi(\mathbf{Y}_i; \mathbf{\Theta})), \tag{5}$$

where $\mathbf{\Theta}$ are learnable parameters. The particular choice of ℓ will be discussed in Section V.

III. Model-based Source Localization on Graphs

Here we review the model-based solution to the blind graph filter identification problem proposed in [26], which relies on a convex relaxation of (4) when the diffusion filter is invertible. Then we develop novel ADMM iterations to solve said relaxation, which we unroll in Section IV to obtain the SLoG-Net model that we train using data by minimizing (5).

A. Convex relaxation for invertible graph filters

Note from (3) that graph filter $\mathbf{H}$ is invertible if and only if $\tilde{h}_i = \sum_{l=0}^{L-1} h_l\lambda_i^l \neq 0$, for all $i = 1, \ldots, N$. In words, the frequency response of the filter should not vanish at the graph frequencies $\{\lambda_i\}$. In such case one can show that the inverse operator $\mathbf{G} := \mathbf{H}^{-1}$ is also a graph filter on $\mathcal{G}$, which can be uniquely represented as a polynomial in the shift $\mathbf{S}$ of degree at most $N-1$ [20, Theorem 4]. To be more specific, let $\mathbf{g} \in \mathbb{R}^N$ be the vector of inverse-filter coefficients, i.e., $\mathbf{G} = \sum_{l=0}^{N-1} g_l\mathbf{S}^l$. Then one can equivalently rewrite the generative model $\mathbf{Y} = \mathbf{H}\mathbf{X}$ for the observations as

$$\mathbf{X} = \mathbf{G}\mathbf{Y} = \mathbf{V}\text{diag}(\tilde{\mathbf{g}})\mathbf{V}^\top\mathbf{Y}, \tag{6}$$

where $\tilde{\mathbf{g}} := \mathbf{\Psi}_N\mathbf{g} \in \mathbb{R}^N$ is the inverse filter's frequency response and $\mathbf{\Psi}_N \in \mathbb{R}^{N\times N}$ is Vandermonde. Naturally, $\mathbf{G} = \mathbf{H}^{-1}$ implies the condition $\tilde{\mathbf{g}} \circ \tilde{\mathbf{h}} = \mathbf{1}_N$ on the frequency responses, where $\mathbf{1}_N$ denotes the $N \times 1$ vector of all ones. Leveraging (6), one can recast (4) as a *linear* inverse problem

$$\min_{\{\mathbf{X}, \tilde{\mathbf{g}}\}} \|\mathbf{X}\|_0, \text{ s. to } \mathbf{X} = \mathbf{V}\text{diag}(\tilde{\mathbf{g}})\mathbf{V}^\top\mathbf{Y},\ \mathbf{X} \neq \mathbf{0}. \tag{7}$$

The ℓ_0 norm in (7) makes the problem NP-hard to optimize. Over the last decade or so, convex-relaxation approaches to tackle sparsity minimization problems have enjoyed remarkable success, since they often entail no loss of optimality. Accordingly, we instead: (i) seek to minimize the ℓ_1-norm convex surrogate of the cardinality function, that is $\|\mathbf{X}\|_1 = \sum_{i,j}|X_{ij}|$; and (ii) express the filter in the graph spectral domain as in (6) to obtain the cost

$$\|\mathbf{X}\|_1 = \|\mathbf{G}\mathbf{Y}\|_1 = \|\mathbf{V}\text{diag}(\tilde{\mathbf{g}})\mathbf{V}^\top\mathbf{Y}\|_1 = \|(\mathbf{Y}^\top\mathbf{V}\odot\mathbf{V})\tilde{\mathbf{g}}\|_1,$$

where $\odot$ denotes the Khatri-Rao (i.e., columnwise Kronecker) product. This suggests solving the convex ℓ_1-synthesis problem (in this case a linear program), e.g., [27], namely

$$\widehat{\tilde{\mathbf{g}}} = \underset{\tilde{\mathbf{g}}\in\mathbb{R}^N}{\text{argmin}} \ \|(\mathbf{Y}^\top\mathbf{V}\odot\mathbf{V})\tilde{\mathbf{g}}\|_1, \quad \text{s. to } \ \mathbf{1}_N^\top\tilde{\mathbf{g}} = 1. \tag{8}$$

While the linear constraint in (8) avoids the trivial solution $\widehat{\tilde{\mathbf{g}}} = 0$, it also serves to fix the scale of the estimated filter. Once the frequency response $\widehat{\tilde{\mathbf{g}}}$ of the inverse filter is recovered, one can readily reconstruct the sources via $\text{vec}[\hat{\mathbf{X}}] = (\mathbf{Y}^\top\mathbf{V}\odot\mathbf{V})\tilde{\mathbf{g}}$ as well as the filter $\mathbf{H}$, if desired.

As a result, under the pragmatic assumption that the diffusion filter is invertible, one can readily use e.g., an off-the-shelf interior-point method or a specialized sparsity-minimization algorithm to solve (8) efficiently.

B. ADMM algorithm

Problem (8) can be solved using the ADMM. Let $\mathbf{x} = \text{vec}[\mathbf{X}] \in \mathbb{R}^{NP}$ and denote $\mathbf{Z} := \mathbf{Y}^\top\mathbf{V}\odot\mathbf{V}$. Using variable splitting, problem (8) can be equivalently written as

$$\min_{\{\mathbf{x},\tilde{\mathbf{g}}\}} \|\mathbf{x}\|_1, \ \text{ s. to } \ \mathbf{Z}\tilde{\mathbf{g}} - \mathbf{x} = \mathbf{0}_{NP}, \ \mathbf{1}_N^\top\tilde{\mathbf{g}} = c, \tag{9}$$

where $c = 1$, but will henceforth treat it as a generic constant in case we want to adjust the scale of $\tilde{\mathbf{g}}$. Associating dual variables $\boldsymbol{\lambda}$ and μ to the equality constraints in (9), the augmented Lagrangian function of the problem becomes

$$\mathcal{L}_\rho(\mathbf{x},\tilde{\mathbf{g}},\boldsymbol{\lambda},\mu) = \|\mathbf{x}\|_1 + \frac{\rho_\lambda}{2}\|\mathbf{Z}\tilde{\mathbf{g}} - \mathbf{x} + \boldsymbol{\lambda}/\rho_\lambda\|_2^2 + \frac{\rho_\mu}{2}(\mathbf{1}_N^\top\tilde{\mathbf{g}} - c + \mu/\rho_\mu)^2, \tag{10}$$

where ρ_λ and ρ_μ are non-negative penalty coefficients. Letting $\boldsymbol{\Gamma} := \rho_\lambda\mathbf{Z}^\top\mathbf{Z} + \rho_\mu\mathbf{1}_N\mathbf{1}_N^\top$ for notational convenience, then the ADMM [2], [3] update rules are given by ($k = 0, 1, 2, \ldots$ will henceforth denote iterations)

$$\tilde{\mathbf{g}}[k+1] = \boldsymbol{\Gamma}^{-1}\left[\mathbf{Z}^\top(\rho_\lambda\mathbf{x}[k] - \boldsymbol{\lambda}[k]) + (\rho_\mu c - \mu[k])\mathbf{1}_N\right], \tag{11}$$

$$\mathbf{x}[k+1] = \mathcal{S}_{\rho_\lambda^{-1}}(\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \boldsymbol{\lambda}[k]/\rho_\lambda), \tag{12}$$

$$\boldsymbol{\lambda}[k+1] = \boldsymbol{\lambda}[k] + \rho_\lambda(\mathbf{Z}\tilde{\mathbf{g}}[k+1] - \mathbf{x}[k+1]), \tag{13}$$

$$\mu[k+1] = \mu[k] + \eta_\mu(\mathbf{1}_N^\top\tilde{\mathbf{g}}[k+1] - c). \tag{14}$$

The soft-thresholding operator $S_{\rho_\lambda^{-1}}(\cdot)$ in (12) acts component-wise on the entries of its vector argument. Different from the solvers in [19], [23], the provably convergent ADMM updates are free of expensive singular-value decompositions per iteration. The inversion of the $N \times N$ matrix $\boldsymbol{\Gamma}$ is done once, and $\boldsymbol{\Gamma}^{-1}\mathbf{Z}^\top$, $\boldsymbol{\Gamma}^{-1}\mathbf{1}_N$ are cached to run the iterations.

In the next section, we unroll the ADMM iterations (11)-(14) to arrive at the trainable parametric model $\Phi(\mathbf{Y};\boldsymbol{\Theta})$.

IV. Localizing Sources via Algorithm Unrolling

The idea of algorithm unrolling was introduced in [9]. In the context of sparse coding, [9] advocated identifying *iterations* of proximal-gradient algorithms with *layers* in a deep network of fixed depth that can be trained from examples using backpropagation. One can view this process as effectively truncating the iterations of an asymptotically convergent procedure, to yield a template architecture that learns to approximate solutions with substantial computational savings relative to the optimization algorithm. Beyond parsimonious signal modeling, there has been a surge in popularity of unrolled deep networks for a wide variety of applications; see e.g., [14]. Most relevant to our approach is the unrolling of ADMM iterations for undersampled image reconstruction [25], and recent advances to learn from graph data [4], [18].

We construct the SLoG-Net architecture by unrolling the iterations (11)-(14) into a deep neural network. This entails mapping individual update rules as sub-layers within a layer, and stacking a prescribed number K of layers together to form $\Phi(\mathbf{Y};\boldsymbol{\Theta})$. The ADMM penalty coefficients $\{\rho_\lambda, \rho_\mu\}$ will be treated as learnable parameters in $\boldsymbol{\Theta}$. In designing SLoG-Net's sub-layers, we will introduce additional parameters to broaden the model's expressive power. We will also forgo the parameter sharing constraint imposed by the unrolled ADMM iterations.

Filter sub-layer. This sub-layer refines the inverse filter coefficient estimate $\tilde{\mathbf{g}}[k]$ at layer k, based on the source estimates $\mathbf{x}[k-1]$ and the dual variables $\{\boldsymbol{\lambda}[k-1], \mu[k-1]\}$ from the previous layer. We mimic the $\tilde{\mathbf{g}}$ update in (12), and introduce some minor tweaks. To circumvent problems with the inversion of $\boldsymbol{\Psi}$ in the eventuality $\rho_\lambda = \rho_\mu = 0$ during training, we introduce change of variables $\rho_1 := 1/\rho_\lambda$ and $\rho_2 := \rho_\mu/\rho_\lambda$ and impose non-negativity constraints on both parameters. Besides, we consider different parameters $\{\rho_1^{(k)}, \rho_2^{(k)}\}_{k=1}^K$ across layers to increase the network capacity, thus obtaining [cf. (11)]

$$\tilde{\mathbf{g}}[k+1] = (\mathbf{Z}^\top\mathbf{Z} + \rho_2^{(k)}\mathbf{1}_N\mathbf{1}_N^\top)^{-1}\Big[\mathbf{Z}^\top(\mathbf{x}[k] - \rho_1^{(k)}\boldsymbol{\lambda}[k]) + (\rho_2^{(k)}c - \rho_1^{(k)}\mu[k])\mathbf{1}_N\Big], \tag{15}$$

where $\rho_1^{(k)}, \rho_2^{(k)} \geq 0$, for $k = 1, \ldots, K$. Once training concludes, the matrix inverse and its products with $\mathbf{Z}^\top$ and $\mathbf{1}_N$ can be precomputed and cached for fast inference.

Sources sub-layer. Here we update the source estimates $\mathbf{x}[k]$ based on $\tilde{\mathbf{g}}[k]$ in (15) and the multiplier $\boldsymbol{\lambda}[k-1]$. The sub-layer imitates (13), but instead of a single tunable parameter ρ_λ we introduce learnable combination weights $\{\alpha_1^{(k)}, \alpha_2^{(k)}\}_{k=1}^K$ and thresholds $\{\tau^{(k)}\}_{k=1}^K$. We propose [cf. (12)]

$$\mathbf{x}[k+1] = \mathcal{S}_{\tau^{(k)}}\left(\alpha_1^{(k)}\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \alpha_2^{(k)}\boldsymbol{\lambda}[k]\right), \tag{16}$$

where the thresholds are naturally constrained as $\tau^{(k)} \geq 0$ for $k = 1, \ldots, K$. Notice how (16) implements a simple linear filter followed by a point-wise nonlinear activation, which is reminiscent of vanilla neural network layers.

Multiplier sub-layer. In this simple linear sub-layer, we perform parallel updates of the Lagrange multipliers $\{\boldsymbol{\lambda}[k], \mu[k]\}$ by combining $\{\boldsymbol{\lambda}[k-1], \mu[k-1]\}$ and the primal variable

inputs $\{\tilde{\mathbf{g}}[k], \mathbf{x}[k]\}$. The combination weights are learnable parameters $\{\beta_1^{(k)}, \beta_2^{(k)}, \beta_3^{(k)}\}_{k=1}^K$ and $\{\gamma_1^{(k)}, \gamma_2^{(k)}, \gamma_3^{(k)}\}_{k=1}^K$, resulting in [cf. (13)-(14)]

$$\boldsymbol{\lambda}[k+1] = \beta_1^{(k)}\boldsymbol{\lambda}[k] + \beta_2^{(k)}\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \beta_3^{(k)}\mathbf{x}[k+1], \quad (17)$$

$$\mu[k+1] = \gamma_1^{(k)}\mu[k] + \gamma_2^{(k)}\mathbf{1}_N^\top\tilde{\mathbf{g}}[k+1] + \gamma_3^{(k)}c. \quad (18)$$

In closing, we note that the intial states $\{\mathbf{x}[0], \boldsymbol{\lambda}[0], \mu[0]\}$ can be: (i) used as a means to incorporate prior information (especially on the source locations $\mathbf{x}$); (ii) randomly initialized as we do in the ensuing experiments; or (iii) learned from data along with $\boldsymbol{\Theta}$ as it is customary with recurrent neural networks (RNNs). Going all the way to layer K, source location predictions are generated as $\Phi(\mathbf{Y}, \boldsymbol{\Theta}) = \text{unvec}[(\mathbf{Y}^\top\mathbf{V} \odot \mathbf{V})\tilde{\mathbf{g}}[K]]$. Inspection of SLoG-Net's sub-layers leads to a parameter count of $|\boldsymbol{\Theta}| = 11 \times K$, independent of the problem dimensions N and P. Parameter efficiency is a well-documented feature of unrolled architectures [14]. Given a training set $\mathcal{T} := \{\mathbf{X}_i, \mathbf{Y}_i\}_{i=1}^{|\mathcal{T}|}$ of e.g., syntethic data, or, real signals $\mathbf{Y}_i$ and source estimates obtained using ADMM, learning is accomplished by using mini-batch stochastic gradient descent to minimize the loss function $L(\boldsymbol{\Theta})$ in (5). Further training details, including the specification of the loss, are outlined in the following numerical evaluation section.

V. Preliminary Numerical Experiments

We present preliminary numerical results on a source localization task, using simulated data and a random graph.

Synthetic data generation. The graph shift operator is selected as the normalized adjacency matrix $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{D} := \text{diag}(\mathbf{A}\mathbf{1}_N)$ is the diagonal matrix of node degrees and $\mathbf{A}$ is a realization of an Erdos-Renyi random graph with $N = 20$ and $p = 0.3$. For $|\mathcal{T}| = 64000$, we generate sparse $\mathbf{X} \in \mathbb{R}^{N\times|\mathcal{T}|}$ adhering to the Bernoulli-Gaussian model. Specifically, $\mathbf{X} = \boldsymbol{\Omega} \circ \mathbf{R}$, where $\boldsymbol{\Omega} \in \mathbb{R}^{N\times|\mathcal{T}|}$ is an i.i.d. Bernoulli matrix with parameter $\theta = 0.2$ (i.e., $\mathrm{P}(\Omega_{ij} = 1) = \theta$), and $\mathbf{R} \in \mathbb{R}^{N\times|\mathcal{T}|}$ is an independent random matrix with i.i.d. symmetric random variables drawn from a standard Gaussian distribution. Realizations of filter coefficients $\mathbf{h}$ are generated as $\mathbf{h} = (\mathbf{e}_1 + \alpha\mathbf{b})/\|\mathbf{e}_1 + \alpha\mathbf{b}\|_1$, where $\mathbf{e}_1 = [1, 0, \ldots, 0]^\top \in \mathbb{R}^L$ is the first canonical basis vector and entries of $\mathbf{b} \in \mathbb{R}^L$ are drawn independently from a standard Gaussian distribution. We have shown in [26] that recovery is harder for "less-impulsive" filters, so we focus on a challenging instance where $\alpha = 1$.

For each training epoch, the training samples in $\mathcal{T}$ are randomly split into $Q = 1600$ mini-batches of $P_b = 40$ signals, namely $\{\mathbf{X}_q\}_{q=1}^Q \in \mathbb{R}^{N\times P_b}$. We sample Q graph filter coefficients $\{\mathbf{h}_q\}_{q=1}^Q$ (with $L = 3$, $\alpha = 1$) and randomly assign them to the input signal mini-batches to generate the observations $\mathbf{Y}_q = \mathbf{V}\text{diag}(\boldsymbol{\Phi}_L\mathbf{h}_q)\mathbf{V}^\top\mathbf{X}_q$, $q = 1, \ldots, Q$. We use a validation set $\mathbf{X}_{\text{val}}$ of size $P_{\text{val}} = 0.01 \times |\mathcal{T}| = 640$, with observations $\mathbf{Y}_{\text{val}} = \mathbf{V}\text{diag}(\boldsymbol{\Psi}_L\mathbf{h}_{\text{val}})\mathbf{V}^\top\mathbf{X}_{\text{val}}$, where $\mathbf{h}_{\text{val}}$ is also generated from the same distribution as $\{\mathbf{h}_q\}_{q=1}^Q$.

Training details. We train SLoG-Net with $K = 5$ layers and use the normalized root mean square error (NRMSE) of $\mathbf{X}$ as loss function. Notice that if $\{\hat{\mathbf{X}}, \hat{\mathbf{h}}\}$ is a solution to the bilinear problem, then so is $\{-\hat{\mathbf{X}}, -\hat{\mathbf{h}}\}$ and accordingly we minimize

$$L(\boldsymbol{\Theta}) = \sum_{q=1}^{Q} \min\left(\frac{\|\Phi(\mathbf{Y}_q;\boldsymbol{\Theta}) - \mathbf{X}_q\|_F}{\|\mathbf{X}_q\|_F}, \frac{\|\Phi(\mathbf{Y}_q;\boldsymbol{\Theta}) + \mathbf{X}_q\|_F}{\|\mathbf{X}_q\|_F}\right)$$

using the Adam optimizer [11] implemented in PyTorch. We initialize $\{\rho_1^{(k)}, \rho_2^{(k)}, \tau^{(k)}\}_{k=1}^K$ as i.i.d. samples from the uniform distribution in $[0, 1]$, since these parameters are constrained to be non-negative. All other parameters in $\boldsymbol{\Theta}$ are randomly drawn from a standard Gaussian distribution.

We consider 30 epochs for training. In each epoch, we estimate the sparse sources $\{\Phi(\mathbf{Y}_q; \hat{\boldsymbol{\Phi}}_q)\}_{q=1}^Q$ using the training batches $\{\mathbf{Y}_q; \mathbf{X}_q\}_{q=1}^Q$. We choose one batch out of every 200 batches to compute the loss on the validation set $\{\mathbf{Y}_{\text{val}}; \mathbf{X}_{\text{val}}\}$ and record both the value of loss and the network parameters. In the end, we select the model $\hat{\boldsymbol{\Theta}}$ that has minimum validation loss across the entire training process.

Comparisons with the ADMM algorithm. For testing, we generate a test set $\{\mathbf{X}_{\text{test}}, \mathbf{h}_{\text{test}}\}$ where $N = 20, P_{\text{test}} = 100$. Fig. 1 depicts the diffused signals $\mathbf{Y}_{\text{test}} = \mathbf{V}\text{diag}(\boldsymbol{\Phi}_L\mathbf{h}_{\text{test}})\mathbf{V}^\top\mathbf{X}_{\text{test}}$ that were fed as inputs to the trained SLoG-Net model, the ground-truth sources $\mathbf{X}_{\text{test}}$ and the predictions $\Phi(\mathbf{Y}_{\text{test}}; \hat{\boldsymbol{\Theta}})$, as well as the recovered frequency response of the inverse filter $\hat{\tilde{\mathbf{g}}}_{\text{test}}$. Visual inspection confirms the predictions are quite accurate.

SLoG-Net is also compared with the model-based convex optimization approach in [26] using the ADMM solver developed in this paper. We consider two figures of merit to carry out the comparisons. Firstly, we consider the relative error (RE) given by $\|\Phi(\mathbf{Y}_{\text{test}}; \hat{\boldsymbol{\Theta}}) - \mathbf{X}_{\text{test}}\|_F / \|\mathbf{X}_{\text{test}}\|_F$. We also

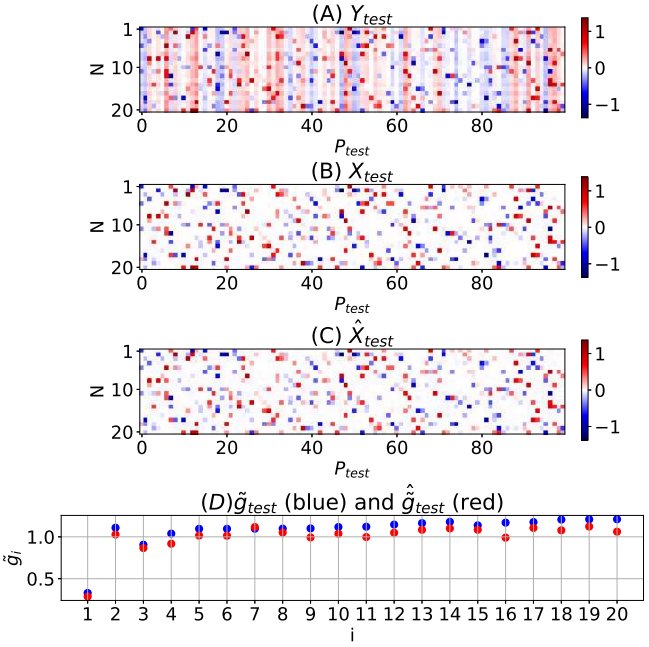

Fig. 1. Recovery performance for SLoG-Net with $K = 5$ layers and $P_{\text{test}} = 100$ observations. (A) Diffused signals $\mathbf{Y}_{\text{test}}$. (B) Ground-truth sparse sources $\mathbf{X}_{\text{test}}$. (C) SLoG-Net predictions $\Phi(\mathbf{Y}_{\text{test}}; \hat{\boldsymbol{\Theta}})$. (D) Frequency response of the inverse filter $\tilde{\mathbf{g}}_{\text{test}}$ (blue) and recovered inverse filter coefficient $\hat{\tilde{\mathbf{g}}}_{\text{test}}$ (red). The predictions are quite accurate. The relative error of the recovered $\mathbf{X}_{\text{test}}$ and $\tilde{\mathbf{g}}_{\text{test}}$ are 0.090 and 0.086, respectively.

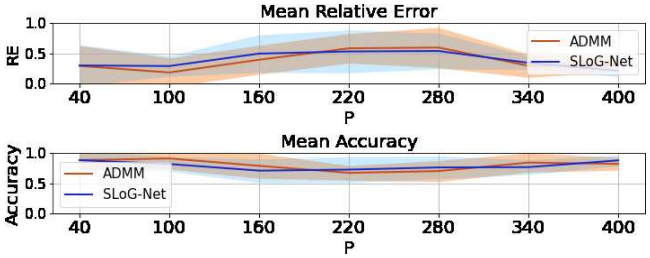

Fig. 2. Recovery performance of SLoG-Net ($K = 5$) and the model-based ADMM, or $\theta = 0.2$ and different values of P_{test}. (top) Mean relative error of recovered $\hat{\mathbf{X}}$ via SLoG-Net (blue) and ADMM (red). (Bottom) Mean accuracy in identifying the support of $\hat{\mathbf{X}}$ for a threshold $\kappa = 0.1$ via SLoG-Net (blue) and ADMM (red). The shaded region indicates the corresponding standard deviation. The mean elapsed time to form a prediction of the input signals (20 nodes $\times$ 100 observations) via a single forward of SLoG-Net and ADMM iterations are 0.005s and 0.067s, respectively.

compute the accuracy in recovering the support of $\mathbf{X}_{\text{test}}$, i.e., the source locations. To identify the support, we introduce a thresholding approach $\text{supp}_\kappa(\cdot)$ with threshold κ that if the entry of the recovered signal satisfies $|[\Phi(\mathbf{Y}_{\text{test}}; \hat{\mathbf{\Theta}})]_{ij}| \geq \kappa$, the index pair (i, j) will be considered a member of the estimated support. Accordingly, the recovered signal support is $\hat{\mathcal{I}}_{\text{test}} := \text{supp}_\kappa(\Phi(\mathbf{Y}_{\text{test}}; \hat{\mathbf{\Theta}}))$. We also apply the threshold to the ground-truth sources so the sought support set is $\mathcal{I}_{\text{test}} := \text{supp}_\kappa(\mathbf{X}_{\text{test}})$. Fig. 2 depicts the mean relative error (MRE) and mean accuracy of SLoG-Net and ADMM across 50 realizations, for $\theta = 0.2, \kappa = 0.1$ and different values of P_{test}. Apparently, the MRE and mean accuracy performance of SLoG-Net is on par with that ADMM. Notice that recovering the sources of network diffusion and the filter coefficients only requires a single forward pass through the neural network, while ADMM requires hundreds of iterations to converge. To reconstruct the test signal of size $N = 20, P = 100$, the mean elapsed time (over 50 realizations) is about 0.005s and 0.067s for SLoG-Net and ADMM, respectively. This (post-training) order-of-magnitude speed-up is likely to become more pronounced as the problem size grows.

VI. Conclusions and Future Work

We developed SLoG-Net, a novel deep learning approach to tackle the challenging problem of localizing sources of network diffusion. The unrolled architecture fruitfully leverages inductive biases stemming from model-based ADMM iterations, is parameter efficient, and can offer controllable complexity after training. Our promising preliminary results with simulated data demonstrate that SLoG-Net exhibits performance on par with an iterative ADMM baseline, while attaining order-of-magnitude speedups to generate predictions for source localization. We also observe SLoG-Net transfers well to problems with different number of observations from what was used during training. Ongoing work includes expanding our performance evaluation protocol to study robustness to noise, generalization and transfer to larger graphs, as well as tests with real brain, seismic, and epidemiological data.

References

[1] A. Ahmed, B. Recht, and J. Romberg, "Blind deconvolution using convex programming," *IEEE Trans. Inf. Theory*, vol. 60, no. 3, pp. 1711–1732, 2014.

[2] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, 2nd ed. Athena-Scientific, 1999.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[4] 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.

[5] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, pp. 118–121, 1974.

[6] S. Feizi, M. Médard, G. Quon, M. Kellis, and K. Duffy, "Network infusion to infer information sources in networks," *arXiv preprint arXiv:1606.07383 [cs.SI]*, 2016.

[7] F. Gama, E. Isufi, G. Leus, and A. Ribeiro, "Graphs, convolutions, and neural networks: From graph filters to graph neural networks," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 128–138, 2020.

[8] A. Gavili and X.-P. Zhang, "On the shift operator, graph frequency, and optimal filtering in graph signal processing," *IEEE Trans. Signal Process.*, vol. 65, no. 23, pp. 6303–6318, 2017.

[9] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," 2010, p. 399–406.

[10] C. Hu, X. Hua, J. Ying, P. M. Thompson, G. E. Fakhri, and Q. Li, "Localizing sources of brain disease progression with network diffusion model," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 7, pp. 1214–1225, 2016.

[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[12] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding blind deconvolution algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2354–2367, 2011.

[13] S. Ling and T. Strohmer, "Self-calibration and biconvex compressive sensing," *Inverse Problems*, vol. 31, no. 11, p. 115002, 2015.

[14] 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.

[15] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," vol. 106, no. 5, pp. 808–828, 2018.

[16] R. Pena, X. Bresson, and P. Vandergheynst, "Source localization on graphs via ℓ_1 recovery and spectral graph theory," in *Proc. IEEE Image, Video, and Multidimensional Signal Process. Workshop*, 2016, pp. 1–5.

[17] P. C. Pinto, P. Thiran, and M. Vetterli, "Locating the source of diffusion in large-scale networks," *Physical Review Letters*, vol. 109, no. 6, p. 068702, 2012.

[18] X. Pu, T. Cao, X. Zhang, X. Dong, and S. Chen, "Learning to learn graph topologies," in *Advances in Neural Information Processing Systems*, 2021.

[19] D. Ramírez, A. G. Marques, and S. Segarra, "Graph-signal reconstruction and blind deconvolution for diffused sparse inputs," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.*, Mar. 2017, pp. 4104–4108.

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

[21] E. Sefer and C. Kingsford, "Diffusion archeology for diffusion progression history reconstruction," *Knowledge and information systems*, vol. 49, no. 2, pp. 403–427, 2016.

[22] S. Segarra, A. G. 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, Aug 2017.

[23] S. Segarra, G. Mateos, A. G. Marques, and A. Ribeiro, "Blind identification of graph filters," *IEEE Trans. Signal Process.*, vol. 65, no. 5, pp. 1146–1159, Mar. 2017.

[24] L. Wang and Y. Chi, "Blind deconvolution from multiple sparse inputs," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1384–1388, 2016.

[25] Y. Yang, J. Sun, H. Li, and Z. Xu, "ADMM-CSNet: A deep learning approach for image compressive sensing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 3, pp. 521–538, 2020.

[26] C. Ye, R. Shafipour, and G. Mateos, "Blind identification of invertible graph filters with multiple sparse inputs," in *Proc. of European Signal Process. Conf.*, 2018, pp. 121–125.

[27] H. Zhang, M. Yan, and W. Yin, "One condition for solution uniqueness and robustness of both l1-synthesis and l1-analysis minimizations," *Adv. Comput. Math.*, vol. 42, no. 6, pp. 1381–1399, 2016.

[28] P. Zhang, J. He, G. Long, G. Huang, and C. Zhang, "Towards anomalous diffusion sources detection in a large network," *ACM T. Internet Techn.*, vol. 16, no. 1, p. 2, 2016.