ONLINE TOPOLOGY INFERENCE FROM STREAMING STATIONARY GRAPH SIGNALS

Rasoul Shafipour[†], *Abolfazl Hashemi*[*], *Gonzalo Mateos*[†] *and Haris Vikalo*[*]

[†]Dept. of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA
[*]Dept. of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX, USA

ABSTRACT

We address the problem of online topology inference from streaming nodal observations of graph signals generated by linear diffusion dynamics on the sought graph. To that end, we leverage the *stationarity* of the signals and use the so-called graph-shift operator (GSO) as a matrix representation of the graph. Under this model, estimated covariance eigenvectors obtained from streaming independent graph signals diffused on the sought network are a valid estimator of the GSO's *spectral templates*. We develop an ADMM algorithm to find a sparse and structurally admissible GSO given the eigenvectors estimate. Then, we propose an online scheme that upon sensing new diffused observations, efficiently updates eigenvectors (thus makes more accurate on expectation) and performs only one or a few iterations of the mentioned ADMM until the new data is observed. Numerical tests illustrate the effectiveness of the proposed topology inference approach in recovering large scale graphs, adapting to streaming information, and accommodating changes in the sought network.

Index Terms— Network topology inference, graph signal processing, diffusion process, online algorithm.

1. INTRODUCTION

Network data supported on the vertices of a graph $\mathcal{G}$ are becoming ubiquitous across disciplines spanning the bio-behavioral sciences and engineering. Such data, in a snapshot, can be thought of as graph signals represented by vectors indexed by the nodes of $\mathcal{G}$. In this context, the goal of graph signal processing (GSP) is to broaden the scope of traditional signal processing by developing algorithms that fruitfully exploit the relational structure of said signals [1]. However, the underlying graph is not readily available in many cases and the task would be to use observations of graph signals to learn the underlying network structure or a meaningful network model that facilitates signal representations and prediction tasks; see [2, 3] for a tutorial treatment.

To present the problem at hand, consider a weighted undirected graph $\mathcal{G}$, consisting of a node set $\mathcal{N}$ of cardinality N, and symmetric adjacency matrix $\mathbf{A}$ with entry $A_{ij} = A_{ji} \neq 0$ denoting the edge weight between node i and node j. Also, we assume that $\mathcal{G}$ contains no self-loop; i.e., $A_{ii} = 0$. Generally speaking, we can define a generic *graph-shift operator* (GSO) $\mathbf{S} \in \mathbb{R}^{N\times N}$ as any matrix capturing the same sparsity pattern as $\mathbf{A}$ on its non-diagonal entries [4]. Common choices for $\mathbf{S}$ are the adjacency $\mathbf{A}$, the Laplacian $\mathbf{L} := \text{diag}(\mathbf{A1}) - \mathbf{A}$, or their normalized counterparts [5].

In this paper, we present an online framework that estimates sparse graphs that explain the structure of streaming random signals. Particularly, in a snapshot, let $\mathbf{y} = [y_1, ..., y_N]^T \in \mathbb{R}^N$ be a zero-mean graph signal in which the ith element y_i denotes the signal value at node i of an *unknown graph* $\mathcal{G}$ with shift operator $\mathbf{S}$. Further consider a zero-mean white signal $\mathbf{x}$ with covariance matrix $\mathbf{C_x} = \mathbf{I}$ (identity matrix). We state that the graph $\mathbf{S}$ represents the structure of the signal $\mathbf{y} \in \mathbb{R}^N$ if there exists a diffusion process in the GSO $\mathbf{S}$ that produces the signal $\mathbf{y}$ from the input signal $\mathbf{x}$, that is

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

Under the assumption that $\mathbf{C_x} = \mathbf{I}$, (1) is equivalent to the *stationarity* of $\mathbf{y}$ in $\mathbf{S}$; see e.g., [6, Def. 1], [7], [8]. The justification to say that $\mathbf{S}$ represents the structure of $\mathbf{y}$ is that we can think of the edges of $\mathcal{G}$, i.e. the non-zero entries in $\mathbf{S}$, as direct (one-hop) relations between the elements of the signal. The diffusion described by (1) modifies the original correlation by inducing indirect (multi-hop) relations. In this context, our goal is to recover the fundamental relations dictated by $\mathbf{S}$ from a set of independent samples of streaming stationary random signals $\mathcal{Y} := \{\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(p)}, \mathbf{y}^{(p+1)}, \cdots\}$ that each of them adhere to linear diffusion dynamics as in (1). Assuming that time differences of the signals arrival is low relative to the time of fully running the topology learning algorithm, our goal is to derive simpler updates for each time step that can be used to estimate the GSO in an online fashion. Due to the stationarity assumption, true underlying covariance matrix of the observations should share the same eigenvectors as those of the sought GSO as elaborated in Section 2. Leveraging this result, our online inference entails two steps, where we: (i) update eigenvectors efficiently using methods described in Section 4; and (ii) take one or a few steps of a learning algorithm developed in Section 3. We further corroborate the effectiveness of the proposed approaches in offline batch setup as well as online tracking of the network in Section 5.

Relation to prior work. Workhorse topology inference approaches construct graphs whose edge weights correspond to nontrivial correlations between signals at incident nodes [9, 10]. Acknowledging that the observed correlations can be due to latent network effects, alternative statistical methods rely on inference of partial correlations [9, Ch. 7.3.2]. Under Gaussianity assumptions, this line of work has well-documented connections with covariance selection [11] and sparse precision matrix estimation [12–15], as well as high-dimensional sparse linear regression [16]. In the context of topology inference for undirected graphs, GSP-based frameworks postulate that the network exists as a latent underlying structure, and that observations are generated as a result of a network process defined in such a graph [17–22]. Different from [17, 19, 23, 24] that infer structure from signals assumed to be smooth over the sought undirected graph, here the measurements are assumed related to the graph via filtering. Few works have recently explored this approach by identifying a symmetric GSO given its eigenvectors, either assuming that the input is white [20, 21] – equivalently implying $\mathbf{y}$ is graph stationary [6, 25, 26]; or, colored [27, 28]. Here we address online topology inference from streaming stationary diffused signals; see e.g. [29] where instead the samples are from a graph process evolving according to a differential equation.

2. PROBLEM STATEMENT

To formally state the online topology inference problem, we consider the symmetric GSO $\mathbf{S}$ associated with the undirected graph $\mathcal{G}$. Upon defining the vector of coefficients $\mathbf{h} := [h_0, \ldots, h_{L-1}]^T \in \mathbb{R}^L$ and

Work in this paper was supported by the NSF award CCF-1750428.

the *symmetric* graph filter $\mathbf{H} := \sum_{l=0}^{L-1} h_l \mathbf{S}^l \in \mathbb{R}^{N\times N}$ [4], Cayley-Hamilton theorem asserts that the model in (1) boils down to

$$\mathbf{y} = \left(\sum_{l=0}^{L-1} h_l \mathbf{S}^l\right)\mathbf{x} = \mathbf{H}\mathbf{x}, \tag{2}$$

for some particular $\mathbf{h}$ and $L \leq N$. It is worth mentioning that L depends on the dependency range of the diffusion on the neighhbors.

Consider that streaming observations in $\mathcal{Y}$ correspond to independent realizations of a process adhering to the generative model in (2). The goal is to use $\mathcal{Y}$ to estimate the spectral templates $\mathbf{V}$ of the filter $\mathbf{H}$ that governs the diffusion in (2).

To gain insights, we first start with the offline setting, where $\mathbf{x}$ is white so that $\mathbf{C}_\mathbf{x} = \mathbf{I}$ [20]. Then the covariance matrix of $\mathbf{y} = \mathbf{H}\mathbf{x}$ is

$$\mathbf{C}_\mathbf{y} := \mathbb{E}[\mathbf{y}\mathbf{y}^T] = \mathbb{E}[\mathbf{H}\mathbf{x}(\mathbf{H}\mathbf{x})^T] = \mathbf{H}\mathbb{E}[\mathbf{x}\mathbf{x}^T]\mathbf{H} = \mathbf{H}^2. \tag{3}$$

In obtaining the third equality we used that $\mathbf{H}$ is symmetric, because it is a polynomial in the symmetric GSO $\mathbf{S}$. Using the spectral decomposition of $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ to express the filter as $\mathbf{H} = \sum_{l=0}^{L-1} h_l(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T)^l = \mathbf{V}(\sum_{l=0}^{L-1} h_l \mathbf{\Lambda}^l)\mathbf{V}^T$, we can diagonalize the covariance matrix as

$$\mathbf{C}_\mathbf{y} = \mathbf{V}\left(\sum_{l=0}^{L-1} h_l \mathbf{\Lambda}^l\right)^2 \mathbf{V}^T. \tag{4}$$

Such a covariance expression is the requirement for a graph signal to be stationary in $\mathbf{S}$ [6, Def. 2.b]. Remarkably, if $\mathbf{y}$ is graph stationary, or equivalently if $\mathbf{x}$ is white, (4) shows that the *eigenvectors* of the shift $\mathbf{S}$, the filter $\mathbf{H}$, and the covariance $\mathbf{C}_\mathbf{y}$ are *all the same*. As a result, to estimate $\mathbf{V}$ from the observations $\{\mathbf{y}^{(p)}\}_{p=1}^P$ it suffices to form the *sample covariance* $\hat{\mathbf{C}}_\mathbf{y} = \frac{1}{P}\sum_{p=1}^P \mathbf{y}^{(p)}(\mathbf{y}^{(p)})^T$ and use its eigenvectors as spectral templates to recover $\mathbf{S}$ [20,21]. Note that in estimating $\mathbf{C}_\mathbf{y}$, we assume that the observed signals are zero-mean without loss of generality, otherwise we can subtract the mean from the signals.

Given estimates $\hat{\mathbf{V}}$ of the covariance eigenvectors, recovery of $\mathbf{S}$ amounts to selecting its eigenvalues $\mathbf{\Lambda}$ and to that end we assume that the shift of interest is sparse. At the same time, we should account for the discrepancies between $\hat{\mathbf{V}}$ and the actual eigenvectors of $\mathbf{S}$, due to finite sample size constraints and unavoidable errors in estimating the filter. Accordingly, we build on [20] and seek for the shift operator $\mathbf{S}$ that: (a) is sparse, meaning that few edge weights are non-zero; (b) belongs to a convex set $\mathcal{S}$ that specifies the desired type of shift operator (e.g., the adjacency $\mathbf{A}$ or Laplacian $\mathbf{L}$); and (c) is close to $\hat{\mathbf{V}}\mathbf{\Lambda}\hat{\mathbf{V}}^T$ in the Frobenius-norm sense. One can thus solve

$$\mathbf{S}^* := \underset{\mathbf{\Lambda},\mathbf{S}\in\mathcal{S}}{\operatorname{argmin}} \ \|\mathbf{S}\|_1, \quad \text{s. to } \|\mathbf{S} - \hat{\mathbf{V}}\mathbf{\Lambda}\hat{\mathbf{V}}^T\|_F \leq \epsilon, \tag{5}$$

which is a convex optimization problem for the choice of a sparsity-promoting ℓ_1-norm criterion, and ϵ is a tuning parameter chosen based on a priori information on the imperfections.

The constraint $\mathbf{S} \in \mathcal{S}$ in (5) incorporates a priori knowledge about $\mathbf{S}$. If we let $\mathbf{S} = \mathbf{A}$ represent the adjacency matrix of an undirected graph with non-negative weights and no self-loops, we can explicitly write $\mathcal{S} = \mathcal{S}_\text{A}$ as follows

$$\mathcal{S}_\text{A} := \{\mathbf{S} \,|\, S_{ij} \geq 0, \ \mathbf{S}\in\mathcal{M}_N, \ S_{ii} = 0, \ \textstyle\sum_j S_{j1} = 1\}. \tag{6}$$

The first condition in $\mathcal{S}_\text{A}$ encodes the non-negativity of the weights whereas the second condition incorporates that $\mathcal{G}$ is undirected, hence, $\mathbf{S}$ must belong to the set $\mathcal{M}_N$ of real and symmetric $N\times N$ matrices. The third condition encodes the absence of self-loops, thus, each diagonal entry of $\mathbf{S}$ must be null. Finally, the last condition fixes the scale of the admissible graphs by setting the weighted degree of the first node to 1, and rules out the trivial solution $\mathbf{S} = \mathbf{0}$. Other GSOs (e.g., the Laplacian $\mathbf{L}$ and its normalized variants) can be accommodated via minor modifications to $\mathcal{S}$; see [20], but in this paper we focus on recovering adjacency matrices.

Going back to the online setting with the streaming stationary signals $\mathcal{Y} = \{\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(p)}, \mathbf{y}^{(p+1)}, \cdots\}$ arriving in a timely manner, the goal is to update the GSO in a computationally efficient way at each time step. We assume that in the online setting the time differences between signals arrival is much smaller than updating the GSO by eigendecomposition of updated $\hat{\mathbf{C}}_\mathbf{y}$ and completely solving (5) using off-the-shelf algorithms. Then our idea is to only take one or a few steps of an iterative algorithm for solving (5) upon sensing new diffused output signals until new signals are observed. In particular, upon arrival of new output signals, we first update the covariance $\hat{\mathbf{C}}_\mathbf{y}$ using a weighted re-averaging or resort to a cheaper approach to directly update the covariance eigenvectors $\hat{\mathbf{V}}$ using rank-1 perturbations of $\hat{\mathbf{C}}_\mathbf{y}$; see Section 4. Then, using the updated eigenvectors $\hat{\mathbf{V}}$, we take one or more iterations of the algorithm until we sense new outputs. In the sequel, we first derive an iterative algorithm in the offline batch setting to deal with (5) and then utilize it for the online setting.

3. TOPOLOGY INFERENCE VIA ADMM

As mentioned in the preceding section, in order to identify the underlying structure of the network, i.e. finding $\mathbf{S}$ and its eigenvalues $\mathbf{\Lambda}$, one needs to solve the optimization problem (5), which can equivalently be written as

$$\min_{\mathbf{S},\mathbf{\Lambda}} \ \|\mathbf{S} - \hat{\mathbf{V}}\mathbf{\Lambda}\hat{\mathbf{V}}^\top\|_F^2 + \lambda\|\mathbf{S}\|_1 \quad \text{s.t.} \quad \mathbf{S} \in \mathcal{S}_A, \tag{7}$$

where $\lambda > 0$ is an appropriately chosen regularization parameter. In this section we derive an Alternating Direction Method of Multipliers (ADMM) algorithm to solve (7). The algorithm (summarized as Algorithm 1) entails a block-coordinate descent procedure along with dual variable updates. Notice that since the problem in (7) is convex, this scheme will converge to a global minimizer [30].

To reformulate (7) in an amenable form for ADMM, define convex sets $\mathcal{C}_1 = \{\mathbf{M}|\mathbf{M} = \mathbf{M}^\top, \operatorname{diag}(\mathbf{M}) = \mathbf{0}\}$ and $\mathcal{C}_2 = \{\mathbf{M}|\mathbf{M} \geq \mathbf{0}, \sum_{i=1}^N M_{1i} = 1\}$ such that $\mathcal{S}_A = C_1 \cap \mathcal{C}_2$, introduce an auxiliary matrix $\mathbf{D}$, and consider the optimization

$$\begin{aligned} \min_{\mathbf{S},\mathbf{\Lambda},\mathbf{D}} \quad & \|\mathbf{S} - \hat{\mathbf{V}}\mathbf{\Lambda}\hat{\mathbf{V}}^\top\|_F^2 + \lambda\|\mathbf{S}\|_1 \\ \text{s.t.} \quad & \mathbf{D} \in \mathcal{C}_1 \cap \mathcal{C}_2, \quad \mathbf{S} - \mathbf{D} = \mathbf{0}, \end{aligned} \tag{8}$$

which is equivalent to the optimization problem (7). Form the augmented Lagrangian of (8) to obtain

$$\begin{aligned} \mathcal{L}_{\rho_1}(\mathbf{S}, \mathbf{D}, \mathbf{U}_1, \mathbf{\Lambda}) = & \|\mathbf{S} - \hat{\mathbf{V}}\mathbf{\Lambda}\hat{\mathbf{V}}^\top\|_F^2 + \lambda\|\mathbf{S}\|_1 \\ & + \frac{\rho_1}{2}\|\mathbf{S} - \mathbf{D} + \mathbf{U}_1\|_F^2, \end{aligned} \tag{9}$$

where $\rho_1 > 0$ and $\mathbf{U}_1$ are the so-called penalty parameter and scaled dual variable, respectively. At the k^{th} iteration, let $\mathbf{B}^{(k)} = \hat{\mathbf{V}}\mathbf{\Lambda}^{(k)}\hat{\mathbf{V}}^\top$. Then the ADMM consists of the following iterative steps to optimize (8):

Step 1. $\mathbf{S}^{(k+1)} = \operatorname{argmin}_\mathbf{S} \mathcal{L}_{\rho_1}(\mathbf{S}, \mathbf{D}^{(k)}, \mathbf{U}_1^{(k)}, \mathbf{\Lambda}^{(k)})$, which has a closed-form solution that can be expressed as

$$\mathbf{S}^{(k+1)} = \mathcal{T}_{\frac{\lambda}{2+\rho_1}}\left(\frac{\mathbf{B}^{(k)} + \frac{\rho_1}{2}(\mathbf{D}^{(k)} - \mathbf{U}_1^{(k)})}{1 + \frac{\rho_1}{2}}\right), \tag{10}$$

where $\mathcal{T}_\eta(x) = (|x| - \eta)_+ \operatorname{sgn}(x)$ is the so-called soft-thresholding operator that acts on each element of a given matrix.

Step 2. $\mathbf{D}^{(k+1)} = \operatorname{argmin}_{\mathbf{D}\in\mathcal{C}_1\cap\mathcal{C}_2} \mathcal{L}_{\rho_1}(\mathbf{S}^{(k+1)}, \mathbf{D}, \mathbf{U}_1^{(k)}, \mathbf{\Lambda}^{(k)})$.

To update $\mathbf{D}^{(k+1)}$ one needs to solve a constrained convex program. However, since projection onto $\mathcal{C}_1 \cap \mathcal{C}_2$ is not straightforward, we propose establishing an *inner* ADMM. More specifically, consider the optimization problem

$$\begin{aligned} \min_{\mathbf{E},\mathbf{Z}} \quad & \|\mathbf{E} - (\mathbf{S}^{(k+1)} + \mathbf{U}_1^{(k)})\|_F^2 + g_1(\mathbf{E}) + g_2(\mathbf{Z}) \\ \text{s.t.} \quad & \mathbf{E} - \mathbf{Z} = \mathbf{0}, \end{aligned} \tag{11}$$

Algorithm 1 Topology inference using ADMM

```
1: Input: estimated covariance eigenvectors $\hat{\mathbf{V}}$, penalty parameter $\rho_1$, regularization parameter $\lambda$, number of iterations $T_1$
2: Initialize: $\mathbf{\Lambda}^{(0)} = \text{diag}(\mathbf{1})$, $\mathbf{D}^{(0)} = \mathbf{0}$, $\mathbf{U}_1^{(0)} = \mathbf{0}$.
3: for $k = 0, \dots, T_1 - 1$
4:     $\mathbf{B}^{(k)} = \hat{\mathbf{V}}\mathbf{\Lambda}^{(k)}\hat{\mathbf{V}}^\top$
5:     $\mathbf{S}^{(k+1)} = \mathcal{T}_{\frac{\lambda}{2+\rho_1}}\left(\frac{\mathbf{B}^{(k)}+\frac{\rho_1}{2}(\mathbf{D}^{(k)}-\mathbf{U}_1^{(k)})}{1+\frac{\rho_1}{2}}\right)$
6:     Update $\mathbf{D}^{(k+1)}$ using Algorithm 2
7:     $\mathbf{\Lambda}^{(k+1)} = \text{diag}(\hat{\mathbf{V}}^\top \mathbf{S}^{(k+1)}\hat{\mathbf{V}})$
8:     $\mathbf{U}_1^{(k+1)} = \mathbf{U}_1^{(k)} + \mathbf{S}^{(k+1)} - \mathbf{D}^{(k+1)}$
9: end for
10: return $\mathbf{S}^{(T_1)}$ and $\mathbf{\Lambda}^{(T_1)}$
```

where $g_1(.)$ and $g_2(.)$ are indicator functions associated with convex sets $\mathcal{C}_1$ and C_2, respectively. The indicator function (e.g., $g_1(.)$) is zero if its input belongs to the corresponding set (e.g., $\mathcal{C}_1$), and is $+\infty$, otherwise.

Essentially in (11) we attempt to project $\mathbf{S}^{(k+1)} + \mathbf{U}_1^{(k)}$ onto the intersection of $\mathcal{C}_1$ and $\mathcal{C}_2$. We propose to solve (11) iteratively using an ADMM with T_2 number of iterations in order to find the update $\mathbf{D}^{(k+1)} = \mathbf{E}^{(T_2)}$. Consider the augmented Lagrangian of (11) for $\rho_2 > 0$ and $\mathbf{U}_2$ as the penalty parameter and the scaled dual variable for the inner ADMM, respectively. Following the ADMM procedure, the update rules for updating $\mathbf{E}$, $\mathbf{Z}$, and $\mathbf{U}_2$ are tabulated under Algorithm 2 which entails two projection operators. The first one denoted by $\mathcal{P}_1(.)$ is the projection onto the convex set C_1 defined as

$$\mathcal{P}_1(\mathbf{M}) = \frac{\mathbf{M} + \mathbf{M}^\top}{2} - \text{diag}(\mathbf{M}). \tag{12}$$

The second projection operator, $\mathcal{P}_2(.)$, is the projection onto the convex set C_2 and acts an each row of a given matrix. Denoting the row i of $\mathbf{\Omega}$ $(\mathbf{M})$ as ω_i $(\mathbf{m}_i)$, $\mathbf{\Omega} := \mathcal{P}_2(\mathbf{M})$ has the entries of the form

$$\begin{cases} \Omega_{11} := 0 & \\ \omega_i := \max(\mathbf{m}_i, \mathbf{0}_N) & i \neq 1 \\ [\Omega_{i2}, \dots, \Omega_{iN}] := \triangle^{N-1}([M_{i2}, \dots, M_{iN}]) & i = 1, \end{cases} \tag{13}$$

where $\mathbf{0}_N$ is the N-dimensional vector of zeros, max(.) is element-wise maximum, and $\triangle^{N-1}(.)$ is projection onto the $N-1$ dimensional probability simplex which can be efficiently computed by the method in [31].

Algorithm 2 The Inner ADMM to update $\mathbf{D}^{(k+1)}$

```
1: Input: penalty parameter $\rho_2$, number of iterations $T_2$.
2: Initialize: $\mathbf{E}^{(0)} = \mathbf{0}$, $\mathbf{Z}^{(0)} = \mathbf{0}$, $\mathbf{U}_2^{(0)} = \mathbf{0}$.
3: for $i = 0, \dots, T_2 - 1$
4:     $\mathbf{E}^{(i+1)} = \mathcal{P}_1\left(\frac{\mathbf{S}^{(k+1)}+\mathbf{U}_1^{(k)}+\frac{\rho_2}{2}(\mathbf{Z}^{(i)}-\mathbf{U}_2^{(i)})}{1+\frac{\rho_2}{2}}\right)$
5:     $\mathbf{Z}^{(i+1)} = \mathcal{P}_2(\mathbf{E}^{(i+1)} + \mathbf{U}_2^{(i)})$
6:     $\mathbf{U}_2^{(i+1)} = \mathbf{U}_2^{(i)} + \mathbf{E}^{(i+1)} - \mathbf{Z}^{(i+1)}$
7: end for
8: return $\mathbf{D}^{(k+1)} := \mathbf{E}^{(T_2)}$
```

After T_2 iterations of the inner ADMM we update $\mathbf{D}^{(k+1)}$ according to $\mathbf{D}^{(k+1)} = \mathbf{E}^{(T_2)}$.

Step 3. $\mathbf{\Lambda}^{(k+1)} = \text{argmin}_{\mathbf{\Lambda}}\, \mathcal{L}_{\rho_1}(\mathbf{S}^{(k+1)}, \mathbf{D}^{(k+1)}, \mathbf{U}_1^{(k)}, \mathbf{\Lambda})$ which takes the form

$$\mathbf{\Lambda}^{(k+1)} = \text{diag}(\hat{\mathbf{V}}^\top \mathbf{S}^{(k+1)} \hat{\mathbf{V}}). \tag{14}$$

Step 4. Finally, we update $\mathbf{U}_1$ according to

$$\mathbf{U}_1^{(k+1)} = \mathbf{U}_1^{(k)} + \mathbf{S}^{(k+1)} - \mathbf{D}^{(k+1)}, \tag{15}$$

which is a dual gradient ascent update.

The summarized Algorithm 1 converges to the global optimum of (7) for a properly chosen T_1 and T_2. In the next section, we expand on the online topology inference from streaming signals using the developed iterative algorithms which is capable of adapting to large scale graphs as well as inferring the networks in real time; see Section 5 for numerical demonstrations.

4. ONLINE TOPOLOGY INFERENCE

In the online setting where the observations are being streamed, an important first step prior to topology inference is to efficiently find updated eigenvectors of $\hat{\mathbf{C}}_{\mathbf{y}}$ without the need of employing eigenvalue decomposition to find the eigenvectors after receiving new observations. Here we show how $\hat{\mathbf{V}}$ can be updated efficiently with $\mathcal{O}(N^2)$ complexity without using eigenvalue decomposition.

Let $\hat{\mathbf{C}}_{\mathbf{y}}^{(P)}$ denote the covariance after receiving P streaming observations. Then, the new covariance after receiving $\mathbf{y}^{(P+1)}$ takes the form

$$\hat{\mathbf{C}}_{\mathbf{y}}^{(P+1)} = \frac{1}{P+1}(P\hat{\mathbf{C}}_{\mathbf{y}}^{(P)} + \mathbf{y}^{(P+1)}\mathbf{y}^{(P+1)}). \tag{16}$$

Therefore, receiving a new observation can be thought of as rank-one update of the estimated covariance matrix $\hat{\mathbf{C}}_{\mathbf{y}}^{(P)}$. According to [32], the eigenvalues of the rank-one modification of a symmetric matrix with known eigenvalues can be efficiently computed by solving the characteristic equation

$$1 + \sum_{j=1}^{N} \frac{\mathbf{z}_j^2}{Pd_j - \gamma} = 0, \tag{17}$$

where $\mathbf{z} = \hat{\mathbf{V}}^\top \mathbf{y}^{(p+1)}$, $\{d_j\}_{j=1}^N$ denote the eigenvalues of $\hat{\mathbf{C}}_{\mathbf{y}}^{(P)}$, and roots γ would be the eigenvalues of the updated covariance. The solution to (17) can be found using the Newton method [33] with $\mathcal{O}(N^2)$ complexity. For the updated eigenvalue γ_j, the corresponding eigenvector $\mathbf{v}_j$is given by

$$\mathbf{v}_j = q_j \mathbf{y}^{(p+1)} \circ \mathbf{q}_j \tag{18}$$

where q_j is a normalizing factor that ensures $\|\mathbf{v}_j\|_2 = 1$, $\mathbf{q}_j = [1/(Pd_1 - \gamma_j), \dots, 1/(Pd_N - \gamma_j)]$, and $\circ$ denotes the Hadamard product. Therefore, the updated eigenvectors for each newly observed signal can be computed with $\mathcal{O}(N^2)$ compared to $\mathcal{O}(N^3)$ complexity of the direct eigendecomposition.

Consolidating all the prerequisites, our online topology inference upon sensing new observations would entail two iterative step: (i) Update eigenvectors $\hat{\mathbf{V}}$; and (ii) Take one or a few steps of Algorithm 1 until new data is received. These two steps are performed in $\mathcal{O}(N^3)$ and at each step can efficiently output an estimate for the sought GSO. Note that if the signals arrive faster, one can create a buffer and perform each iteration of Algorithm 1 on a $\hat{\mathbf{V}}$ updated by a larger number of newly observed signals. On the other hand, for a slower rate of arrivals, increasing T_1 and T_2 accordingly would favor the performance.

5. NUMERICAL RESULTS

Here we assess the performance of the proposed algorithms in recovering sparse synthetic and real-world graphs. To that end, we: (i) illustrate the scalability of Algorithm 1 using a large scale random graph; (ii) evaluate the performance of the proposed online scheme in a setting with streaming signals; and (iii) demonstrate the effectiveness of the same approach in adapting to dynamical behavior of the network.

Throughout this section, we infer synthetic or real-world networks from the observation of diffusion processes that are synthetically generated via graph filtering as in (2). For the graph shift $\mathbf{S} = \mathbf{A}$, the adjacency matrix of the sought network, we consider a

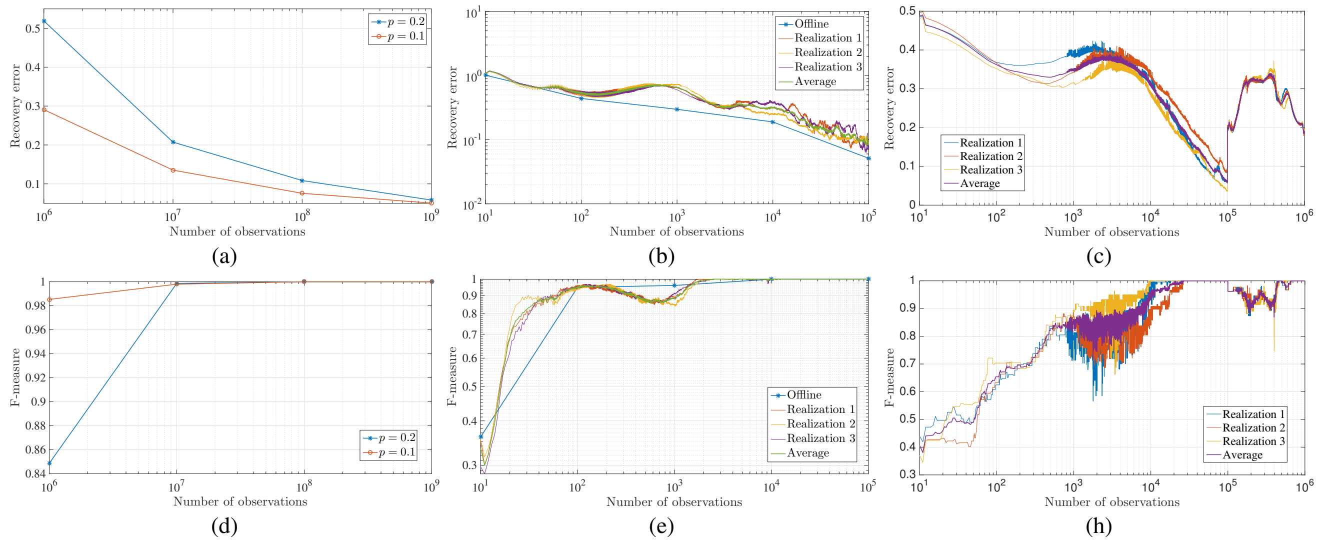

Fig. 1: Top: (a) Recovery error of the proposed scheme versus number of observations for an Erdős-Rényi (ER) graph with $N = 1000$ nodes and different degrees of connectivities (p). (b) Evolution of recovery error in inferring a brain network by performing one step of the proposed online algorithm upon sensing each new signal, superimposed with the offline batch counterpart. (c) Online recovery error versus the number of observed streaming signals for an ER graph with $N = 20$ nodes, where the structure of the network changes after sensing the 10^5th signal. Again, one iteration of the online scheme is performed for each newly observed signal. Bottom (d–h): Counterparts of the top figures (a–c, respectively) but for the F-measure. Similar patterns can be obtained using F-measure which shows the algorithm success in recovering the sparsity pattern in both online and batch offline settings.

second-order filter $\mathbf{H} = \sum_{l=0}^{2} h_l \mathbf{S}^l$, where the coefficients $\{h_l\}$ are drawn uniformly from $[0, 1]$.

As a measure of recovery error, we adopt $\|\mathbf{S}^* - \mathbf{S}\|_F / \|\mathbf{S}\|_F$, where $\mathbf{S}^*$ is the estimated GSO using our proposed approaches and $\mathbf{S}$ denotes the ground-truth GSO. To assess the edge-support recovery, we also compute the F-measure defined as the harmonic mean of edge precision and recall (precision is the percentage of correct edges in $\mathbf{S}^*$, and recall is the fraction of edges in $\mathbf{S}$ that are successfully retrieved).

Offline: Synthetic graph. To evaluate the scalability of Algorithm 1, consider an Erdős-Rényi (ER) graph with $N = 1000$, where edges are formed independently with probability $p = 0.2$. Here we assume that we observe all the P observations, and perform Algorithm 1 in an offline batch fashion. To that end, we apply Algorithm 1 on noisy eigenvectors $\hat{\mathbf{V}}$ of sample covariances of synthetic signals $\{\mathbf{y}^{(p)}\}_{p=1}^{P}$ generated through diffusion process $\mathbf{H}$. The entries of the inputs $\{\mathbf{x}^{(p)}\}_{p=1}^{P}$ are drawn independently from the normal Gaussian distribution ($\mathbf{C_x} = \mathbf{I}$) to make the observations stationary. Fig. 1-a(d) plots the recovery error (F-measure) averaged over 10 experiments as a function of the number of observed signals P. As the number of observations increase, the estimate $\hat{\mathbf{V}}$ becomes more reliable which leads to a better performance (i.e., larger F-measure and lower recovery error) of the underlying GSO. As predicted, for sparser graphs (smaller p) performance enhances, since our algorithm is tailored for recovering sparse graphs. Moreover, the illustrated results corroborate the effectiveness of Algorithm 1 in recovering large scale graphs.

Online: Brain graph. Consider a brain graph $\mathcal{G}$ with $N = 66$ nodes or neural regions and edge weights given by the density of anatomical connections between regions [34]. We generate streaming signals $\{\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(p)}, \mathbf{y}^{(p+1)}, \cdots\}$ by diffusing inputs $\{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(p)}, \mathbf{x}^{(p+1)}, \cdots\}$ through filter $\mathbf{H}$, where the inputs are formed similarly to the ones used for previous experiment. Upon sensing an observation $\mathbf{y}^{(p)}$, we first update sample covariance eigenvectors $\hat{\mathbf{V}}$ using the procedure described in Section 4 and then run Algorithm 1 only for one iteration (i.e., $T_1 = 1$) and ten inner iterations of Algorithm 2 (i.e., $T_2 = 10$). This is under the assumption that differences between the arrival times of signals are longer than one step of our pipeline. In Fig. 1-b(e), we depict the evolution of recovery error (F-measure) averaged over 10 instances as well as three realizations. We further plot the (average) offline behavior similar to the previous experiment in order to gauge the loss of online estimation. We notice that the proposed online scheme can successfully track the performance of the offline counterpart on expectation. The expected fluctuations are due to the nature of ADMM and the online scheme.

Online: Synthetic perurbations. Finally, we consider an Erdős-Rényi (ER) graph with $N = 20$ where edges are formed independently with probability $p = 0.2$. We generate the streaming signals similar to the previous online experiment; however, after observing 10^5 realizations, we remove 10% of the existing edges and add the same number of edges elsewhere. This would affect the output covariances accordingly. To examine the tracking capability of the online estimator, we run one iteration of Algorithm 1 with $T_2 = 10$ upon arrival of each signal and in Fig. 1-c(f) plot the recovery error (F-measure) averaged over 10 instances as well as three realizations. We observe that after $P = 10^5$, the performance deteriorates at first due to the sudden change of the network structure, but after observing large enough number of new samples, the online algorithm can adapt and learn the new structure as well. This demonstrates the effectiveness of the put forth online algorithm to adapt after perturbations.

6. CONCLUSION

We studied the inference of an undirected network from streaming observations of *stationary* signals diffused on an unknown graph. We developed an iterative algorithm to find sparse representations explaining the diffused signals. This was done by leveraging the sought eigenvectors of GSO preserved by the covariance of the observations. Then, our online scheme updated the eigenvectors upon sensing new signals and would take one or a few iterations of the developed learning algorithm until new data was received. The overall procedure was validated on streaming stationary signals supported on synthetic and real-world graphs.

7. REFERENCES

[1] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre 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.

[2] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 44–63, May 2019.

[3] Gonzalo Mateos, Santiago Segarra, Antonio G Marques, and Alejandro Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 16–43, May 2019.

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

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

[6] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Trans. Signal Process.*, vol. 65, no. 22, pp. 5911–5926, Aug. 2017.

[7] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3462–3477, July 2017.

[8] B. Girault, "Stationary graph signals using an isometric graph translation," in *European Signal Process. Conf. (EUSIPCO)*, Aug 2015, pp. 1516–1520.

[9] E. D. Kolaczyk, *Statistical Analysis of Network Data: Methods and Models*, Springer, New York, NY, 2009.

[10] O. Sporns, *Discovering the Human Connectome*, MIT Press, Boston, MA, 2012.

[11] A. P. Dempster, "Covariance selection," *Biometrics*, vol. 28, no. 1, pp. 157–175, 1972.

[12] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[13] B. M. Lake and J. B. Tenenbaum, "Discovering structure by learning sparse graph," in *Annual Cognitive Sc. Conf.*, 2010, pp. 778 – 783.

[14] M. Slawski and M. Hein, "Estimation of positive definite M-matrices and structure learning for attractive Gaussian Markov random fields," *Linear Algebra and its Applications*, vol. 473, pp. 145–179, 2015.

[15] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825–841, Sept. 2017.

[16] N. Meinshausen and P. Buhlmann, "High-dimensional graphs and variable selection with the lasso," *Ann. Stat.*, vol. 34, pp. 1436–1462, 2006.

[17] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, Dec 2016.

[18] J. Mei and J. M. F. Moura, "Signal processing on graphs: Causal modeling of unstructured data," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 2077–2092, Apr. 2017.

[19] V. Kalofolias, "How to learn a graph from smooth signals," in *Intl. Conf. Artif. Intel. Stat. (AISTATS)*. J Mach. Learn. Res., 2016, pp. 920–929.

[20] S. Segarra, A.G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 467–483, Aug. 2017.

[21] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Trans. Signal Inf. Process. Netw.*, vol. PP, no. 99, pp. 1–1, 2017.

[22] D. Thanou, X. Dong, D. Kressner, and P. Frossard, "Learning heat diffusion graphs," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 484–499, Sept 2017.

[23] S. P. Chepuri, S. Liu, G. Leus, and A. O. Hero, "Learning sparse graphs under smoothness prior," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, New Orleans, LA, Mar. 5-9, 2017, pp. 6508–6512.

[24] M. G. Rabbat, "Inferring sparse graphs from smooth signals with theoretical guarantees," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, New Orleans, LA, Mar. 5-9, 2017, pp. 6533–6537.

[25] B. Girault, "Stationary graph signals using an isometric graph translation," in *European Signal Process. Conf. (EUSIPCO)*, Aug. 2015, pp. 1516–1520.

[26] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3462–3477, Jul. 2017.

[27] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Identifying the topology of undirected networks from diffused nonstationary graph signals," *IEEE Trans. Signal Process.*, 2018, (submitted; see also arXiv:1801.03862 [eess.SP]).

[28] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Network topology inference from non-stationary graph signals," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, New Orleans, LA, Mar. 5-9, 2017.

[29] Stefan Vlaski, Hermina P Maretić, Roula Nassif, Pascal Frossard, and Ali H Sayed, "Online graph learning from sequential data," in *2018 IEEE Data Science Workshop (DSW)*. IEEE, 2018, pp. 190–194.

[30] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan 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.

[31] Yunmei Chen and Xiaojing Ye, "Projection onto a simplex," *arXiv preprint arXiv:1101.6081*, 2011.

[32] James R Bunch, Christopher P Nielsen, and Danny C Sorensen, "Rank-one modification of the symmetric eigenproblem," *Numerische Mathematik*, vol. 31, no. 1, pp. 31–48, 1978.

[33] Ming Gu and Stanley C Eisenstat, "A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem," *SIAM journal on Matrix Analysis and Applications*, vol. 15, no. 4, pp. 1266–1276, 1994.

[34] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J Wedeen, and O. Sporns, "Mapping the structural core of human cerebral cortex," *PLoS Biol*, vol. 6, no. 7, pp. e159, 2008.