TIME-VARYING GRAPH LEARNING BASED ON SPARSENESS OF TEMPORAL VARIATION

Koki Yamada[1]*, Yuichi Tanaka*[1,2]*, and Antonio Ortega*[3]

[1]Graduate School of BASE, Tokyo University of Agriculture and Technology, Tokyo, Japan
[2]PRESTO, Japan Science and Technology Agency, Saitama, Japan
[3]Dept. of Electrical and Computer Engineering, University of Southern California, Los Angeles, USA

ABSTRACT

We propose a method for graph learning from spatiotemporal measurements. We aim at inferring time-varying graphs under the assumption that changes in graph topology and weights are sparse in time. The problem is formulated as a convex optimization problem to impose a constraint on the temporal relation of the time-varying graph. Experimental results with synthetic data show the effectiveness of our proposed method.

***Index Terms*—** Graph learning, time-varying graph, time-varying network, network inference, dynamic graph,

1. INTRODUCTION

Many signals in various networks, e.g., social, brain, traffic, electrical, and sensor networks, exhibit structure that can be represented as a graph. A graph, consisting of sets of nodes and edges, enables us to efficiently analyze data by utilizing such structural information [1–7]. However, in many cases graphs are not given *a priori*; hence, graph learning, a technique that estimates a graph from observed data, is needed in various applications.

Two main types of approaches have been proposed for graph learning. *Smoothness-based approaches* estimate a graph by solving an optimization problem so that the measurements are smooth over the transform associated to the estimated graph, i.e., their 2-Dirichlet form is small [8–10]. *Statistical approaches* use a probabilistic graphical model perspective, where an optimization problem is solved to identify model parameters [11–13]. Both types of approaches aim at identifying a single graph from all observed data, but they do not take into consideration the fact that relationships between data variables may be time-varying.

This paper focuses on learning a time-varying graph from spatiotemporal data. There are many promising applications using time-varying graphs: Estimation of the time-varying brain functional connectivity from EEGs or fMRI [14], identification of temporal transit of biological networks, e.g., protein, RNA, and DNA [15], and inference of relationships among companies from historical stock price data [16], to name a few. It is often desirable for the time-varying networks estimated in the above (and other) applications to have the following properties:

P1 Most edges and their weights should remain unchanged over a short-term time horizon; however, they should be allowed to change over a longer time horizon. For example, it is reported that time-varying graphs in fMRI and various biological networks follow this property [15, 17].

This work was supported in part by JST PRESTO (JPMJPR1656), JST CREST (JPMJCR1784), and JSPS KAKENHI (JP16H04362).

P2 The graphs chosen at any given time slot should be sparse; i.e., the number of edges should be small enough so as to represent the relationship among data simply and effectively.

P3 Measurements should be smooth over the estimated graph.

Additionally, in many real situations, it is difficult to collect large amounts of data under exactly the same observation conditions. Thus, it would be desirable to have the ability to learn time-varying graphs from a small number of measurements.

In [18], a time-varying graph learning problem is formulated under the assumption that edge weights in the time-varying graph change smoothly over time. The problem is formulated by using the signal smoothness term in each time slot (**P3**) along with Tikhonov regularization (ℓ_2-regularization) of the temporal variation of weighted adjacency matrices. Unfortunately, this method does not work well in the above applications. Tikhonov regularization promotes smooth variation of edge weights over time, i.e., it allows *changes* of both edges and edge weights over short-term time horizons. Experimentally, this approach tends to yield a large number of edges in the estimated graph, which does not satisfy **P1** and **P2** listed above.

In this paper, we formulate a time-varying graph learning problem with the goal of learning graphs having a relatively small number of edges at all times, while also limiting fast changes to the topology and/or the edge weights. We achieve this by introducing a constraint on *sparseness* of the temporal variation, i.e., the number of nonzero elements in the difference between adjacency matrices in neighboring time slots. Our proposed method modifies the smoothness-based optimization approach by introducing an ℓ_1-regularization term for the temporal variation. The main contributions of this paper are:

- A convex optimization formulation is presented to estimate time-varying graphs with sparse temporal variations.
- The proposed method can successfully learn high-quality time-varying graphs from a small amount of available data.

The remainder of this paper is organized as follows. We summarize graph learning methods with signal smoothness in Section 2. Section 3 introduces our spatiotemporal model, defines an optimization problem to estimate its parameters and proposes an algorithm to find a solution. Experimental results with synthetic data are provided in Section 4, while conclusions are given in Section 5.

2. GRAPH LEARNING BASED ON SIGNAL SMOOTHNESS

An undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ consists of a set of nodes $\mathcal{V}$, a set of edges $\mathcal{E}$, and a weighted adjacency matrix $\mathbf{W}$. The

number of nodes is given by $N = |\mathcal{V}|$. The graph Laplacian is given by $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the degree matrix.

Graph learning can be regarded as the problem of learning a weighted adjacency matrix $\mathbf{W}$ from K samples of data available for learning $\mathbf{x}_1, \ldots, \mathbf{x}_K \in \mathbb{R}^N$. The learned weighted adjacency matrix has to be in the following set:

$$\mathcal{W}_m = \left\{ \mathbf{W} \mid \mathbf{W} \in \mathbb{R}_+^{N \times N}, \mathbf{W} = \mathbf{W}^\top, \mathrm{diag}(\mathbf{W}) = \mathbf{0} \right\} \quad (1)$$

where $\mathbb{R}_+$ is the set of nonnegative real numbers.

Under the assumption that a (graph) signal is smooth over the graph, the graph structure can be obtained by solving the following optimization problem:

$$\min_{\mathbf{W} \in \mathcal{W}_m} \sum_{k=1}^{K} \mathbf{x}_k^\top \mathbf{L} \mathbf{x}_k + f(\mathbf{W}), \quad (2)$$

where $f(\mathbf{W})$ is a regularization function to avoid obtaining a trivial solution i.e., a zero matrix. The first term of (2) quantifies the smoothness of all observed signals over the selected graph and can be rewritten as follows [19, 20]:

$$\sum_{k=1}^{K} \mathbf{x}_k^\top \mathbf{L} \mathbf{x}_k = \frac{1}{2} \mathrm{Tr}(\mathbf{W}\mathbf{Z}) = \|\mathbf{W} \circ \mathbf{Z}\|_1, \quad (3)$$

where $\mathbf{Z}$ is the pairwise distance matrix. Several graph learning methods have been proposed based on this smoothness criterion [8–10]. For example, Kalofolias [8] formulates the following optimization problem:

$$\min_{\mathbf{W} \in \mathcal{W}_m} \|\mathbf{W} \circ \mathbf{Z}\|_1 - \alpha \mathbf{1}^\top \log(\mathbf{W}\mathbf{1}) + \beta ||\mathbf{W}||_F^2, \quad (4)$$

where $\mathbf{1} = [1, \ldots, 1]^\top$, $\circ$ is the Hadamard product, and α and β are parameters. The second term of (4) forces the degree on each node to be positive without preventing edge weights from becoming zero.

3. PROPOSED METHOD

In this section, we formulate our proposed time-varying graph learning problem as a convex optimization problem, and present an algorithm for efficiently solving it. While we also use the smoothness criterion of (3), in addition our formulation introduces terms that penalize temporal variations.

3.1. Problem Formulation

We define time-varying graph learning as the problem that estimates the weighted adjacency matrices $\mathbf{W}_1, \ldots, \mathbf{W}_T$ from spatiotemporal data $\{\mathbf{X}_1, \ldots, \mathbf{X}_T\}$, where $\mathbf{X}_t = [\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_K^{(t)}] \in \mathbb{R}^{N \times K}$ is the tth measurement of time-varying signals in which the number of time slots (frames) is T. Our proposed method handles this problem by extending (4).

Our goal is to find a solution such that the temporal variation between $\mathbf{W}_t$ and $\mathbf{W}_{t-1}$ will be sparse. Based on the assumption, we formulate the optimization problem as follows:

$$\min_{\mathbf{W} \in \mathcal{W}_m} \sum_{t=1}^{T} f(\mathbf{W}_t) + \eta \sum_{t=2}^{T} \|\mathbf{W}_t - \mathbf{W}_{t-1}\|_1, \quad (5)$$

where

$$f(\mathbf{W}_t) = \|\mathbf{W}_t \circ \mathbf{Z}_t\|_1 - \alpha \mathbf{1}^\top \log(\mathbf{W}_t \mathbf{1}) + \beta ||\mathbf{W}_t||_F^2, \quad (6)$$

in which $\mathbf{W}_t$ and $\mathbf{Z}_t$ denote the weighted matrix and the pairwise matrix at a certain time frame t, respectively. The parameter η controls the temporal sparseness. In this formulation, we characterize the relation between $\mathbf{W}_t$ and $\mathbf{W}_{t-1}$ by imposing the ℓ_1 norm regularization term. Though it seems similar to (4) and is convex, it cannot be solved with the method in [8], i.e., we need a nontrivial reformulation.

In order to make the optimization problem tractable, we rewrite (5) and (6) in vector form. The representations of the upper-right elements in $\mathbf{W}_t$ and $\mathbf{Z}_t$ in the vector form are denoted by $\mathbf{w}_t \in \mathbb{R}_+^{N(N-1)/2}$ and $\mathbf{z}_t \in \mathbb{R}_+^{N(N-1)/2}$, respectively.

Let $\mathbf{w} = [\mathbf{w}_1^\top \ \mathbf{w}_2^\top \ldots \mathbf{w}_T^\top]^\top$, $\mathbf{z} = [\mathbf{z}_1^\top \ \mathbf{z}_2^\top \ldots \mathbf{z}_T^\top]^\top$, and $\mathbf{d} = [\mathbf{d}_1^\top \ \mathbf{d}_2^\top \ldots \mathbf{d}_T^\top]^\top$ with $\mathbf{d}_t = \mathbf{W}_t \mathbf{1}$. We also introduce the linear operators $\mathbf{S}$ and $\mathbf{\Phi}$ that satisfy $\mathbf{S}\mathbf{w} = \mathbf{d}$ and $\mathbf{\Phi}\mathbf{w} = \mathbf{w} - \hat{\mathbf{w}}$ where $\hat{\mathbf{w}} = [\mathbf{w}_1^\top \ \mathbf{w}_1^\top \ \mathbf{w}_2^\top \ldots \mathbf{w}_{T-1}^\top]^\top$, respectively. Then, we can rewrite (5) and (6) as

$$\min_{\mathbf{w} \in \mathcal{W}_v} \ 2\mathbf{z}^\top \mathbf{w} - \alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}) + \beta \|\mathbf{w}\|_2^2 + \eta \|\mathbf{\Phi}\mathbf{w}\|_1, \quad (7)$$

where

$$\mathcal{W}_v = \left\{ \mathbf{w} \in \mathbb{R}^{TN(N-1)/2} \mid w_i \geq 0 \ (i = 1, 2, \ldots) \right\} \quad (8)$$

corresponds to the nonnegative constraint. By expressing (1) in vector form, the constraints of (1) can be simplified.

3.2. Optimization

We solve (5) with a primal-dual splitting (PDS) method [21], which can be applied to convex optimization problems written as follows:

$$\min_{\mathbf{w}} f_1(\mathbf{w}) + f_2(\mathbf{w}) + f_3(\mathbf{A}\mathbf{w}), \quad (9)$$

where f_1 is a differentiable convex function with gradient ∇f_1 having Lipschitz constant ξ, f_2 and f_3 are proper lower semicontinuous convex functions that are proximable, and $\mathbf{A}$ is a linear operator.

Many algorithms to solve (9) have been proposed. In this paper, we use a method based on a forward-backward-forward (FBF) approach [22]. This is because the FBF-based method makes it possible to compute the proximal operators[1] of functions f_2 and f_3 in parallel.

We can rewrite (7) with the indicator function for $\mathcal{W}_v$ as follows:

$$\min_{\mathbf{w}} \ 2\mathbf{z}^\top \mathbf{w} - \alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}) + \beta \|\mathbf{w}\|_2^2 + \eta \|\mathbf{\Phi}\mathbf{w}\|_1 + \iota_{\mathcal{W}_v}(\mathbf{w}), \quad (10)$$

where the indicator function $\iota_{\mathcal{W}_v}$ is defined by

$$\iota_{\mathcal{W}_v}(\mathbf{w}) = \begin{cases} 0 & w_i \geq 0 \\ \infty & \text{otherwise.} \end{cases} \quad (11)$$

By introducing the dual variable given by $\mathbf{v} = (\mathbf{v}_1^\top \ \mathbf{v}_2^\top)^\top$, the objective function (10) reduces to the applicable form of the PDS as follows:

$$\begin{aligned} f_1(\mathbf{w}) &= \beta \|\mathbf{w}\|_2^2 \ \text{ with } \xi = 2\beta, \\ f_2(\mathbf{w}) &= 2\mathbf{z}^\top \mathbf{w} + \iota_{\mathcal{W}_v}(\mathbf{w}), \\ f_3(\mathbf{v}) &= -\alpha \mathbf{1}^\top \log(\mathbf{v}_1) + \eta \|\mathbf{v}_2\|_1, \\ \mathbf{A} &= \begin{bmatrix} \mathbf{S} \\ \mathbf{\Phi} \end{bmatrix}. \end{aligned} \quad (12)$$

[1] Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ be a proper lower semicontinuous convex function. The proximal operator $\mathrm{prox}_{\gamma f} : \mathbb{R}^n \to \mathbb{R}^n$ of f with a parameter $\gamma > 0$ is defined by $\mathrm{prox}_{\gamma f}(\mathbf{x}) = \arg\min_{\mathbf{y}} f(\mathbf{y}) + \frac{1}{2\gamma} \|\mathbf{y} - \mathbf{x}\|_2^2$. If a proximal operator $\mathrm{prox}_{\gamma f}$ can be computed efficiently, the function f is called proximable.

The proximal operator of f_2 is given by

$$\left(\text{prox}_{\gamma(\|\cdot\|_1+\iota_{\mathcal{W}_v})}(\mathbf{x})\right)_i = \begin{cases} 0 & x_i \le \gamma \\ x_i - \gamma & \text{otherwise.} \end{cases} \tag{13}$$

Since f_3 is separable across variables, the proximal operator can be computed separately for each term. The proximal operator of the log barrier function [23] in the first term of f_3 is given by

$$\left(\text{prox}_{\gamma(-\mathbf{1}^\top \log(\cdot))}(\mathbf{x})\right)_i = \frac{x_i + \sqrt{x_i^2 + 4\gamma}}{2}. \tag{14}$$

In the second term, the proximal operator of the ℓ_1 norm is well known to be the element-wise soft-thresholding operation:

$$\left(\text{prox}_{\gamma\|\cdot\|_1}(\mathbf{x})\right)_i = \begin{cases} 0 & |x_i| \le \gamma \\ \text{sgn}(x_i)(|x_i| - \gamma) & \text{otherwise.} \end{cases} \tag{15}$$

Finally, the detail of the proposed algorithm is summarized in Algorithm 1. The computational complexity of the proposed method is $\mathcal{O}(N^2)$ per iteration. Since the optimization problem in the proposed method is convex and lower-semicontinuous, convergence of the algorithms is guaranteed.

Algorithm 1 Primal-dual splitting algorithm

Input: $\mathbf{w}^{(0)}, \mathbf{v}_1^{(0)}, \mathbf{v}_2^{(0)}$
Output: $\mathbf{w}^{(i)}$
  for $i = 0$ to $i_{\max}$ **do**
    $\mathbf{y}^{(i)} = \mathbf{w}^{(i)} - \gamma(2\beta\mathbf{w}^{(i)} + \mathbf{S}^\top\mathbf{v}_1^{(i)} + \mathbf{\Phi}^\top\mathbf{v}_2^{(i)})$
    $\bar{\mathbf{y}}_1^{(i)} = \mathbf{v}_1^{(i)} + \gamma(\mathbf{S}\mathbf{w}^{(i)})$
    $\bar{\mathbf{y}}_2^{(i)} = \mathbf{v}_2^{(i)} + \gamma(\mathbf{\Phi}\mathbf{w}^{(i)})$
    $\mathbf{p}^{(i)} = \text{prox}_{\gamma(\|\cdot\|_1+\iota_{\mathcal{W}_v})}(\mathbf{y}^{(i)})$
    $\bar{\mathbf{p}}_1^{(i)} = \bar{\mathbf{y}}_1^{(i)} - \gamma\text{prox}_{\frac{1}{\gamma}(-\mathbf{1}^\top\log(\cdot))}\left(\frac{\bar{\mathbf{y}}_1^{(i)}}{\gamma}\right)$
    $\bar{\mathbf{p}}_2^{(i)} = \bar{\mathbf{y}}_2^{(i)} - \gamma\text{prox}_{\frac{1}{\gamma}\|\cdot\|_1}\left(\frac{\bar{\mathbf{y}}_2^{(i)}}{\gamma}\right)$
    $\mathbf{q}^{(i)} = \mathbf{p}^{(i)} - \gamma(2\beta\mathbf{p}^{(i)} + \mathbf{S}^\top\bar{\mathbf{p}}_1^{(i)} + \mathbf{\Phi}^\top\bar{\mathbf{p}}_2^{(i)})$
    $\bar{\mathbf{q}}_1^{(i)} = \bar{\mathbf{p}}_1^{(i)} + \gamma(\mathbf{S}\mathbf{p}^{(i)})$
    $\bar{\mathbf{q}}_2^{(i)} = \bar{\mathbf{p}}_2^{(i)} + \gamma(\mathbf{\Phi}\mathbf{p}^{(i)})$
    $\mathbf{w}^{(i+1)} = \mathbf{w}^{(i)} - \mathbf{y}^{(i)} + \mathbf{q}^{(i)}$
    $\mathbf{v}_1^{(i+1)} = \mathbf{v}_1^{(i)} - \bar{\mathbf{y}}_1^{(i)} + \bar{\mathbf{q}}_1^{(i)}$
    $\mathbf{v}_2^{(i+1)} = \mathbf{v}_2^{(i)} - \bar{\mathbf{y}}_2^{(i)} + \bar{\mathbf{q}}_2^{(i)}$
    if $\|\mathbf{w}^{(i+1)} - \mathbf{w}^{(i)}\|/\|\mathbf{w}^{(i)}\| < \epsilon$ **then**
      break
    end if
  end for

4. EXPERIMENTAL RESULTS

In this section, we present experimental results with synthetic data. We compare the performance of our proposed method with the graph learning methods based on signal smoothness assumption. Specifically, the proposed method (hereinafter called TVGL-Sparse: Time-Varying Graph Learning with Sparseness of Temporal Variation) is compared with the static graph learning with the smoothness-based approach (SGL-Smooth) [8] and the time-varying graph learning method with temporal smoothness (TVGL-Smooth) [18]. The tolerance value ϵ in Algorithm 1 is set to 1.0×10^{-4}.

4.1. Synthetic Datasets

To evaluate the performance of the proposed method, we create two synthetic datasets generated from time-varying graphs: A time-varying graph based on random waypoint model (abbreviated as TV-RW graph) [24] and a time-varying Erdős–Rényi graph (abbreviated as TV-ER graph).

The dataset construction consists of two steps: the first step is constructing a time-varying graph, and the second step generates time-varying graph signals from distributions based on graph Laplacians of the created time-varying graph. We describe these details next by referring to the desired properties **P1**–**P3** described in Section 1.

4.1.1. Time-Varying Graph Construction

The TV-RW graph is constructed in two steps. First, we simulate the RW model to obtain time-varying coordinates over time. The model simulates some sensors moving around a square space with random speed and directions. With this simulation, we obtain the position data of each sensor at each time; this data is time-varying. We set the number of sensors $N = 36$ and the motion speeds to be in the range 0.05–0.5 (m/s). Sensor positions are sampled every 0.1 (s) for total 300 time frames. In the second step, we construct a 3-nearest-neighbor graph at each time from the position data. Hence, the graph satisfies **P2**. These edge weights are given by

$$w(i,j) = \exp\left(-\text{dist}(i,j)/(2\theta)\right), \tag{16}$$

where $\text{dist}(i,j)$ is the Euclidean distance between nodes i and j, and θ is a parameter. Edge weights in this graph generally vary smoothly as in (16); however, some edges will (dis)appear in the next time slot due to the nature of the 3-neighborhood graph. This graph partially satisfies **P1** because the topology mostly remains unchanged while the edge weights change smoothly over time.

TV-ER graph is constructed by varying edges in an ER graph over time. We first construct an initial static ER graph $\mathbf{W}_1$ with $N = 36$ and edge connection probability $p = 0.05$: the obtained graph satisfies **P2**. The tth graph $\mathbf{W}_t$ is obtained by resampling 5% of edges from the previous graph $\mathbf{W}_{t-1}$. The weight of the edges in this graph is randomly determined by a uniform distribution from the interval $[0, 1]$. In this graph, only a few edges switch at a time while most of the edges remain unchanged, i.e., this graph also follows **P1**.

4.1.2. Generating Data Samples

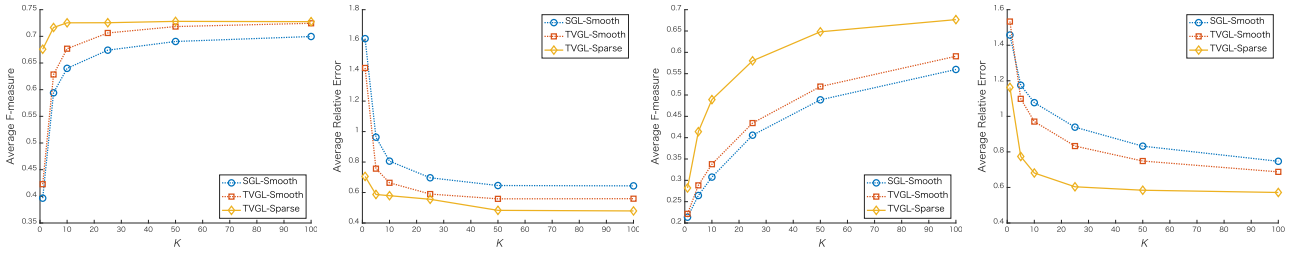
We create data samples from the constructed time-varying graph. Let $\mathbf{L}_t$ be the graph Laplacian of a graph at a certain time slot t. A data sample $\mathbf{x}_t$ is generated from a Gaussian Markov random field defined by

$$p(\mathbf{x}_t|\mathbf{L}^{(t)}) = \mathcal{N}(\mathbf{x}_t|0, (\mathbf{L}_t + \sigma^2\mathbf{I})^{-1}), \tag{17}$$

where σ^2 is the covariance of the Gaussian noise; we set $\sigma = 0.5$ in this experiments. Pairs of variables generated from this distribution have closer values each other when the corresponding nodes connect with a larger edge weight. Hence, the data generated from this distribution satisfies **P3**.

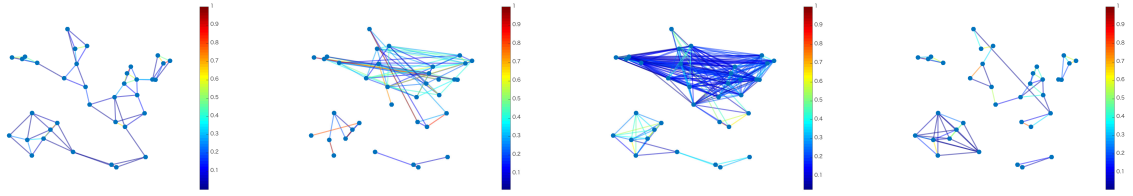
4.2. Performance Comparison

We evaluate the performance in terms of the relative error and F-measure, each averaged over all time slots. Relative error reflects the accuracy of edge weights on the estimated graph. The F-measure,

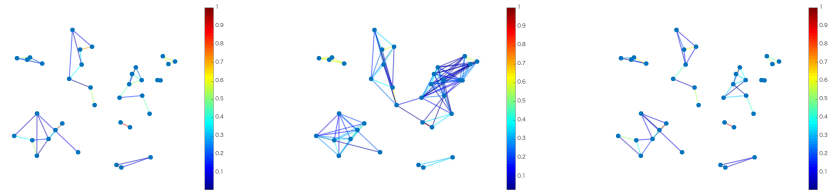


(a) (b) (c) (d)

Fig. 1. The performance of time-varying graph learning for different number of data samples. (a) and (b) demonstrate the F-measure and the relative error for the dataset based on TV-RW graph. (c) and (d) demonstrate those for TV-ER graph.



(a) Ground truth (b) SGL-Smooth ($K = 1$) (c) TVGL-Smooth ($K = 1$) (d) TVGL-Sparse ($K = 1$)



(e) SGL-Smooth ($K = 100$) (f) TVGL-Smooth ($K = 100$) (g) TVGL-Sparse ($K = 100$)

Fig. 2. The visualization of a graph at a certain time in the time-varying graph learned from the dataset based on TV-RW graph. Edge colors represent weights of the edges.

which is the harmonic average of the precision and recall, represents the accuracy of the estimated graph structure.

In our experiments, we set the parameter for each method to the best one which is found by grid search. We evaluate the performance for each method with the different number of data samples $K = \{1, 5, 10, 25, 50, 100\}$ and measure the average of the relative error and the F-measure over all time frames.

Fig. 1 shows the performance comparisons according to the number of the available data K. Figs. 1(a) and (b) show the average F-measure and relative error for TV-RW graphs; Figs. 1(c) and (d) show those for TV-ER graphs. As shown in Fig. 1, SGL-Smooth presents the worst performance in both datasets. This is because SGL-Smooth learns a graph from each time slot independently; it does not consider the temporal relation of graphs. Fig. 1(a) and 1(b) also present that TVGL-Sparse outperforms TVGL-Smooth despite of the fact that the edges in a TV-RW graph vary smoothly in this case. This would be because undesirable edges are yielded by TVGL-Smooth. TVGL-Sparse significantly outperforms the other methods for TV-ER graphs as shown in Figs. 1(c) and (d). This indicates that the sparseness constraint on the difference between graphs in neighboring time slots works well.

It is also worth noting that, thanks to the regularization term, TVGL-Sparse can successfully learn time-varying graphs with a relatively small amount of data. As can be seen in Fig. 1, the performance gain for TV-ER graphs is much better than that of TV-RW graphs. This is because TV-ER graphs have all of the properties **P1**–**P3** while TV-RW graphs lack a part of **P1**, as previously mentioned.

Fig. 2 visualizes the learned TV-RW graphs. As can be seen, SGL-Smooth and TVGL-Smooth cannot capture the original graph structure when $K = 1$. In contrast, TVGL-Sparse estimated the original structure well even for that case. When $K = 100$, SGL-Smooth and TVGL-Sparse yields similar graphs while TVGL-Smooth still yields many undesirable edges.

5. CONCLUSION

In this work, we presented the learning method of time-varying graphs under the constraint on the sparseness of the temporal variation of edges. We demonstrated that our proposed model can be efficiently solved using the primal-dual splitting algorithm and successfully learn the time-varying graph, especially under the condition that the number of observations is small. Our future work includes implementing an automatic parameter tuning method and extending the algorithm for online graph learning.

6. REFERENCES

[1] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst, "Fast robust PCA on graphs," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 4, pp. 740–756, 2016.

[2] F. Shang, L. C. Jiao, and F. Wang, "Graph dual regularization non-negative matrix factorization for co-clustering," *Pattern Recognition*, vol. 45, no. 6, pp. 2237–2250, 2012.

[3] Z. Zhang and K. Zhao, "Low-rank matrix approximation with manifold regularization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1717–1729, 2013.

[4] A. Gadde, S. K. Narang, and A. Ortega, "Bilateral filter: Graph spectral interpretation and extensions," in *Proc. IEEE Int. Conf. Image Process.*, 2013, pp. 1222–1226.

[5] Y. Wang, A. Ortega, D. Tian, and A. Vetro, "A graph-based joint bilateral approach for depth enhancement," in *Proc. IEEE Int. Conf. Acoust. Speech. Signal Process.*, 2014, pp. 885–889.

[6] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 2, pp. 137–148, 2016.

[7] S. Ono, I. Yamada, and I. Kumazawa, "Total generalized variation for graph signals," in *Proc. IEEE Int. Conf. Acoust. Speech. Signal Process.*, 2015, pp. 5456–5460.

[8] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.

[9] 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, 2016.

[10] S. P. Chepuri, S. Liu, G. Leus, and A. O. Hero, "Learning sparse graphs under smoothness prior," in *Proc. IEEE Int. Conf. Acoust. Speech. Signal Process.*, 2017, pp. 6508–6512.

[11] R. Mazumder and T. Hastie, "The graphical Lasso: New insights and alternatives," *Electron. J. Stat.*, vol. 6, no. 0, pp. 2125–2149, 2012.

[12] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the Lasso," *Ann. Statist.*, vol. 34, no. 3, pp. 1436–1462, 2006.

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

[14] M. G. Preti, T. A. Bolton, and D. Van De Ville, "The dynamic functional connectome: State-of-the-art and perspectives," *NeuroImage*, vol. 160, pp. 41–54, 2017.

[15] Y. Kim, S. Han, S. Choi, and D. Hwang, "Inference of dynamic networks using time-course data," *Brief. Bioinformatics*, vol. 15, no. 2, pp. 212–228, 2014.

[16] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical Lasso," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 205–213.

[17] R. P. Monti, P. Hellyer, D. Sharp, R. Leech, C. Anagnostopoulos, and G. Montana, "Estimating time-varying brain connectivity networks from functional MRI time series," *NeuroImage*, vol. 103, pp. 427–443, 2014.

[18] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, "Learning time varying graphs," in *Proc. IEEE Int. Conf. Acoust. Speech. Signal Process.*, 2017, pp. 2826–2830.

[19] 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, 2013.

[20] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, 2014.

[21] N. Komodakis and J. Pesquet, "Playing with duality: An overview of recent primal?dual approaches for solving large-scale optimization problems," *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 31–54, 2015.

[22] L. Condat, "A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *J. Optim. Theory Appl.*, vol. 158, no. 2, pp. 460–479, 2013.

[23] N. Parikh and S. Boyd, "Proximal algorithms," *Found Trends Optim.*, vol. 1, no. 3, pp. 127–239, 2014.

[24] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, 1996, pp. 153–181.