DYNAMIC K-GRAPHS: AN ALGORITHM FOR DYNAMIC GRAPH LEARNING AND TEMPORAL GRAPH SIGNAL CLUSTERING

Hesam Araghi[⋆] *Massoud Babaie-Zadeh*[⋆] *Sophie Achard*[†]

[⋆]Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran
[†]University of Grenoble Alpes, CNRS, Grenoble INP, GIPSA-Lab, Grenoble, France
Email: hesam.araghi@ee.sharif.edu, mbzadeh@yahoo.com, sophie.achard@gipsa-lab.grenoble-inp.fr

ABSTRACT

Graph signal processing (GSP) have found many applications in different domains. The underlying graph may not be available in all applications, and it should be learned from the data. There exist complicated data, where the graph changes over time. Hence, it is necessary to estimate the dynamic graph. In this paper, a new dynamic graph learning algorithm, called dynamic K-graphs, is proposed. This algorithm is capable of both estimating the time-varying graph and clustering the temporal graph signals. Numerical experiments demonstrate the high performance of this algorithm compared with other algorithms.

Index Terms— Dynamic K-graphs, dynamic graph learning, graph Laplacian matrix, temporal graph signal clustering, dynamic programming

1. INTRODUCTION

Developments in the field of data science enables us to analyze complicated data with irregular structures. Graphs are one of the ubiquitous tools to model the relationship between the elements of complex signals, and the need for representing and inferring the data can be answered by the emerging field of graph signal processing (GSP) [1, 2]. This new approach has found various applications in different domains such as sensor and brain networks [3, 4]. In many of these applications, applying the GSP-based techniques is only possible when the underlying graph for the dataset is well-defined. However, it is obvious that the graph structure may not be available in all applications, and it should be computed from the data. Thus, many graph learning algorithms have recently been proposed to identify the topology of the underlying graph in a dataset [5–7].

Probabilistic models can be used to learn a graph from the dataset, and Gaussian Markov random field (GMRF) is one of these models that represents the graph with precision matrix. Then, the precision matrix can be estimated by the methods like graphical LASSO [8]. However, the resulted graph may not be easily interpretable due to the negative weights [5]. So to prevent this problem, [9] restricts the precision matrix to have a graph Laplacian shape. In [10], a stochastic model is introduced that assumes the graph signals are smooth over a graph, and a maximum a posteriori estimator used to obtain the graph. Similar to [10], smoothness of the signals over the graph is utilized in [11], but cases with different regularization terms for the Laplacian matrix are also studied, and computationally efficient algorithms are proposed in each case. Authors in [12], based on diffusion process model, characterize the set of admissible diffusion matrices and select a graph topology with desirable properties from this set.

In addition to single graph learning, there are other methods that consider learning multiple latent graphs from more complicated data or finding a time-varying graph from the temporal graph signals. In [13] based on the concept of Gaussian mixture model (GMM), an algorithm called graph Laplacian mixture model (GLMM) is proposed. This work assumes the dataset consists of smooth graph signals over different graphs, and tries to estimate the Laplacian matrices of the graphs. K-graphs [14] is also another multiple graph learning algorithm inspired by K-means. Unlike GLMM and K-graphs, which suppose the graph signals are independent in different time stamps, there are other dynamic graph learning algorithms that consider the time dependency of signals. In [15], a dynamic graphical LASSO algorithm is introduced, and a precision matrix for each time stamp is computed. The time-varying version of graph learning algorithm [11] is proposed in [16]. In [17], another graphical LASSO based algorithm, called TICC, is proposed that partitions the whole time span into multiple segments, and assigns for each segment a constant precision matrix. This algorithm can also be used for multivariate time series clustering.

In this paper, we propose a dynamic graph learning algorithm, called dynamic K-graphs, where unlike [15] and [16], the underlying graph only should be chosen from a fixed number of different graphs. Similar to TICC [17], it can extract the change points and the time intervals. Then in each time interval, it finds a constant structure for the signals of that interval. Moreover, dynamic K-graphs is also able to cluster temporal graph signals. However, since our method works with Laplacian matrices, it can offer more interpretable graph

structures, and as will be seen in simulations, our algorithm has higher clustering accuracy compared with TICC.

2. GSP BACKGROUND

In this section, we introduce some necessary preliminaries of GSP that are used in this paper. A weighted and undirected graph $\mathcal{G}$ can be represented by a triple $(\mathcal{V}, \mathcal{E}, \mathbf{A})$ where $\mathcal{V}$ is a finite set of N nodes (or vertices), $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ is a set of edges, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the symmetric weighted adjacency matrix with zero values for its diagonal entries. The (i, j)-th entry of the matrix $\mathbf{A}$ is denoted by $\mathbf{A}[i, j]$, and the value of $\mathbf{A}[i, j]$ (similarly $\mathbf{A}[j, i]$) is the edge weight between the nodes i and j. The weighted degree matrix $\mathbf{D}$ is a diagonal matrix containing the degree of each node: $\mathbf{D}[i, i] = \sum_{j=1}^{N} \mathbf{A}[i, j]$, $i = 1, \ldots, N$. Then, the graph Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, and the valid set of graph Laplacian matrices becomes

$$\mathcal{L} = \{ \mathbf{L} \in \mathbb{R}^{N \times N} : \mathbf{L} = \mathbf{L}^T, \mathbf{L}[i, j] \leq 0 \ (\forall i \neq j), \mathbf{L} \cdot \mathbf{1} = \mathbf{0} \}, \quad (1)$$

where $\mathbf{0}$ and $\mathbf{1}$ are all-zero and all-one vectors, respectively, and $(\cdot)^T$ denotes matrix transposition.

A graph signal is a vector $\mathbf{x}$, whose i-th entry $\mathbf{x}[i]$ assigns a real value to the node i of the graph $\mathcal{G}$. To measure the smoothness of a graph signal $\mathbf{x}$ over a graph, we utilize the following popular graph Laplacian quadratic form [1]

$$\mathbf{x}^T \mathbf{L} \mathbf{x} = \sum_{i,j} \mathbf{A}[i, j](\mathbf{x}[i] - \mathbf{x}[j])^2, \quad (2)$$

where a small value of $\mathbf{x}^T \mathbf{L} \mathbf{x}$ indicates that signal $\mathbf{x}$ has small variations across the strongly connected nodes. Moreover, since the edge weights $\mathbf{A}[i, j]$ are non-negative, the equality in (2) yields the Laplacian matrix to be positive semi-definite.

One of the important concepts for undirected graphs is graph Fourier transform (GFT) obtained by eigendecomposition of graph Laplacian $\mathbf{L} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^T$ [1], where $\mathbf{V}$ is an orthonormal matrix containing the eigenvectors $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_N]$, and $\mathbf{\Lambda}$ is a diagonal matrix with non-negative eigenvalues, $\lambda_1, \ldots, \lambda_N$, on its diagonal entries. These eigenvectors and eigenvalues are used as the basis and the corresponding frequencies, respectively. The GFT of signal $\mathbf{x}$ is defined as $\hat{\mathbf{x}} = \mathbf{V}^T \mathbf{x}$, and the inverse GFT becomes $\mathbf{x} = \mathbf{V} \hat{\mathbf{x}}$.

To filter a graph signal $\mathbf{x}$, similar to the procedure in the classical digital signal processing, we can take GFT, $\hat{\mathbf{x}} = \mathbf{V}^T \mathbf{x}$, modify the GFT coefficient, $\hat{\mathbf{y}} = g(\hat{\mathbf{x}})$, and perform inverse GFT, $\mathbf{y} = \mathbf{V} \hat{\mathbf{y}}$. For lowpass filtering, one common choice is $\hat{\mathbf{y}} = \mathbf{\Lambda}^{-1/2} \hat{\mathbf{x}}$ [10], where $\Lambda^{-1/2} = \mathrm{diag}(h(\lambda_1), \ldots, h(\lambda_N))$ with $h(\lambda) = \lambda^{-1/2}$ for $\lambda \neq 0$, and $h(\lambda) = 0$ for $\lambda = 0$. Then, the overall filtering is equivalent to multiplying the input signal $\mathbf{x}$ by the matrix $\mathbf{L}^{-1/2} \triangleq \mathbf{V} \mathbf{\Lambda}^{-1/2} \mathbf{V}^T$.

3. DYNAMIC K-GRAPHS ALGORITHM

Let the multivariate time series $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$ consist of T temporal graph signals $\mathbf{x}_t \in \mathbb{R}^N$, $t = 1, \ldots, T$, and the whole time span $\{t\}_{t=1}^{T}$ is divided into $q + 1$ different time intervals $\mathcal{T}_i = \{t\}_{t=t_{i-1}^{(c)}+1}^{t_i^{(c)}}$, $i = 1, \ldots, q+1$, where $\{t_i^{(c)}\}_{i=1}^{q}$ is the set of q change points, $t_0^{(c)} = 0$, and $t_{q+1}^{(c)} = T$. The signals in each time interval $\{\mathbf{x}_t\}_{t \in \mathcal{T}_i}$ comes from only one undirected graph, but between the two consecutive time segments, the topology of the underlying graph may change. Moreover, we assume that the dynamic graph at each time stamp is chosen from one of the graphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_K$. Therefore in each of the intervals $\mathcal{T}_1, \ldots, \mathcal{T}_{q+1}$, the underlying graph must be one of the K different undirected graphs $\mathcal{G}_1, \ldots, \mathcal{G}_K$, which are not known at first. Hence, our goal in dynamic graph learning is to jointly estimate the time intervals $\mathcal{T}_i$'s (or equivalently, the set of change points $\{t_i^{(c)}\}_{i=1}^{q}$) and the set of graphs $\{\mathcal{G}_k\}_{k=1}^{K}$. In addition, we want to find that the signals in each interval correspond to which of the graphs. The graph $\mathcal{G}_k$ is represented by its Laplacian matrix $\mathbf{L}_k$, and the criterion that a graph signal $\mathbf{x}$ comes from the graph $\mathcal{G}_k$ is the smoothness of that signal over the graph using the Laplacian quadratic form in (2).

To address the above problem, we propose a dynamic graph learning algorithm, called dynamic K-graphs, which is the modification of the K-graphs algorithm [14] in such a way that the temporal dependency of the neighboring graph signals also to be incorporated into the algorithm. The dynamic K-graph algorithm can be described as the following minimization problem

$$\min_{\substack{\{\mathbf{L}_k\}_{k=1}^{K}, \\ \{n_t\}_{t=1}^{T}}} \sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{X}_k} \mathbf{x}^T \mathbf{L}_k \mathbf{x} + \sum_{k=1}^{K} f(\mathbf{L}_k) + \alpha \sum_{t=2}^{T} \mathbb{1}(n_t \neq n_{t-1}), \quad (3)$$

$$\text{s.t.} \quad \mathbf{L}_k \in \mathcal{L}, \ (1 \leq k \leq K),$$
$$n_t \in \{1, \ldots, K\}, \ (1 \leq t \leq T),$$
$$\mathcal{X}_k = \{\mathbf{x}_t : t \in \{1, \ldots, T\}, \ n_t = k\}, \ (1 \leq k \leq K),$$

where the set $\mathcal{L}$ is the set of valid graph Laplacian defined in (1) and $\mathbb{1}(n_t \neq n_{t-1})$ is an indicator function that equals to 0 if $n_t = n_{t-1}$ and 1 otherwise. The graph signals that come from the graphs $\mathcal{G}_1, \ldots, \mathcal{G}_K$ are collected in clusters $\mathcal{X}_1, \ldots, \mathcal{X}_K$, respectively, and each cluster $\mathcal{X}_k$ consists of graph signals in one or more time intervals $\mathcal{T}_1, \ldots, \mathcal{T}_{q+1}$. For the graph signal $\mathbf{x}_t$, the variable n_t determines the clusters $1, \ldots, K$, to which the signal belongs to. The first two terms of the objective function (3) is the same as in K-graphs algorithm [14] where $\mathbf{x}^T \mathbf{L}_k \mathbf{x}$ is for measuring the smoothness of $\mathbf{x}$ on $\mathbf{L}_k$, and $f(\mathbf{L}_k)$, as described in [11], contains the regularization terms to eliminate the trivial so-

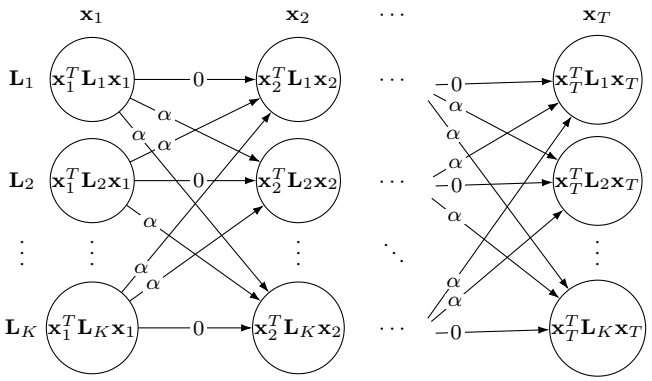


Fig. 1. Dynamic programming for finding the minimum cost path for the optimization problem in (5).

lution of all-zero matrices for $\mathbf{L}_k$'s and also to control the sparsity of the off-diagonal elements of $\mathbf{L}_k$'s. The last term $\alpha \sum_{t=2}^{T} \mathbb{1}(n_t \neq n_{t-1})$ is similar to the "temporal consistency" term used in TICC algorithm [17] and encourages adjacent graph signals to be generated from the same graph. It imposes a cost of α for each change point and results in creation of larger and more integrated time segments. In practice, this parameter can be chosen by cross-validation, or in the case of existing prior knowledge about the size of the time segments or the temporal dependency of the graph signals, the proper value of α can be selected accordingly.

In dynamic K-graphs, we adopt an alternating minimization for $\{\mathbf{L}_k\}_{k=1}^K$ and $\{n_t\}_{t=1}^T$ to solve the problem of (3). For the initialization of the algorithm, each graph signal is randomly and independently assigned to one of K clusters, $\mathcal{X}_1, \ldots, \mathcal{X}_K$.

In the first step of the algorithm, fixing n_t's in (3) yields the following optimization problem for $\mathbf{L}_k$

$$\mathbf{L}_k = \underset{\mathbf{L}\in\mathcal{L}}{\operatorname{argmin}} \sum_{\mathbf{x}\in\mathcal{X}_k} \mathbf{x}^T\mathbf{L}\mathbf{x} + f(\mathbf{L}), \tag{4}$$

where many single graph learning algorithms [5, 11] can be expressed as the form of (4). Therefore, $\mathbf{L}_k$'s can be updated by a single graph learning from the graph signals in $\mathcal{X}_k$.

In the second step of the algorithm, the Laplacian matrices $\mathbf{L}_k$'s are fixed and the graph signals are assigned to one of K graphs $\mathcal{G}_1, \ldots, \mathcal{G}_K$, that is, we intend to solve the following minimization problem for $\{n_t\}_{t=1}^T$

$$\begin{aligned} \min_{\{n_t\}_{t=1}^T} \quad & \sum_{k=1}^{K} \sum_{\mathbf{x}\in\mathcal{X}_k} \mathbf{x}^T\mathbf{L}_k\mathbf{x} + \alpha \sum_{t=2}^{T} \mathbb{1}(n_t \neq n_{t-1}), \\ \text{s.t.} \quad & n_t \in \{1, \ldots, K\}, \quad (1 \leq t \leq T), \\ & \mathcal{X}_k = \{\mathbf{x}_t : t \in \{1, \ldots, T\},\, n_t = k\}, \quad (1 \leq k \leq K). \end{aligned} \tag{5}$$

Similar to the TICC algorithm [17], we can solve (5) through a dynamic programming method as shown in Fig. 1. Equivalently, it is the same as finding the minimum cost path in Viterbi algorithm [18]. At time t, the cost of choosing $n_t = k$ is $\mathbf{x}_t^T\mathbf{L}_k\mathbf{x}_t$ and the transition cost for changing n_t is also α. The path with minimum cost is the answer for the optimization problem (5). The dynamic programming is able to find the optimal solution in $\mathcal{O}(KT)$ operations [17], and the size of the graph, i.e., N, is only involved in the calculation of $\mathbf{x}_t^T\mathbf{L}_k\mathbf{x}_t$. The set of change points is equal to $\{t : t \in \{1, \ldots, T-1\}, n_t \neq n_{t+1}\}$, and the time intervals can be calculated by the change points. As in other clustering methods like K-means, to escape from local minima, it is better to run the algorithm with different initialization points.

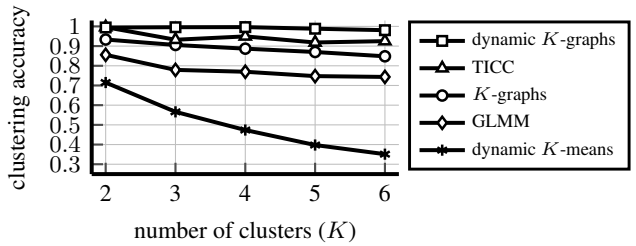
The single graph learning and dynamic programming steps are alternately performed until the algorithm converges. Since both of these steps do not increase the objective function of (3), after each assignment of T graph signals to K clusters, the objective function decreases or does not change. Moreover, the total number of assignments is finite and equals K^T. Therefore, the algorithm converges in a finite number of iterations. The convergence occurs when the two consecutive assignments of graph signals are identical.

4. SIMULATION RESULTS

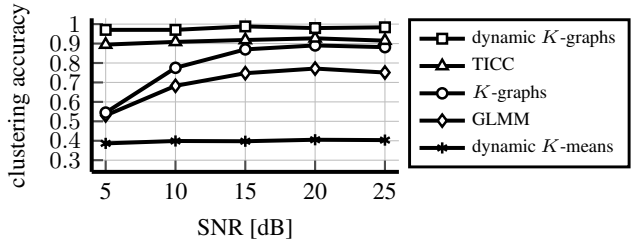
In this section, the performance of **dynamic K-graphs** is numerically compared with four other algorithms: **TICC** [17], ***K*-graphs** [14], **GLMM** [13], and **dynamic *K*-means** (a K-means like algorithm that considers the temporal dependency in the same way of the proposed method in Section 3). We use the `gsp_sensor` command from GSPBOX toolbox [19] to create K different random graphs, $\mathcal{G}_1, \ldots, \mathcal{G}_K$. To generate a synthetic graph signal $\mathbf{x}$ from a graph $\mathcal{G}$, a white noise $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is lowpass filtered by $\mathbf{L}_{\mathcal{G}}^{-1/2}$, i.e., $\mathbf{s}_{\mathcal{G}} = \mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{w}$. Then, the filtered signal $\mathbf{s}_{\mathcal{G}}$ is corrupted by a white Gaussian noise $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2\mathbf{I})$ independent of $\mathbf{w}$, i.e., $\mathbf{x} = \mathbf{s}_{\mathcal{G}} + \mathbf{n}$, where $\mathbf{L}_{\mathcal{G}}$ is the Laplacian matrix of graph $\mathcal{G}$, $\mathbf{I}$ denotes the identity matrix, and σ_n^2 is the variance of the additive noise.

In the following experiments, the time series $\mathbf{X}$ consists of $T = 1000$ temporal graph signals $\mathbf{x}_t \in \mathbb{R}^{30}$, $t = 1, \ldots, T$. The number of change points, q, is randomly selected from the integer set $\{K+1, \ldots, 3K\}$ to include signals from all the K graphs. The q unique change points $t_1^{(c)}, t_2^{(c)}, \ldots, t_q^{(c)}$ are also randomly chosen from the integer set $\{1, \ldots, T-1\}$. Thus, there would be $q+1$ time intervals with random length, where the signals in each interval come from one graph, and the graphs are periodically assigned to each of the time intervals, i.e., $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_K, \mathcal{G}_1, \mathcal{G}_2$, and so on. For a given signal to noise ratio (SNR) in dB, the variance of additive noise σ_n^2 is chosen such that we have $\text{SNR} = 10\log_{10}(\mathcal{E}_{\mathbf{s}_{\mathcal{G}}}/\sigma_n^2)$, where $\mathcal{E}_{\mathbf{s}_{\mathcal{G}}}$ is the empirical mean of graph signals, i.e., $\mathcal{E}_{\mathbf{s}_{\mathcal{G}}} = \sum_{t=1}^{T} \|\mathbf{s}_{\mathcal{G}}(t)\|_2^2/(NT)$.

For single graph learning in (4), similar to K-graphs and GLMM methods, we utilize the graph learning algorithm proposed in [11]. The implementation of the algorithm is done in GSPBOX toolbox [19] with the command `gsp_learn_graph_log_degrees`. Then, the resulting

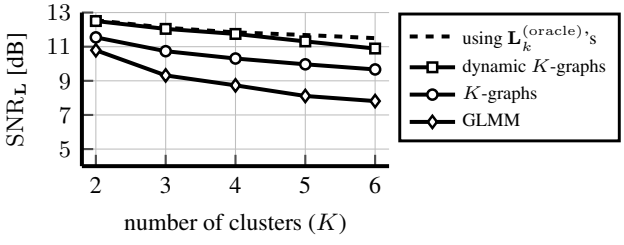


(a) different number of clusters for SNR $= 15$ dB

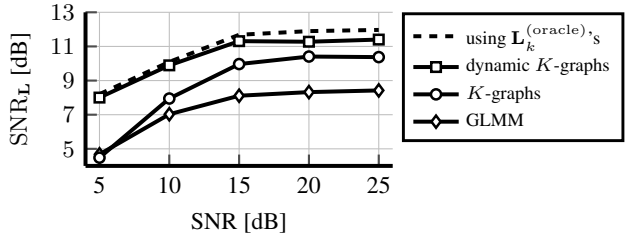


(b) different values of SNR for $K = 5$

Fig. 2. Comparing clustering accuracy of dynamic K-graphs, TICC, GLMM, and dynamic K-means for a) different number of clusters (K) and b) for different values of SNR, with $N = 30$ and $T = 1000$.



(a) different number of clusters for SNR $= 15$ dB



(b) different values of SNR for $K = 5$

Fig. 3. Comparing graph learning performance for dynamic K-graphs, K-graphs, and GLMM for a) different number of clusters (K) and b) for different values of SNR, with $N = 30$ and $T = 1000$.

Laplacian matrices $\mathbf{L}_k$'s are normalized by dividing to their traces to eliminate the dependency of $\mathbf{x}_t^T \mathbf{L}_k \mathbf{x}_t$ on the scale of the Laplacian matrix $\mathbf{L}_k$. The cost parameter in the proposed algorithm is set to $\alpha = 2$. In TICC, parameters of $\lambda = 0.11$, $\beta = 50$, and $w = 1$ are chosen in order to maximize the performance of the algorithm.

For initializing the algorithms, K-graphs has the same initialization as dynamic K-graphs described in Section 3, GLMM starts with K random Laplacian matrices, TICC performs GMM-based clustering for the graph signals in the beginning, and `k-means++` [20] is used for initialization of dynamic K-means. All the algorithms are run with 30 different random starting points in each realization. Among 30 runs, the final output is the one that minimizes the objective function $\sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{X}_k} \mathbf{x}^T \mathbf{L}_k \mathbf{x}$ for K-graphs and GLMM, the objective function of (5) for dynamic K-graphs, and the overall objective function of [17, Eq. (1)] for TICC. For dynamic K-means, we use the objective function of (5) with the first term replaced by $\sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{X}_k} \|\mathbf{x} - \mathbf{c}_k\|_2^2$, where $\mathbf{c}_k$'s are the cluster centers. Each point in the figures is the average of 100 independent realizations.

In Fig. 2, the clustering accuracy of temporal graph signals is evaluated for dynamic K-graphs algorithm with different number of clusters and different values of SNR. To find the optimal mapping of the calculated cluster labels to the true cluster labels, we use the Hungarian method proposed in [21], and the clustering accuracy is the ratio of the number of matched labels to the total number of graph signals. It can be seen from Fig. 2a that the accuracy of dynamic K-graphs is close to 1, and TICC has lower accuracy. The higher performance of dynamic K-graphs can be explained by the better graph representation of Laplacian matrices compared with precision matrices used in TICC algorithm. However, dynamic K-graphs and TICC both have higher clustering accuracy than K-graphs and GLMM, since the latter algorithms do not consider time dependency of graph signals. The worst performance belongs to dynamic K-means due to its inability to model the structure of graph signals. Figure 2b shows the clustering accuracy for different SNRs.

In the second experiment, the graph learning capability of dynamic K-graphs is compared with two other algorithms that provide Laplacian matrices, i.e., K-graphs and GLMM. Our performance criterion is $\mathrm{SNR}_{\mathbf{L}}$ defined in [14, Eq. (11)], which measures the similarity between the estimated Laplacian matrices and the true ones. Moreover, we compute $\mathrm{SNR}_{\mathbf{L}}$ for oracle matrices $\mathbf{L}_k^{(\mathrm{oracle})}$'s which are learned from the truly clustered graph signals, i.e., $\mathcal{X}_k^{(\mathrm{true})}$'s. Calculating $\mathrm{SNR}_{\mathbf{L}}$ with $\mathbf{L}_k^{(\mathrm{oracle})}$'s provides a good upper bound for the performance of our proposed algorithm.

Figure 3 demonstrates $\mathrm{SNR}_{\mathbf{L}}$ for dynamic K-graphs, K-graphs, and GLMM along with the one obtained using $\mathbf{L}_k^{(\mathrm{oracle})}$'s. We can see that for different number of clusters (Fig. 3a) and different values of SNR (Fig. 3b), dynamic K-graphs closely follows the ideal case (obtained from the matrices $\mathbf{L}_k^{(\mathrm{oracle})}$'s).

5. CONCLUSION

In this paper, we proposed a dynamic graph learning algorithm, called dynamic K-graphs. In this algorithm, the Laplacian matrix of the underlying graph does not take arbitrary values at each time stamps, instead it divides the time into

different segments, and assigns one of the K graphs to each segment. This property of the algorithm enables to cluster the temporal graph signals. Numerical simulations show that dynamic K-graphs has a high clustering accuracy and achieves a good performance in graph learning.

6. REFERENCES

[1] 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, pp. 83–98, May 2013.

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

[3] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[4] W. Huang, T. A. W. Bolton, J. D. Medaglia, D. S. Bassett, A. Ribeiro, and D. V. D. Ville, "A graph signal processing perspective on functional brain imaging," *Proc. IEEE*, vol. 106, no. 5, pp. 868–885, May 2018.

[5] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Process. Mag.*, vol. 36, pp. 44–63, May 2019.

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

[7] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proc. IEEE*, vol. 106, no. 5, pp. 787–807, May 2018.

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

[9] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," in *Proc. 32nd Annu. Meeting Cogn. Sci. Soc. (CogSci'10)*, Portland, OR, Aug. 2010, pp. 778–784.

[10] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, pp. 6160–6173, Dec. 2016.

[11] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS'16)*, vol. 51, Cadiz, Spain, Jun. 2016, pp. 920–929.

[12] 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. 4, no. 3, pp. 481–496, Sep. 2018.

[13] H. P. Maretic and P. Frossard, "Graph laplacian mixture model," arXiv:1810.10053v1 [cs.LG], Oct. 2018.

[14] H. Araghi, M. Sabbaqi, and M. Babaie-Zadeh, "K-Graphs: An algorithm for graph signal clustering and multiple graph learning," *IEEE Signal Process. Lett.*, vol. 26, no. 10, pp. 1486–1490, Oct. 2019.

[15] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD'17)*, Halifax, Canada, Aug. 2017.

[16] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, "Learning time varying graphs," in *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'17)*, New Orleans, LA, Mar. 2017, pp. 2826–2830.

[17] D. Hallac, S. Vare, S. Boyd, and J. Leskovec, "Toeplitz inverse covariance-based clustering of multivariate time series data," in *ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD'17)*, Halifax, Canada, Aug. 2017.

[18] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, Apr. 1967.

[19] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," arXiv:1408.5781v2 [cs.IT], Aug. 2014. [Online]. Available: https://epfl-lts2.github.io/gspbox-html/index.html

[20] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA'07)*, New Orleans, LA, Jan. 2007, pp. 1027–1035.

[21] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, Mar. 1955.