LEARNING TIME VARYING GRAPHS

Vassilis Kalofolias[†], *Andreas Loukas*[†⋆], *Dorina Thanou*[⋆], *Pascal Frossard*[⋆]

Signal Processing Laboratory (LTS2[†]/LTS4[⋆])
Ecole Polytechnique Fédérale de Lausanne (EPFL)

ABSTRACT

We consider the problem of inferring the hidden structure of high-dimensional time-varying data. In particular, we aim at capturing the dynamic relationships by representing data as valued nodes in a sequence of graphs. Our approach is motivated by the observation that imposing a meaningful graph topology can help solving the generally ill-posed and challenging problem of structure inference. To capture the temporal evolution in the sequence of graphs, we introduce a new prior that asserts that the graph edges change smoothly in time. We propose a primal-dual optimization algorithm that scales linearly with the number of allowed edges and can be easily parallelized. Our new algorithm is shown to outperform standard graph learning and other baseline methods both on a synthetic and a real dataset.

Keywords: Graph learning, time varying graph, network inference, covariance estimation, graph quality.

1. INTRODUCTION

Graphs are invaluable tools for data analysis, as they can encode the complex structure inherent to many high-dimensional datasets. The importance of the graph construction in data analysis and processing has led to many recent works about graph learning [1, 2, 3, 4, 5, 6, 7, 8, 9]. The motivation of *graph learning* or *network topology inference* is dual. First, by construction many datasets in network science possess graph structure. *Inferring the topology often provides deeper insights into the inner-workings of complex networks*, such as recommendation, biological, social, and financial networks. Second, even when a data matrix does not come from a "tangible" graph, its structure is very well captured by a learned graph when few samples are available.

The goal of this paper is to extend graph learning for these applications where the inherent data structure is time-varying. From recommendation to social, biological, financial, and sensor networks, time is an essential aspect of many of the systems that can be analyzed with graphs. Yet, though the importance of the time-dimension is recognized for graph signal analysis [10, 11, 12, 13], time has been largely excluded from the analysis of the graph itself. In fact, there is a fundamental trade-off between the temporal resolution and the number of samples available for learning. Depending on the speed with which the hidden structure changes, we might have only very few samples available that correspond to the same or almost the same distribution at a given point in time. Here, instead of following classical graph learning and inferring an average graph capturing the common structure between all nodes (therefore ignoring the time dimension), we propose to learn a sequence of graphs, one per time-window. To regularize learning and improve performance, we introduce a new prior which asserts that the data structure changes slowly in time. Experiments in diverse settings confirm the benefits of our solution that outperforms baseline solutions from the perspective of a new metric specifically designed to capture the accuracy of the graph estimates.

Concretely, our contributions are summarized as follows:

1) We propose a novel framework for time-varying graph learning under smoothness assumptions (Sec. 2). The framework enhances known graph learning models by imposing an additional prior that takes into account the variation of edge weights. An efficient primal-dual optimization procedure based on [7, 14] allows us to scale our algorithm: the algorithm presented scales *linearly* in the size of the variables, i.e., the number of different graphs we learn times the number of allowed edges in each of them.

2) To obtain a measure of comparison, we ask a fundamental question: *How well does a graph fit a data matrix or distribution?* To this end, we bring forth TCER, which, to the best of our knowledge, is the first measure capturing the quality of a graph with respect to a set of graph signals (Sec. 3.1). Our criterion relates a graph to a data distribution and quantifies the quality of the obtained graph basis in comparison to the theoretically best basis.

The rest of the paper is organized as follows. We present the graph learning problem in Section 2 along with a description of our new optimization algorithm. Section 3 first proposes a new evaluation criteria for graph estimation, and then presents experimental results on synthetic as well as realistic visual data.

2. LEARNING TIME VARYING GRAPHS

2.1. Smooth Data on Static Graphs

The representation of data living on graphs permits to capture explicitly the dependency between data samples through the graph connections. In particular, a widely used assumption about data residing on graphs is that the values change smoothly across adjacent nodes. The smoothness of a set of vectors $x_1, \dots, x_T \in \mathbb{R}^N$ on a given weighted undirected graph is usually quantified by the *Dirichlet energy* [15]

$$\frac{1}{2}\sum_{i,j} W_{ij}\|x_i - x_j\|^2 = \operatorname{tr}(X^\top L X), \tag{1}$$

where $W_{ij} \in \mathbb{R}_+$ denotes the weight of the edge between nodes i and j, $L = D - W$ is the graph Laplacian, and $D_{ii} = \sum_j W_{ij}$ are the entries of the diagonal degree matrix D. Regularization using the Dirichlet energy has been used extensively, to enhance for example image processing [16, 17], non-negative matrix factorization [18, 19], matrix completion [20], or principal component analysis (PCA) [21, 22, 23]. In most of these works the graph is known, and the Dirichlet energy is minimized with respect to (w.r.t.) the observation matrix X. However, when the graph is not known, the same energy can be minimized w.r.t. to L, or equivalently W, in order to learn a graph under the assumption that X is smooth on it.

It leads to the following general formulation of the (static) graph learning problem:

$$\min_{W \in \mathcal{W}} \|W \circ Z\|_{1,1} + f(W), \tag{2}$$

where $Z_{ij} = \|x_i - x_j\|^2$, the first term using the Hadamard product $\circ$ is equal to $\mathrm{tr}(X^\top L X)$ [7], and $\mathcal{W}$ denotes the set of valid adjacency matrices (positive and symmetric). The role of the matrix function $f(W)$ is to prevent W from obtaining a trivial zero value, control sparsity, and impose further structure. State-of-the-art methods assume different models for $f(W)$. In particular, Kalofolias [7] defines

$$f(W) = -\alpha \mathbf{1}^\top \log(W\mathbf{1}) + \frac{\alpha}{2}\|W\|_F^2, \tag{3}$$

for $\mathbf{1} = [1, \dots 1]^\top$, while Hu et al. [5] and Dong et al. [6] impose

$$f(W) = \alpha \|W\mathbf{1}\|^2 + \alpha \|W\|_F^2 + \mathbb{1}\{\|W\|_{1,1} = N\}, \tag{4}$$

where $\mathbb{1}\{\text{condition}\} = 0$ if condition holds, ∞ otherwise. Since $W\mathbf{1}$ is the vector with the node degrees, the first model prevents the formation of disconnected nodes due to the logarithmic barrier, while the second one controls sparsity by penalizing the formation of big degrees due to the first term. The choice therefore depends on the data and application in question.

2.2. Smooth Data on Time-Varying Graphs

Next, we consider the case of a graph that changes slowly over time. We denote our data matrix $X \in \mathbb{R}^{N \times T}$, containing columns x_k as time samples of a time-varying graph with N nodes. To make the problem tractable, we further discretize time in K windows. We denote by $W^{(k)}$ for $k = 1, \dots, K$ the adjacency matrix of the k-th window. Obviously, $K \leq T$ and for simplicity we suppose that the size of each time window is the same.

Then, we enhance the general graph learning problem of (2) with a term that penalizes fast changes of the adjacency matrices. To facilitate optimization, we use a differentiable term that associates each adjacency matrix with its previous one. The graph learning problem for time varying graphs can then be formulated as follows

$$\min_{\{W^{(k)} \in \mathcal{W}\}} \sum_{k=1}^{K} \left[\left\| W^{(k)} \circ Z^{(k)} \right\|_{1,1} + f(W^{(k)}) \right] + \cdots \\ \gamma \sum_{k=2}^{K} f_{\text{time}} \left(W^{(k)}, W^{(k-1)} \right), \tag{5}$$

where $f(W^{(k)})$ can be any of the models in Eq. (3) or Eq. (4).

Above, $Z^{(k)}$ denotes the squared pairwise distances, computed only from the columns of X that correspond to the time window k and $f(W^{(k)})$ is defined as in (2). Moreover, f_{time} is a dissimilarity measure between the adjacency matrix at time window k and the one of time window $k-1$. We choose f_{time} to be a Tikhonov smoothness prior defined as $\|W^{(k)} - W^{(k-1)}\|_F^2$, which enforces that the graph edges change smoothly over time. Note that one could choose different priors depending on the application settings, like a Total Variation (TV) prior for graphs that are expected to have switching edges, for example.

Finally, it is worthwhile to remark on the role of the newly introduced objective function parameter. First, the value of the positive constant γ should be tied to how slowly the learned graphs change over time. By increasing γ, we achieve a continuum of solutions, from learning each graph independently (fast change) for $\gamma = 0$, to learning K times the 'average' graph (no change) for $\gamma \to \infty$.

2.3. Primal Dual Optimization Framework

The graph learning problem of (5) can be solved using the primal-dual based algorithms presented in [14]. Furthermore, these algorithm can be run in a parallel fashion: in our scheme the code for learning each graph is executed in a separate processing unit and only units at consecutive time-steps need common memory access.

Due to space limitations we only outline the main idea and the changes over previous models, and refer the reader to [7, 24] for more details. The general idea of the algorithm is to split the objective function (5) as $f = f_1 + f_2 + f_3$, and iterate between approximately minimizing f_1, f_2, f_3 in a round robin fashion, switching between primal and dual domains when needed. We present below the case where $f(W^{(k)})$ follows the log-degree model of Eq. (3) (the case of the model of Eq. (4) can be developed similarly). If we denote by $w^{(k)}$ the vector forms of $W^{(k)}$, the splitting of the objective function can be written as

$$f_1(w^{\text{all}}) = \mathbb{1}\{w^{\text{all}} \geq 0\} + 2{w^{\text{all}}}^\top z^{\text{all}}, \\ f_2(d^{\text{all}}) = -\mathbf{1}^\top \log(d^{\text{all}}), \\ f_3(w^{\text{all}}) = \|w^{\text{all}}\|_2^2 + \gamma \sum_{t=2}^{T} f_{\text{time}}^t \left(w^{(k)}, w^{(k-1)} \right),$$

where $w^{\text{all}} = [w^{(1)}, \dots, w^{(K)}]$, and similarly for the distances z^{all} and the degrees d^{all}. The indicator function $\mathbb{1}\{w^{\text{all}} \geq 0\}$ ensures that the learned graph has positive weights, whereas ${w^{\text{all}}}^\top z^{\text{all}}$ is equivalent to the Dirichlet energy term in (2).

The first two functions f_1, f_2 are non-smooth, the first in the primal and the second in the dual domain. Since the gradient steps are not well defined for them, we approximately minimize them using their proximal operators, that are the same as the ones of [7]. The last term f_3 contains our differentiable time-prior for which we need to provide the gradient.

The proximal operators of f_1 and f_2 can be computed in parallel for each window independently, and as we show next the same holds for the gradient step of f_3. Furthermore, to switch from the primal (weights) domain to the dual (degrees) domain, we need to use the linear operator S (see [7]) for each window independently, i.e., $d^{(k)} = S w^{(k)}$. Equivalently, for the space containing weights of all windows, we use the linear operator $\mathrm{diag}\,(S, \dots, S)$, that is a block diagonal operator with K repetitions of S in the diagonal.

The gradient of f_3 is easy to compute:

$$\nabla f_3(w^{\text{all}}) = 2w^{(k)} + \gamma \nabla f_{\text{time}}^k + \gamma \nabla f_{\text{time}}^{k+1},$$

where, for compactness, we write ∇ and f_{time}^k instead of $\nabla_{w^{(k)}}$ and $f_{\text{time}}^k \left(w^{(k)}, w^{(k-1)} \right)$. When f_{time} represents a Tikhonov prior, the gradient becomes

$$\nabla_{w^{(k)}} f_3(w^{\text{all}}) = (2 + 4\gamma) w^{(k)} - \gamma (w^{(k-1)} + w^{(k+1)}).$$

We see that in order to compute the gradient for a window k, we need to use *message passing*, from neighboring windows $k-1$ and $k+1$. The computation of this gradient is the only operation where information from neighboring windows is exchanged during the primal-dual algorithm.

The per iteration computational complexity of the algorithm for N nodes and K windows is $\mathcal{O}(N^2 K)$, and the number of iterations is typically within the hundreds.

3. PERFORMANCE EVALUATION

3.1. TCER: Measuring the Quality of a Graph

We first propose a new metric for the quality of a graph. The rationale behind this is that, when a data matrix X contains a few samples x_t of a distribution, we should measure how well the learned graph fits the whole distribution and not only the samples X we used to infer it.

We first observe that one of the primary uses of the Laplacian of a graph is that it provides the *graph Fourier transform matrix* [25, 26], defined as its eigenvector matrix. Furthermore, a smooth signal is well summarized by the first few eigenvectors [23, 27]. Summarizing data using only the first few vectors of an orthogonal basis is the idea behind the success of singular vector decomposition (SVD) and PCA, as the first for example minimizes the Frobenius norm (or energy) of the approximation error using the first R singular vectors of a matrix. Naturally, when seeking a good basis Q for signals X, *we want to quantify their ability to explain most of the data energy with as few basis vectors as possible*.

We can quantify how much energy of a matrix $X \in \mathbb{R}^{N\times T}$ is carried by the first R vectors of an *ordered* orthogonal basis $Q = [q_1, \dots q_N]$ with the *data cumulative energy* $\mathcal{S}_R\{X, Q\} = \sum_{r=1}^{R} \|q_r^\top X\|^2$, that is maximized by the left singular vector basis for any given R. Note that maximizing $\mathcal{S}_R\{X, Q\}$ minimizes the error achieved when projecting the data X on the first R vectors of Q [24, Section 1.4]. In order for our measure not to depend on a specific approximation rank R, we summarize the compressibility of a dataset on a given basis Q over all possible ranks from 1 to N.

Definition 1. ***Total cumulative energy of data on a basis Q:*** *Given a data matrix $X \in \mathbb{R}^{N\times T}$ and a sorted orthogonal basis $Q = [q_1, \dots q_N]$, the total cumulative energy of the data X is*

$$\mathcal{T}\{X, Q\} = \sum_{R=1}^{N} \mathcal{S}_R\{X, Q\} = \sum_{r=1}^{N} (N + 1 - r)\|q_r^\top X\|^2.$$

Note that the maximum total cumulative energy of a matrix X is also achieved by its left singular vector matrix. If the columns of X follow a *distribution* $p(X)$ with mean μ and covariance C, the *expected cumulative energy* is

$$\mathbb{E}\left[\mathcal{S}_R\{X, Q\}\right] = \mathbb{E}\left[\sum_{r=1}^{R} \left\|q_r^\top X\right\|^2\right] = \sum_{r=1}^{R} q_r^\top \mathbb{E}\left[XX^\top\right] q_r$$

$$= \sum_{r=1}^{R} q_r^\top \left(C + \mu\mu^\top\right) q_r = \sum_{r=1}^{R} \left\|q_r^\top U\Sigma\right\|^2 = \mathcal{S}_R\left\{U\Sigma, Q\right\},$$

where we use the eigenvalue decomposition $C + \mu\mu^\top = U\Sigma^2 U^\top$. The maximum *expected* total cumulative energy is now achieved by the left singular basis U of $\mathbb{E}\left[XX^\top\right]$, which, for zero-mean data, coincides with the Karhunen-Loève basis.

While the SVD basis (that is the PCA basis for a zero-mean matrix) of X explains best its samples (columns), when the number of samples is insufficient, it might fail to explain well other samples of the same distribution. To quantify this compression inefficiency, we propose TCER:

Definition 2. ***Total cumulative energy residual (TCER):*** *Given a data distribution $p(X)$ with mean μ and covariance C, and a sorted orthogonal basis $Q = [q_1, \dots q_N]$, the residual of the total cumulative energy is*

$$\mathcal{R}\left\{p(X), Q\right\} = 1 - \frac{\mathbb{E}\left[\mathcal{T}\{X, Q\}\right]}{\max_{Q\in\mathbb{O}} \mathbb{E}\left[\mathcal{T}\{X, Q\}\right]} = 1 - \frac{\mathcal{T}\{U\Sigma, Q\}}{\mathcal{T}\{U\Sigma, U\}},$$

where we use the eigenvalue decomposition $C + \mu\mu^\top = U\Sigma^2 U^\top$ and the denominator of TCER is simply $\sum_{R=1}^{N}\sum_{r=1}^{R} \Sigma_{rr}^2$.

In words, TCER measures how well an orthogonal matrix Q summarizes the expected energy $\mathbb{E}\left[XX^\top\right]$ of data distribution $p(X)$ by its first few vectors. For a more detailed analysis of TCER and its relation to the energy compaction property, see [24, Section 1.4].

3.2. Gaussian Markov Chain Graph Estimation

We now evaluate the performance of our dynamic graph learning algorithm on synthetic data. Modeling graph data using a Gaussian MRF given by the distribution

$$p\left(x_t|W\right) = \mathcal{N}\left(x_t|0, \left(L + \sigma^2 I\right)^{-1}\right),$$

as done in [28] is convenient, as we can easily add time structure. Given a set of T slowly changing graphs $W^{(t)}$ with N nodes, we can model time varying data as samples from the Gaussian Markov chain

$$p\left(x_t|x_{t-1}, W^{(t)}\right) = \mathcal{N}\left(x_t|0, \left(L^{(t)} + \sigma^2 I\right)^{-1}\right)\mathcal{N}\left(x_t|x_{t-1}, \frac{1}{\mu} I\right),$$

where the second distribution adds a Gaussian dependence between consecutive samples. Arranging samples in a matrix $X \in \mathbb{R}^{N\times T}$, we recover the graph structure $W^{(t)}$ with our time varying graph model. As the dependence between x_t and x_{t+1} is Gaussian, we use an ℓ-2 distance $f_{\text{time}}(w^{(k)}, w^{(k-1)}) = \|w^{(k)} - w^{(k-1)}\|_2^2$. For this experiment we use two types of time varying graphs:

Erdős Rényi : Starting from a random binary Erdős Rényi graph [29] $W^{(1)}$, we sample each subsequent graph $W^{(t+1)}$ by keeping 95% of the edges of $W^{(t)}$ intact, and 5% of them re-sampled from the same distribution that gave $W^{(t)}$. In total, we sample 700 such graphs of 50 nodes, and we use a probability of connection $p = 5/(N-1)$.

Random waypoint geometric graphs: We simulate 50 sensors moving around the 2 dimensional square space $[0, 20]^2$ (meters) according to the mobility model described by [30], commonly called the *random waypoint* model. By assuming each sensor moves with an average speed of 0.05-to-0.5 *m/s*, we sample their positions every 0.1 *s* for a total time of 70 *s*. Given their positions, we construct a total of 700 ϵ-neighborhood graphs with exponential decaying weights.

Quality criterion: We measure the average over all windows TCER achieved by the eigenvalue decomposition of the graph Laplacian, averaged over all samples of a given window. For example, the error of a given algorithm for the first window ($k = 1$) of size M is $\frac{1}{M}\sum_{m=1}^{M} \mathcal{R}(p(x_m)|Q^{(1)})$, where each marginal distribution $p(x_m)$ is "explained" by the same basis $Q^{(1)}$ obtained as the eigenvalue decomposition of the learned graph Laplacian (or empirical covariance) of that window.

In Fig. 1 we see the graph recovery quality for different window sizes. For the Erdős Rényi graphs, we use the log-degree model of Eq. (3), that performed best, while for the random waypoint data we use the ℓ_2 degree model of Eq. (4), that allows the appearance of disconnected nodes. We remark that the parameters of all presented methods (i.e., that corresponding to graph sparsity) have been exhaustively optimized for best overall performance. Focusing only on the covariance estimation that sets the baseline, we see that because the distribution changes in time, the sample covariance estimation of all samples (**dotted black line**) is a worse estimator than the sample covariance with only half of the samples (minimum of **dotted green line**). This justifies the need to split the data in more than one coherent blocks. However, by splitting data in more blocks, we keep fewer vectors in each of them, making covariance estimation worse.

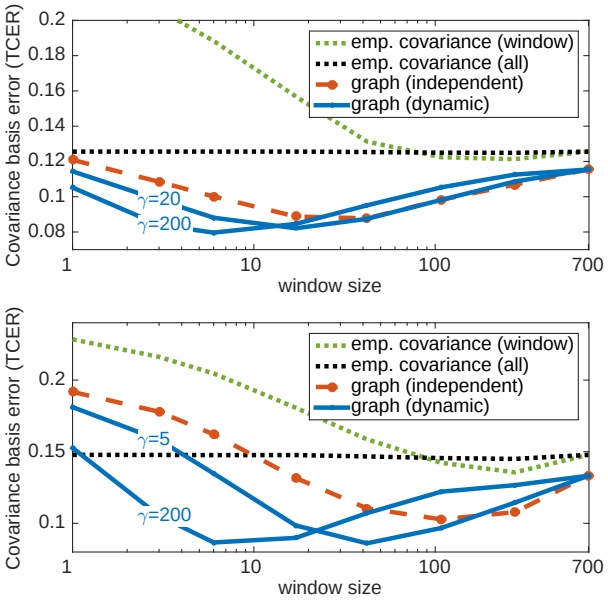

Fig. 1: Quality of time varying graphs for different window sizes on matrix $X \in \mathbb{R}^{50\times 700}$ from a Markov chain. **Up:** Time varying Erdős Rényi graph. **Down:** Random waypoint geometric graph.

As suggested by Fig. 2, this is less of a problem when we learn a graph instead.

The **dashed red line** corresponds to learning a graph from each window independently ($\gamma = 0$), that performs better than the covariance matrix with fewer samples, and can thus achieve greater precision in time. Robustness to few samples is stronger when we add our new time smoothness prior (**blue lines**), achieving the best result for $\gamma = 200$. In this case, graph learning is very robust to having very few samples per window (only 6). As the complexity is linear to the number of time windows, one might choose to have less windows with more samples each, in which case a smaller γ is more appropriate.

3.3. Structure Estimation in Visual Data

We evaluate now the performance of our algorithm in illustrative problems of structure estimation in visual data. First, we want to confirm the benefits of graph learning discussed in Section 2.1 in realistic settings. We propose an experiment on simple handwritten digits images from the MNIST image dataset with the objective of estimating a graph that captures the pixel-wise structure. Fig. 2 shows the accuracy of the graphs obtained by a) graph learning [7], b) empirical covariance estimation, and c) the more computationally expensive sparse inverse covariance estimator[1] also known as graphical LASSO [31]. We see that *unless the number of samples (images) is large, learning a graph "explains" the data structure (variance) better*, which confirms the potential of graph learning approaches.

Next, we evaluate our dynamic graph learning framework on point cloud denoising. When capturing point cloud information, the geometric coordinates are usually inaccurate, while noise is further introduced by point registration error. Furthermore, in the case of dynamic point clouds, the geometry evolves over time. To denoise

[1]Inverse covariance estimation methods struggle to reduce the cost from $\mathcal{O}(n^3)$ with n nodes, while the cost of graph learning is naturally $\mathcal{O}(n^2)$ [7].

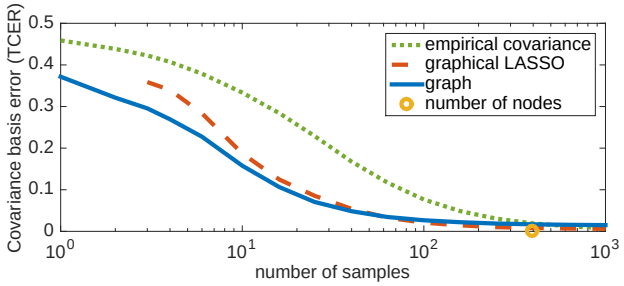

Fig. 2: Quality of different orthogonal bases for MNIST. Unless the number of training samples (images) is large, learning a graph explains the data structure better.

the geometric coordinates Y of a point cloud, we follow the example of manifold denoising with Dirichlet regularization [16] and solve a graph smoothing problem $\min_X \frac{1}{2}\|X - Y\|_F^2 + \frac{\alpha}{2}\operatorname{tr}\left(X^\top L X\right)$, where L is the graph Laplacian computed from Y. The choice of L affects the denoising result significantly.

For our experiment, we use the dynamic point cloud of a dancer [32], downsampled to 214 points and evolving over 240 time frames. The initial data is nearly perfectly registered (using multiview video techniques), which is convenient so that we have a noiseless ground truth. We add Gaussian noise and simulate registration error, by allowing the resampling of any point from the initial set of 1500 points with a probability of 1/500 between consecutive frames.

	Noisy data	1 graph (static)	12 graphs (independent)	12 graphs (dynamic)
SNR (dB)	14.03	17.58	18.77	**18.95**

Table 1: Quality of point cloud denoising with different graph types.

In Table 1, we report the results of denoising by imposing smoothness through the log-degree model of Eq. (3) and a Tikhonov prior over time, on different choices of the graph Laplacian. Capturing the structure of all the frames with only one graph does not perform well, as the point cloud structure changes due to the dynamic nature and the registration error. While learning only one graph yields an SNR of 17.58, by learning 12 different graphs of 20 frames each, we achieve an SNR of 18.77. By imposing further time structure with virtually no extra computational cost, we achieve the best denoising with SNR of 18.95 using our new model.

4. CONCLUSION

This paper provides a new framework for learning dynamically changing graphs from smooth data observations. We split time in consecutive windows and learn a sequence of graphs, one for each of them. A time smoothness prior is added to our model, which imposes that the graphs learnt in successive windows change smoothly over time. We can achieve a good trade-off between temporal resolution and computation cost. The latter is linear with the number of windows and the number of allowed edges. The proposed optimization problem is solved efficiently with an easy to parallelize primal dual algorithm. Our experiments show that the new model can outperform classical graph learning and other baseline methods, both on artificial and real data, in terms of graph accuracy measured with a newly designed metric.

5. REFERENCES

[1] Fei Wang and Changshui Zhang, "Label propagation through linear neighborhoods," *Trans. on Knowledge and Data Engineering*, vol. 20, no. 1, 2008. 1

[2] Samuel I Daitch, Jonathan A Kelner, and Daniel A Spielman, "Fitting a graph to vector data," in *ACM 26th Annual Intl Conference on Machine Learning*, 2009. 1

[3] Tony Jebara, Jun Wang, and Shih-Fu Chang, "Graph construction and b-matching for semi-supervised learning," in *ACM 26th Annual Intl Conference on Machine Learning*, 2009. 1

[4] Brenden Lake and Joshua Tenenbaum, "Discovering structure by learning sparse graph," in *33rd Annual Cognitive Science Conference*, 2010. 1

[5] Chenhui Hu, Lin Cheng, Jorge Sepulcre, Georges El Fakhri, Yue M Lu, and Quanzheng Li, "A graph theoretical regression model for brain connectivity learning of alzheimer's disease," in *IEEE 10th Intl Symposium on Biomedical Imaging*, 2013. 1, 2

[6] Xiaowen Dong, Dorina Thanou, Pascal Frossard, and Pierre Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. on Signal Processing*, vol. 64, no. 23, 2016. 1, 2

[7] Vassilis Kalofolias, "How to learn a graph from smooth signals," in *19th Intl Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016. 1, 2, 4

[8] Santiago Segarra, Antonio G. Marques, Gonzalo Mateos, and Alejandro Ribeiro, "Network topology identification from spectral templates," *arXiv preprint arXiv:1604.02610*, 2016. 1

[9] Hilmi E. Egilmez, Eduardo Pavez, and Antonio Ortega, "Graph learning from data under structural and Laplacian constraints," *arXiv preprint arXiv:1611.05181*, 2016. 1

[10] Andreas Loukas and Damien Foucard, "Frequency analysis of temporal graph signals," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2016. 1

[11] Andreas Loukas and Nathanaël Perraudin, "Stationary time-vertex signal processing," *arXiv preprint arXiv:1611.00255*, 2016. 1

[12] Jonathan Mei and José M. F. Moura, "Signal processing on graphs: Modeling (causal) relations in big data," *arXiv preprint arXiv:1503.00173*, 2015. 1

[13] Elvin Isufi, Andreas Loukas, Andrea Simonetto, and Geert Leus, "Separable autoregressive moving average graph-temporal filters," in *IEEE 24th European Signal Processing Conference (EUSIPCO)*, 2016. 1

[14] Nikos Komodakis and Jean-Christophe Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *arXiv preprint arXiv:1406.5429*, 2014. 1, 2

[15] Mikhail Belkin and Partha Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Neural Information Processing Systems (NIPS)*, 2001, vol. 14. 1

[16] Abderrahim Elmoataz, Olivier Lezoray, and Sébastien Bougleux, "Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing," *Trans. on Image Processing*, vol. 17, no. 7, 2008. 1, 4

[17] Miao Zheng, Jiajun Bu, Chun Chen, Can Wang, Lijun Zhang, Guang Qiu, and Deng Cai, "Graph regularized sparse coding for image representation," *IEEE Trans. on Image Processing*, vol. 20, no. 5, 2011. 1

[18] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang, "Graph regularized nonnegative matrix factorization for data representation," *Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, 2011. 1

[19] Kirell Benzi, Vassilis Kalofolias, Xavier Bresson, and Pierre Vandergheynst, "Song recommendation with non-negative matrix factorization and graph total variation," in *IEEE Intl Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016. 1

[20] Vassilis Kalofolias, Xavier Bresson, Michael Bronstein, and Pierre Vandergheynst, "Matrix completion on graphs," *arXiv preprint arXiv:1408.1717*, 2014. 1

[21] Bo Jiang, Chibiao Ding, Bio Luo, and Jin Tang, "Graph-laplacian pca: Closed-form solution and robustness," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2013. 1

[22] Nauman Shahid, Vassilis Kalofolias, Xavier Bresson, Michael Bronstein, and Pierre Vandergheynst, "Robust principal component analysis on graphs," in *Proceedings of the Intl Conference on Computer Vision*, 2015. 1

[23] Nauman Shahid, Nathanael Perraudin, Vassilis Kalofolias, Gilles Puy, and Pierre Vandergheynst, "Fast robust PCA on graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, 2016. 1, 3

[24] Vassilis Kalofolias, *From data to structures: graph learning under smoothness assumptions and applications in data science*, Ph.D. thesis, École Polytechnique Fédérale de Lausanne, 2016. 2, 3

[25] Fan RK Chung, *Spectral graph theory*, vol. 92, American Mathematical Soc., 1997. 3

[26] David Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, Pierre Vandergheynst, et al., "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, 2013. 3

[27] Nathanaël Perraudin and Pierre Vandergheynst, "Stationary signal processing on graphs," *arXiv preprint arXiv:1601.02522*, 2016. 3

[28] Cha Zhang, Dinei Florêncio, and Philip A Chou, "Graph signal processing–a probabilistic framework," *Technical Report:MSR-TR-2015-31*, 2015. 3

[29] Edgar N Gilbert, "Random graphs," *The Annals of Mathematical Statistics*, 1959. 3

[30] David B Johnson and David A Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile computing*. Springer, 1996. 3

[31] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *The Journal of Machine Learning Research*, vol. 9, 2008. 4

[32] Juergen Gall, Carsten Stoll, Edilson De Aguiar, Christian Theobalt, Bodo Rosenhahn, and Hans-Peter Seidel, "Motion capture using joint skeleton tracking and surface estimation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2009. 4