GRAPH LEARNING UNDER SPARSITY PRIORS

Hermina Petric Maretic, Dorina Thanou and Pascal Frossard

Signal Processing Laboratory (LTS4), EPFL, Switzerland

ABSTRACT

Graph signals offer a very generic and natural representation for data that lives on networks or irregular structures. The actual data structure is however often unknown a priori but can sometimes be estimated from the knowledge of the application domain. If this is not possible, the data structure has to be inferred from the mere signal observations. This is exactly the problem that we address in this paper, under the assumption that the graph signals can be represented as a sparse linear combination of a few atoms of a structured graph dictionary. The dictionary is constructed on polynomials of the graph Laplacian, which can sparsely represent a general class of graph signals composed of localized patterns on the graph. We formulate a graph learning problem, whose solution provides an ideal fit between the signal observations and the sparse graph signal model. As the problem is non-convex, we propose to solve it by alternating between a signal sparse coding and a graph update step. We provide experimental results that outline the good graph recovery performance of our method, which generally compares favourably to other recent network inference algorithms.

Index Terms— graph learning, graph signal processing, Laplacian matrix, sparse signal prior, graph dictionary

1. INTRODUCTION

Graphs provide a flexible tool for modelling and manipulating complex data that resides on topologically complicated domains such as transportation networks, social, and computer networks, brain analysis or even digital images. Typically, once a graph is well-defined, classical data analysis and inference tasks can be effectively performed by using tools such as spectral graph theory or generalization of signal processing notions in the graph domain [1]. However, it is often the case that the graph structures are not defined a priori or the intrinsic relationships between different signal observations are not clear or are defined only in a very subtle manner. Since the definition of a meaningful graph plays a crucial role in the analysis and processing of structured data, the inference of the graph itself from the data is a very important problem that has not been well investigated so far.

The definition of the graph has been initially studied in a machine learning framework with simple models such as K-nearest neighbour graphs and other heuristically defined graph kernels [2], [3] or convex combinations of them [4]. Richer adaptivity to the data is obtained by relating the graph structure more closely to data properties in order to infer a graph topology. Techniques such as sparse inverse covariance estimations [5], [6] rely on Gaussian graphical models to identify partial correlations between random variables and define graph edges. Such techniques have also been extended to very large graphs, by learning highly structured representations [7]. More recent works relax the assumption of a full rank precision matrix by infering the graph topology from a graph signal processing perspective [8], [9], [10], [11] under explicit graph signal smoothness assumptions. The above works assume that the data globally evolves smoothly on the underlying structure. However, such a model might not be very precise for many real world datasets, which can feature highly localized behaviors or piecewise smoothness. The recent framework in [12] that observes graph signals as white signals filtered with a graph shift operator polynomial, is one of the first network inference works to depart from explicit global smoothness assumptions. A similar idea uses adjacency matrix polynomials to model causation in time-varying signals [13], [14].

In this work, we consider a generic model where the graph signals are represented by (sparse) combinations of overlapping local patterns that reside on the graph. That is, given a set of graph signals, we model these signals as a linear combination of only a few components (i.e., atoms) from a graph structured dictionary that captures localized patterns on the graph. We incorporate the underlying graph structure into the dictionary through the graph Laplacian operator. In order to ensure that the atoms are localized in the graph vertex domain, we further impose the constraint that our dictionary is a concatenation of subdictionaries that are polynomials of the graph Laplacian [15]. Based on this generic model, we cast a new graph learning problem that aims at estimating a graph that explains the data observations, which should eventually form a sparse set of localized patterns on the learned graph. We propose an alternating optimization algorithm to address the resulting non-convex inverse problem, which is based on a sparse coding step obtained with orthogonal matching pursuit (OMP) and a graph learning step performed by applying a projected gradient descent step. We finally provide a few illustrative experiments on synthetic data, and we show that our generic graph signal model leads to better graph recovery performance than state-of-the-art algorithms.

The rest of the paper is organized as follows. We describe our graph learning framework in Section 2. We then present our alternating optimization algorithm in Section 3, and evaluate the graph recovery performance in Section 4.

978-1-5090-4117-6/17/$31.00 ©2017 IEEE

2. GRAPH LEARNING FRAMEWORK

2.1. Sparse signal representation on graphs

We first present the graph signal processing framework used in this work. We consider an undirected, weighted graph $G = (V, E, W)$ with a set of N vertices V, edges E and a weighted adjacency matrix W, with the weight value $W_{ij} = 0$ if there is no edge between i and j. We define a signal on the graph G as a function $y : V \to \mathbb{R}$, where $y(v)$ denotes the value of a signal on a vertex v. One of the most useful graph operators that is widely used in graph signal processing tasks, is the normalized Laplacian operator, defined as

$$\mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}},$$

where I is the identity matrix and D is the diagonal degree matrix. This graph operator has a complete set of orthonormal eigenvectors $\chi = \{\chi_0, \chi_1, ..., \chi_{N-1}\}$ with a corresponding set of non-negative eigenvalues. These eigenvectors form a basis for the definition of the graph Fourier transform [1], which provides a spectral representation of graph signals.

Similarly to classical signal processing, one can design an overcomplete dictionary $\mathcal{D}$ for signals on graphs, such that every graph signal y can be represented as a sparse linear combination of dictionary atoms, i.e., $y \approx \mathcal{D}x$, where x is a sparse vector of coefficients. In order to obtain an effective representation of graph signals, the dictionary has to incorporate the structure of the graph. In particular, we consider here a polynomial graph structured dictionary, which has been shown to provide sparse representations of a general class of graph signals that are linear combinations of graph patterns positioned at different vertices of the graph [15]. The dictionary $\mathcal{D} = [\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_S]$ is defined as a concatenation of S subdictionaries of the form

$$\begin{aligned}\mathcal{D}_s = \hat{g}_s(\mathcal{L}) &= \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k \\ &= \sum_{k=0}^{K} \alpha_{sk} (I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}})^k,\end{aligned}$$

where $\hat{g}_s(\cdot)$ is the generating kernel of the subdictionary $\mathcal{D}_s$. In this particular signal model, the graph Laplacian captures the connectivity of the graph, while the polynomial coefficients reflect the distribution of the signal in particular neighbourhoods of the graph. Furthermore, the behaviour of these kernels in graph eigenvalues describes the nature of atom signals in the graph. Namely, a kernel promoting low frequency components will result in smooth atoms on the graph. The polynomial coefficients, together with the graph Laplacian, fully characterize the dictionary. In the rest of this paper, we consider the general class of graph signals that have sparse representation in the dictionary $\mathcal{D}$.

2.2. Problem formulation

Equipped with the sparse graph signal model defined above, we can now formulate our graph learning problem. In particular, given a set of signal observations $Y = [y_1, y_2, ..., y_M] \in \mathbb{R}^{N \times M}$, we want to infer a graph G, such that the observed signals have a sparse representation in the graph dictionary built on G.

More formally, the graph learning problem can be cast as follows:

$$\begin{aligned} \underset{W,X}{\arg\min} \quad & ||Y - \mathcal{D}X||_F^2 + \beta_W ||W||_1 \\ \text{subject to} \quad & \mathcal{D} = [\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_S], \\ & \mathcal{D}_s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k, \forall s \in \{1, ..., S\} \\ & \mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \\ & W_{ij} = W_{ji} \geq 0, \forall i, j, i \neq j \\ & W_{ii} = 0, \forall i \\ & ||x_m||_0 \leq T_0, \forall m \in \{1, ..., M\} \end{aligned} \tag{1}$$

where T_0 is the sparsity level of the coefficients of each signal, β_W is a parameter that controls the graph sparsity i.e., the number of non-zero edges of the graph, through the L_1 norm of the graph adjacency matrix W, and x_m is the m^{th} column of the matrix X. The optimization is performed over the weight matrix W instead of $\mathcal{L}$ that is used explicitly in the dictionary construction, as the constraints defining a valid weight matrix W are much simpler to handle than those defining a valid Laplacian. Namely, a valid $\mathcal{L}$ must be positive semi-definite, while the weight matrix assumes only symmetry and non-negativity of the weights.

Finally, we assume in this work that the dictionary kernels, i.e., the coefficients α_{sk}, are known. We can for example model these generating kernels $\hat{g}_s(\cdot)$ as graph heat kernels and compute the polynomial coefficients α_{sk} as a K-order Taylor approximation coefficients. Other kernel choices are possible, such as spectral graph wavelets [16], where the polynomial coefficients could be inferred from a Chebyshev polynomial approximation, or a priori learned kernels [15]. For the sake of simplicity, we also consider the S and K are determined by a priori information about the nature of the target application, or optimized separately.

3. GRAPH LEARNING ALGORITHM

Next, we discuss the solution obtained by our graph learning framework. As the optimization problem (1) is non-convex, we solve the problem by alternating between the sparse coding and the weight matrix update steps, which is a widely used techniques for solving ill-posed non-convex problems.

In the first step, we fix the weight matrix W and optimize the objective function with respect to the sparse codes X, which leads to the following optimization problem

$$\begin{aligned} & \underset{X}{\arg\min} ||Y - \mathcal{D}X||_F^2 \\ & \text{subject to } ||x_m||_0 \leq T_0, \forall m \in \{1, ..., M\}. \end{aligned}$$

The L_0-"norm" constraint ensures sparsity of the sparse codes. We solve the above problem using orthogonal matching pursuit (OMP) [17]. Before updating X, we normalize the atoms of the dictionary $\mathcal{D}$ to be of unit norm. This step is essential for the OMP step in order to treat all atoms equally. To recover our initial structure, after computing X, we renormalize the atoms of our dictionary and the sparse coding coefficients in such a way that the product $\mathcal{D}X$ remains constant.

In the second step, we fix the sparse codes and we update the dictionary, i.e., the graph. Estimating the weight matrix with fixed sparse codes X, however, remains non-convex, as the dictionary $\mathcal{D}$ is constructed from a K-order polynomials of $\mathcal{L}$, and thus of W. The optimization problem becomes:

$$\begin{aligned}
&\arg\min_{W} ||Y - \mathcal{D}X||_F^2 + \beta_W ||W||_1 \\
&\text{subject to} \quad \mathcal{D} = [\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_S], \\
&\qquad \mathcal{D}_s = \sum_{k=0}^{K} \alpha_{sk} \mathcal{L}^k, \forall s \in \{1, ..., S\} \\
&\qquad \mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \\
&\qquad W_{ij} = W_{ji} \geq 0, \forall i, j, i \neq j \\
&\qquad W_{ii} = 0, \forall i.
\end{aligned}$$

We propose a solution based on a gradient descent step followed by simple projections into the weight matrix constraints. The gradient of the smooth term of the objective function can be given in a closed form as follows

$$\begin{aligned}
&\nabla_W ||Y - \sum_{s=1}^{S} \mathcal{D}_s X_s||_F^2 \\
&= \sum_{s=1}^{S} \sum_{k=1}^{K} \alpha_{sk} \big(-\sum_{r=0}^{k-1} 2A_{k,r}^T + \mathbf{1}_{N\times N}(B_k \circ I) \big),
\end{aligned}$$

where $A_{k,r} = D^{-1/2}\mathcal{L}^{k-r-1} X_s (Y - \mathcal{D}X)^T \mathcal{L}^r D^{-1/2}$, $B_k = \sum_{r=0}^{k-1} D^{-1/2} W A_{k,r} D^{-1/2} + A_{k,r} W D^{-1}$, $\mathbf{1}_{N\times N}$ is a matrix of ones, I is an $N \times N$ identity matrix, and $\circ$ denotes the pairwise (Hadamard) product. This result is obtained by using properties of the trace operator and applying the chain rule for the gradient. However, we omit the detailed derivation of the gradient due to space constraints. To approximate the gradient of the non-differentiable $\beta_W ||W||_1$ term, we use $\beta_W \text{sign}(W)$. This operation ends up shrinking all positive elements by β_W in every step, while not changing the zero ones. To avoid complex projections, we use a symmetric gradient with a zero-diagonal. By doing so, we are optimizing over only $N(N-1)/2$ variables, instead of N^2. Note that the projections are quite simple, i.e.,

$$\underset{\tilde{W_{ij}} \geq 0}{\text{argmin}} ||\tilde{W_{ij}} - W_{ij}||_F^2 = \begin{cases} 0, & \text{if } W_{ij} \leq 0 \\ W_{ij}, & \text{otherwise,} \end{cases}$$

and even promote sparsity. The way that we approximate the gradient of the $\beta_W ||W||_1$ term, together with the projection to the positive space, bears strong similarity to using the proximal operator for the L_1 norm, with the only difference being in the fact that we project to the positive space after performing the gradient descent. It is important to note that since we are alternating between the sparse coding and the graph update step, there is no theoretical guarantee for convergence to a local optimum. However, the method has shown promising results in all conducted tests, as we will see in the next section.

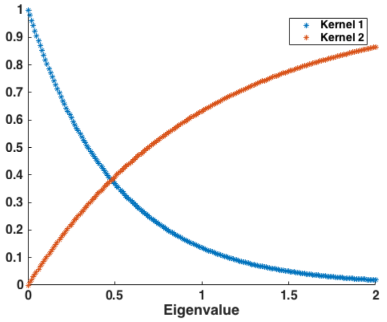


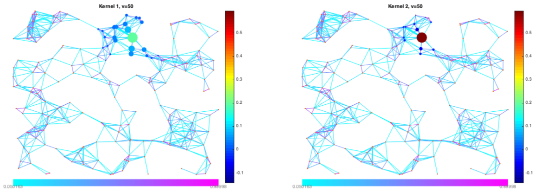
Fig. 1: Generating kernels for a polynomial dictionary

4. SIMULATION RESULTS

We have tested the performance of our algorithms on synthetic data that follow our signal model. We carry our experiments on sparse Erdős-Rényi model (ER) [18] graphs with $N = 20, 50$ and 100 vertices, and on an radial basis function (RBF) random graph. In the case of the RBF graph, we generate the coordinates of the vertices uniformly at random in the unit square, and we set the edge weights based on a thresholded Gaussian kernel function:

$$W(i,j) = \begin{cases} e^{-\frac{[\text{dist}(i,j)]^2}{2\sigma^2}}, & \text{if } dist(i,j) \leq \kappa \\ 0, & \text{otherwise} \end{cases}$$

All graph topologies are designed to have approximately $3N$ edges, and we generate 100 random instances of the graph. For numerical reasons, every graph is stripped of its isolated vertices, but full graph connectedness is not necessarily insured. For each instance of the graph, we then construct a parametric dictionary as a concatenation of $S = 2$ subdictionaries designed as polynomials of degree $K = 15$. The generating kernels for each subdictionary are defined by the Taylor approximation of two heat kernels, one of



(a) Kernel 1

(b) Kernel 2

Fig. 2: An example of atoms generated from two different polynomial kernels and centered at the same vertex.

which is shifted in the spectral domain to cover high graph frequencies, allowing for a more general class of signals. More precisely, we define

$$\hat{g}_1(\lambda) \approx e^{-2\lambda}, \quad \hat{g}_2(\lambda) \approx 1 - e^{-\lambda},$$

where we use $\approx$ because of the fact that each exponential function is approximated by a 15-order polynomial. The obtained kernels are illustrated in Fig. 1. An example of corresponding atoms for a graph with a thresholded Gaussian kernel can be seen on Fig. 2.

Then, we generate a set of 200 training graph signals using randomly generated sparse coding coefficients from a normal distribution. These sparse codes represent each signal as a linear combination of $T_0 = 4$ atoms from the dictionary. We use these training signals and the known polynomial coefficients to learn a graph with our proposed graph learning algorithm. The weight matrix is initialized as a random symmetric matrix with values between 0 and 1, and a zero diagonal. The parameter β_W and the gradient descent step size are determined with grid search.

We threshold the small entries of the learned matrix in order to recover a strongly sparse graph structure. Determining a good threshold value proves to be crucial in obtaining a relevant graph. As a rule of thumb in these synthetic settings, we threshold our results in such a way that the number of learned edges approximately equals to the number of the edges of the groundtruth graph. In more general settings, where the groundtruth graph is not known, we discard the entries of the matrix that are smaller than a predefined threshold of 10^{-4}.

Table 1: Mean accuracies for different graph sizes

mean value/graph size	**20**	**50**	**100**
edges precision	0.9985	0.9948	0.9818
edges recall	0.9983	0.9946	0.9810
sparse codes precision	0.7708	0.9317	0.9244
sparse codes recall	0.8001	0.9464	0.9316

Table 1 shows the mean precision and recall in recovering the graph edges and also the sparse codes, per graph size, averaged over 100 different graph instances. As expected, the edge recovery accuracy drops slightly with the size of the graph, due to the fact that the number of training signals has not been changed proportionally to the graph size. On the other hand, the recovery performance of sparse codes is much higher in graphs of sizes 50 and 100 than in smaller graphs. We note that this is probably due to the fact that all tests are performed on training signals constructed from equal number of atoms. For that reason, the overlapping of atoms is much higher in small graphs, making the recovery of sparse codes a more challenging task.

Moreover, we have tested the influence of both the sparsity level and the size of the training set on an ER graph with 100 vertices. Fig. 3 displays the F-measure score for the recovered edges and the sparse codes. As expected, we can see that a larger number of training signals leads to better recovery. In addition, signals constructed from a smaller number of atoms are more efficient in graph recovery as the atoms are less likely to overlap. Finally, it is worth noting that we have also run experiments in non-ideal settings where the generating kernels of the training signals are not identical to the kernels used in the optimization algorithm. We have observed that our algorithm is pretty robust to such noise which supports its potential towards practical settings.

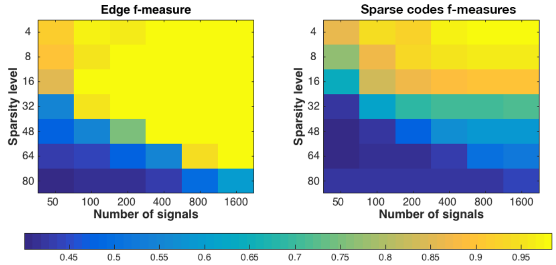


Fig. 3: Recovery accuracy depending on sparsity and the number of signals.

Next, we compare our model with two state-of-the-art graph learning algorithms, i.e., Dong et al. [8] and Segarra et al. [12], which have however been designed for smooth graph signal models. In order to have reasonably fair comparisons, we generate piecewise smooth signals, with both generating kernels favouring low frequencies ($\hat{g}_1(\lambda) \approx e^{-2\lambda}, \hat{g}_2(\lambda) \approx e^{-\lambda}$). We compare these methods on 50 ER graphs of size 20, with a probability of an edge of $p = 0.3$, and 50 RBF graphs of size 20, and train them on 500 signals. All hyper-parameters are optimized with a grid search. The average F-measure scores are given in Table 2 and we can see that our method performs better than the other two algorithms. We notice that, when the signals do not necessarily consist of only low-pass kernels, the improvement over these methods is higher as expected.

Table 2: Edges F-measures for different methods

Type/Method	Our	Dong [8]	Segarra [12]
ER	1	0.9112	0.9738
RBF	1	0.9379	0.9170

5. CONCLUSIONS

In this paper, we have presented a framework for learning graph topologies from signal observations under the assumption that the signals are sparse on a graph-based dictionary. Experimental results confirm the usefulness of the proposed algorithm in recovering a meaningful graph topology and in leading to better data understanding and inference. Even though these results seem promising, more efforts are needed to improve the scalability of learning with the graph size, possibly with help of graph reduction and reconstruction methods.

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 Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[2] A. Ng, M. Jordan, and Y. Weiss, "On Spectral Clustering: Analysis and an algorithm," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 849–856.

[3] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, 2004, pp. 132–137.

[4] A. Argyriou, M. Herbster, and M. Pontil, "Combining Graph Laplacians for Semi-Supervised Learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2005, pp. 67–74.

[5] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data," *Journal of Machine Learning Research*, vol. 9, pp. 485–516, Jun 2008.

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

[7] S. Celik, B. Logsdon, and S.I. Lee, "Efficient dimensionality reduction for high-dimensional network estimation," in *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014, pp. 1953–1961.

[8] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, 2016.

[9] V. Kalofolias, "How to learn a graph from smooth signals," in *International Conference on Artificial Intelligence and Statistics*, 2016, p. 920–929.

[10] E. Pavez and A. Ortega, "Generalized Laplacian precision matrix estimation for graph signal processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

[11] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of weighted graph topologies from observations of diffused signals," *arXiv:1605.02569*, 2016.

[12] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *arXiv:1608.03008v1*, 2016.

[13] Jonathan Mei and José MF Moura, "Signal processing on graphs: Estimating the structure of a graph," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5495–5499.

[14] Jonathan Mei et al., "Signal processing on graphs: Performance of graph structure estimation," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 6165–6169.

[15] D. Thanou, D. I Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3849–3862, Aug 2014.

[16] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, Mar 2011.

[17] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[18] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.