GRAPH LEARNING FROM NOISY AND INCOMPLETE SIGNALS ON GRAPHS

Abdullah Karaaslanli and Selin Aviyente

Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI.

2021 IEEE Statistical Signal Processing Workshop (SSP) | 978-1-7281-5767-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/SSP49050.2021.9513838

ABSTRACT

Learning the graph structure underlying observed graph signals is important in many graph signal processing (GSP) applications. This problem has been extensively addressed as graph Laplacian learning with the constraint that the graph signals have smooth variations on the resulting topology. The current approaches focus primarily on the case that the signals are observed across all nodes and possibly corrupted by additive Gaussian noise. In this paper, we propose a general framework for graph learning where the graph signal is partially observed and corrupted by sparse outliers in addition to Gaussian noise. We present a general optimization framework that addresses this problem and show how this formulation encapsulates a variety of problems in GSP including Laplacian learning and graph regularized low-rank matrix completion. The proposed optimization is solved with ADMM and the resulting algorithms are evaluated on both simulated graphs with different topology and real world graph-based data clustering.

Index Terms— Graph Signal Processing, Graph Learning, Matrix Completion, Graph Signal Recovery

1. INTRODUCTION

In many modern data science applications, the observed high-dimensional data lives in a non-Euclidean space. Some examples include point cloud data and graph signals. In graph signal processing, it is assumed that the observed data live on the vertices of a graph. Numerous examples can be found in real world applications, such as temperatures within a geographical area, transportation capacities at hubs in a transportation network, and neuronal signals recorded across a brain network [1, 2]. While most of the research efforts in GSP have focused on extracting information from such graph signals using the *a priori* known graph structure, in most cases the graph structure may be unknown, corrupted or noisy [3]. Recent research has addressed this issue by proposing graph learning techniques from observed graph signals. The main techniques are statistical methods such as Gaussian graphical models and graphical Lasso; learning graphs from data assumed to be smooth over the graph; diffusion based models, where the graph signals are assumed to be stationary over the graph and are produced by a diffusion process defined by the graph shift operators [4]. In this paper, the focus is on learning graphs from observations of smooth signals.

There are various reasons to focus on learning graphs with the assumption that signals must vary smoothly with respect to the graph structure. First, smooth signals have low-frequency and sparse representation in the graph spectral domain. Thus, the graph learning problem under smoothness assumption is equal to finding efficient information processing transforms for graph signals. Second, smoothness is a fundamental principle for several graph regularized learning tasks, such as denoising, semi-supervised learning, and spectral clustering. Finally, many real-world graph signals are smooth as similarity between nodes' properties are usually employed to construct graphs [4].

The pioneer work on graph learning with smoothness assumption is proposed in [5], where factor analysis model is used to describe graph signals. In particular, the signals are assumed to be generated from a set of latent variables, which represent low-frequency graph spectral representation of signals. The latent variables are then transformed to the observed graph signals by graph Fourier basis, which involves information about graph topology. This modeling results in an optimization problem whose objective function consisting of terms that minimize the error between the observed signal and the underlying signal while ensuring that the graph signals are smooth on the underlying network. Kalofolias et al. [6] extended this framework by establishing the link between smoothness and sparsity and regularizing the problem such that that each vertex has at least one incident edge in the learnt graph. Different variations of these frameworks were considered in [7, 8, 9, 10, 11, 12]. For example, in [7], the Frobenius norm of the error between the observed and constructed data matrices in [5] is replaced by ℓ_1-norm, such that the learned Laplacian is robust against sparse outliers in the graph signals. [11] considers the problem of simultaneous low-rank data recovery and graph learning where the focus is on detecting sparse outliers using the underlying graph structure as a regularizer. More recently, [8] considered missing data in the framework of graph learning.

In this paper, we introduce a comprehensive framework for graph learning from corrupted signals with possibly missing entries. The proposed framework extends and generalizes existing graph learning methods based on signal smoothness assumptions in some key ways. First, the proposed approach can simultaneously recover the low-rank data matrix and the underlying graph from data. Second, compared to previous works, the proposed work handles three types of corruptions simultaneously, i.e. Gaussian noise, sparse outliers and missing data. Finally, the optimization framework is formulated in a general way such that the objective functions of both [5] and [6] are considered. The resulting optimization problem is non-convex and solved by alternating minimization with two subproblems. The first subproblem is reformulated in terms of vectorized Laplacian matrices providing a fast and scalable algorithm, while the second subproblem can be solved with fast algorithms developed in nuclear norm minimization literature [13].

2. BACKGROUND

2.1. Notations

An undirected graph is represented by a tuple $G = (V, E)$, where V is the node set with $|V| = n$ and $E \in V \times V$ is the edge set. Algebraically, a graph is represented by a symmetric adjacency matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$. The degree matrix of G is a diagonal matrix $\mathbf{D}$ with $D_{ii} = \sum_{j=1}^{n} W_{ij}$. A graph signal is a column vector $\mathbf{x} \in \mathbb{R}^n$ with x_i being the graph signal value on node i.

This work was in part supported by NSF CCF-2006800.

All-one and all-zero vectors and matrices are indicated by $\mathbf{1}$ and $\mathbf{0}$, respectively. $\mathrm{diag}(\cdot)$ operator is defined on vectors and matrices. If the input is a vector, it returns a diagonal matrix whose diagonal is equal to the input vector. Otherwise, it returns the diagonal of the input matrix as a vector. We also define the operator $\mathrm{upper}(\cdot)$, which takes an $n \times n$ symmetric matrix and returns a $n(n-1)/2$-dimensional vector that corresponds to the upper triangular part of the input matrix. Finally, we define the matrix $\mathbf{P} \in \mathbb{R}^{n \times n(n-1)/2}$ such that $\mathbf{P}\mathrm{upper}(\mathbf{W}) = \mathbf{W}\mathbf{1} - \mathrm{diag}(\mathbf{W})$.

2.2. Graph Learning from Smooth Signals

A graph signal $\mathbf{x}$ is said to be smooth if strongly connected vertices have similar values, while weakly connected vertices have dissimilar values. Mathematically, the smoothness of $\mathbf{x}$ can be measured using various metrics. One common approach is to calculate the total variation of $\mathbf{x}$ with respect to graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$ as:

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(x_i - x_j)^2. \tag{1}$$

Graph learning aims to infer the connectivity of an unknown graph G from m graph signals that are assumed to be smooth on G by minimizing the smoothness measure in (1) with respect to $\mathbf{L}$. Given a matrix of graphs signals, $\mathbf{X} \in \mathbb{R}^{n \times m}$, $\mathbf{L}$ can be learned by solving the following optimization problem [5]:

$$\min_{\mathbf{L} \in \mathbb{L}} \ \mathrm{tr}(\mathbf{X}^\top \mathbf{L}\mathbf{X}) + \frac{\alpha}{2}||\mathbf{L}||_F^2 \quad \text{s.t. } \mathrm{tr}(\mathbf{L}) = n, \tag{2}$$

where $\mathbb{L} = \{\mathbf{L} | \mathbf{L} = \mathbf{L}^\top, L_{ij} = L_{ji} \leq 0 \forall i \neq j, \mathbf{L}\mathbf{1} = \mathbf{0}\}$ is the set of Laplacian matrices, $||\mathbf{L}||_F$ is the Frobenius norm of $\mathbf{L}$ that controls sparsity of the learned graph and the constraint is employed to prevent the trivial solution $\mathbf{L} = \mathbf{0}$. In [6], the objective function in (2) is augmented with the log-barrier function of $\mathrm{diag}(\mathbf{L})$ to ensure that each node has at least one connection.

2.3. Graph Signal Recovery

In real world applications, measurements of graphs signals are usually contaminated by sparse outliers, Gaussian noise or missing values. One important task in GSP is to recover the graph signals from the observed corrupted signals with the assumption that the underlying signals are smooth on the graph [14]. Given a graph G and the corrupted graph signal matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, graph signal recovery can be formulated as the following optimization problem:

$$\min_{\mathbf{Y},\mathbf{S}} \mathrm{tr}(\mathbf{Y}^\top \mathbf{L}\mathbf{Y}) + \lambda_1 \|\mathbf{Y}\|_* + \lambda_2 \|\mathbf{S}\|_1 + \frac{\lambda_3}{2}\|\mathbf{M} \circ (\mathbf{Y} + \mathbf{S} - \mathbf{X})\|_F^2, \tag{3}$$

where $\mathbf{Y}$ is the signal to be recovered, $\mathbf{S}$ is the sparse outlier matrix, $\mathbf{M}$ is the binary mask matrix with $M_{ij} = 0$ if X_{ij} is missing and 1 otherwise and $\circ$ is Hadamard product. The first term in (3) measures the smoothness of graph signals $\mathbf{Y}$ on G, the second term models $\mathbf{Y}$ to be low-rank based on the assumption that $\mathbf{Y}$ includes redundant information as each graph signal is generated from the same graph topology [14] and the last two terms model sparse outliers and Gaussian noise in the observations, respectively.

3. GRAPH LEARNING FROM CORRUPTED SIGNALS

As mentioned in the introduction, previous approaches on graph learning from corrupted data are limited in some key ways. In this paper, we extend prior work in graph learning by proposing a comprehensive optimization framework that considers different types of corruptions, i.e. sparse outliers, Gaussian noise and missing values, simultaneously. Moreover, the proposed framework addresses the problem of graph signal recovery jointly with graph learning. Thus, aforementioned works are special cases of the proposed framework.

Before formulating our approach, we first present some observations on the problem considered in (2) and its augmented version considered in [6]. Since we are learning a symmetric matrix $\mathbf{L}$, the problem can be written in vector form, where we learn upper triangular and diagonal part of $\mathbf{L}$. Let $\boldsymbol{\ell} = \mathrm{upper}(\mathbf{L})$, $\mathbf{d} = \mathrm{diag}(\mathbf{L})$, $\mathbf{y} = \mathrm{upper}(\mathbf{Y}\mathbf{Y}^\top)$ and $\mathbf{d}_y = \mathrm{diag}(\mathbf{Y}\mathbf{Y}^\top)$. Then we have the following general formulation for (2):

$$\min_{\boldsymbol{\ell} \leq 0, \mathbf{d}} 2\mathbf{y}^\top \boldsymbol{\ell} + \mathbf{d}_y^\top \mathbf{d} + \beta_1 f(\mathbf{d}) + \beta_2 g(\boldsymbol{\ell}) \ \text{ s.t. } \ \mathbf{P}\boldsymbol{\ell} = -\mathbf{d}, \tag{4}$$

where the first two terms correspond to $\mathrm{tr}(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$, $f(\mathbf{d})$ is a function that controls the degree distribution and $g(\boldsymbol{\ell})$ is a function that controls the sparsity of learned graph, respectively. By using different $f(\mathbf{d})$ and $g(\boldsymbol{\ell})$, one can obtain different graph learning approaches that are based on smoothness assumptions. For example, in (2), $f(\mathbf{d})$ is set to $\|\mathbf{d}\|_2^2/2$ with the constraint $\mathbf{1}^\top \mathbf{d} = n$. In [6], $f(\mathbf{d}) = -\mathbf{1}^\top \log(\mathbf{d})$ is log-barrier function to make sure each node has at least a connection. In prior work, $g(\boldsymbol{\ell}) = \|\boldsymbol{\ell}\|_2^2/2$ is employed. In the remaining of this paper, we provide solutions for both choices of $f(\mathbf{d})$ and set $g(\boldsymbol{\ell}) = \|\boldsymbol{\ell}\|_2^2/2$.

To learn the graph from corrupted graph signals, we combine (3) and (4) to obtain the following optimization problem:

$$\begin{aligned} \min_{\mathbf{L} \in \mathbb{L}, \mathbf{Y}, \mathbf{S}} \ & \mathrm{tr}(\mathbf{Y}^\top \mathbf{L}\mathbf{Y}) + \beta_1 f(\mathrm{diag}(\mathbf{L})) + \beta_2 g(\mathrm{upper}(\mathbf{L})) \\ & + \lambda_1 \|\mathbf{Y}\|_* + \lambda_2 \|\mathbf{S}\|_1 + \lambda_3/2 \|\mathbf{M} \circ (\mathbf{Y} + \mathbf{S} - \mathbf{X})\|_F^2. \\ \text{s.t} \quad & \mathbf{P}\mathrm{upper}(\mathbf{L}) = -\mathrm{diag}(\mathbf{L}). \end{aligned} \tag{5}$$

This problem is not jointly convex in $\mathbf{L}$, $\mathbf{Y}$ and $\mathbf{S}$. Therefore, we employ an alternating minimization approach similar to [5], [8]. First, we fix $\mathbf{Y}$ and $\mathbf{S}$ and solve the problem for $\mathbf{L}$, then fixing $\mathbf{L}$ we solve with respect to $\mathbf{Y}$ and $\mathbf{S}$. In the following, both subproblems are solved with alternating direction method of multipliers (ADMM).

L-subproblem: In this section, we present the solution for $f(\mathbf{d}) = \|\mathbf{d}\|_2^2/2$ with the constraint $\mathbf{1}^\top \mathbf{d} = n$ using ADMM. Due to space constraints, the solution for $f(\mathbf{d}) = -\mathbf{1}^\top \log(\mathbf{d})$ is not given. However, its performance is reported in Section 4. The augmented Lagrangian of the problem is:

$$\begin{aligned} \mathcal{L}_{\rho_1}(\boldsymbol{\ell}, \mathbf{d}, \boldsymbol{\eta}, \gamma) = {} & 2\mathbf{y}^\top \boldsymbol{\ell} + \frac{\beta_2}{2}\boldsymbol{\ell}^\top \boldsymbol{\ell} + \frac{\beta_1}{2}\mathbf{d}^\top \mathbf{d} + \mathbf{d}_y^\top \mathbf{d} + \boldsymbol{\eta}^\top(\mathbf{d} + \mathbf{P}\boldsymbol{\ell}) \\ & + \frac{\rho_1}{2}||\mathbf{d} + \mathbf{P}\boldsymbol{\ell}||_2^2 + \gamma(\mathbf{1}^\top \mathbf{d} - n) + \frac{\rho_1}{2}(\mathbf{1}^\top \mathbf{d} - n)^2. \end{aligned} \tag{6}$$

The steps of ADMM are as follows:

$$\boldsymbol{\ell}^{k+1} = \underset{\boldsymbol{\ell} \leq 0}{\mathrm{argmin}} \ \mathcal{L}_{\rho_1}(\boldsymbol{\ell}, \mathbf{d}^k, \boldsymbol{\eta}^k) \tag{7}$$

$$= \Pi_{\mathbb{R}_-^{n(n-1)/2}}\left[-[\beta_2 \mathbf{I} + \rho_1 \mathbf{P}^\top \mathbf{P}]^{-1}(2\mathbf{y} + \mathbf{P}^\top(\boldsymbol{\eta}^k + \rho_1 \mathbf{d}^k))\right], \tag{8}$$

$$\mathbf{d}^{k+1} = \underset{\mathbf{d}}{\mathrm{argmin}} \ \mathcal{L}_{\rho_1}(\boldsymbol{\ell}^{k+1}, \mathbf{d}, \boldsymbol{\eta}^k) \tag{9}$$

$$= [(\beta_1 + \rho_1)\mathbf{I} + \rho_1 \mathbf{1}\mathbf{1}^\top]^{-1}[(\rho_1 n - \gamma)\mathbf{1} - \mathbf{d}_y - \boldsymbol{\eta}^k - \rho_1 \mathbf{P}\boldsymbol{\ell}^{k+1}], \tag{10}$$

$$\boldsymbol{\eta}^{k+1} = \boldsymbol{\eta}^k + \rho_1(\mathbf{d}^{k+1} + \mathbf{P}\boldsymbol{\ell}^{k+1}), \tag{11}$$

$$\gamma^{k+1} = \gamma^k + \rho_1(\mathbf{1}^\top \mathbf{d}^{k+1} - n), \tag{12}$$

where $\Pi_{\mathbb{R}_-^{n(n-1)/2}}$ is the Euclidean projection on negative orthant. The problems in (7) and (9) are quadratic, which yield closed form solutions in (8) and (10), respectively. Note that the inverses in $\boldsymbol{\ell}$ and $\mathbf{d}$ steps have closed form solutions.

978-1-7281-5767-2/21/$31.00 ©2021 IEEE

$(\mathbf{Y}, \mathbf{S})$-subproblem: The solution of $(\mathbf{Y}, \mathbf{S})$-subproblem is computed with ADMM. By introducing an auxiliary variable $\mathbf{Z}$, the optimization problem can be written as:

$$\min_{\mathbf{Y},\mathbf{S},\mathbf{Z}} \operatorname{tr}(\mathbf{Y}^\top \mathbf{L}\mathbf{Y}) + \lambda_1 \|\mathbf{Z}\|_* + \lambda_2 \|\mathbf{S}\|_1 + \frac{\lambda_3}{2} \|\mathbf{M} \circ (\mathbf{Y} + \mathbf{S} - \mathbf{X})\|_F^2 \quad \text{s.t.} \quad \mathbf{Y} - \mathbf{Z} = \mathbf{0}. \tag{13}$$

The corresponding augmented Lagrangian is:

$$\mathcal{L}_{\rho_2}(\mathbf{Y}, \mathbf{S}, \mathbf{Z}, \mathbf{\Lambda}) = \operatorname{tr}(\mathbf{Y}^\top \mathbf{L}\mathbf{Y}) + \lambda_1 \|\mathbf{Z}\|_* + \lambda_2 \|\mathbf{S}\|_1 + \frac{\lambda_3}{2} \|\mathbf{M} \circ (\mathbf{Y} + \mathbf{S} - \mathbf{X})\|_F^2 + \langle \mathbf{\Lambda}, \mathbf{Y} - \mathbf{Z} \rangle + \frac{\rho_2}{2} \|\mathbf{Y} - \mathbf{Z}\|_F^2. \tag{14}$$

The different variables are updated as:

$$\mathbf{Y}_{k+1} = \underset{\mathbf{Y}}{\operatorname{argmin}}\, \mathcal{L}_{\rho_2}(\mathbf{Y}, \mathbf{S}^k, \mathbf{Z}^k, \mathbf{\Lambda}^k), \tag{15}$$

$$\mathbf{y}_i^{k+1} = [2\mathbf{L} + \lambda_3 \operatorname{diag}(\mathbf{m}_i) + \rho_2 \mathbf{I}]^{-1} (\rho_2 \mathbf{z}_i^k - \lambda_3 (\mathbf{s}_i^{k+1} - \mathbf{x}_i) - \mathbf{\Lambda}_i^k),$$

$$\mathbf{S}^{k+1} = \underset{\mathbf{S}}{\operatorname{argmin}}\, \mathcal{L}_{\rho_2}(\mathbf{Y}^{k+1}, \mathbf{S}, \mathbf{Z}^k, \mathbf{\Lambda}^k), \tag{16}$$

$$= \mathcal{S}_{\frac{\lambda_2}{\lambda_3}} (\mathbf{M} \circ (\mathbf{X} - \mathbf{Y}^{k+1})),$$

$$\mathbf{Z}^{k+1} = \underset{\mathbf{Z}}{\operatorname{argmin}}\, \mathcal{L}_{\rho_2}(\mathbf{Y}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Z}, \mathbf{\Lambda}^k) \tag{17}$$

$$= \mathcal{D}_{\frac{\lambda_1}{\rho_2} \|\cdot\|_*} (\mathbf{Y}^{k+1} + \frac{1}{\rho_2} \mathbf{\Lambda}^k),$$

$$\mathbf{\Lambda}^{k+1} = \mathbf{\Lambda}^k + \rho_2 (\mathbf{Y}^{k+1} - \mathbf{Z}^{k+1}), \tag{18}$$

where $\mathbf{y}_i$, $\mathbf{z}_i$, $\mathbf{s}_i$, $\mathbf{x}_i$, $\mathbf{m}_i$ and $\mathbf{\Lambda}_i$ are the ith columns of the corresponding matrices, $\mathcal{S}_\tau(\cdot)$ is the elementwise shrinkage operator and $\mathcal{D}_\tau(\cdot)$ is the singular value thresholding operator. The problem in (15) is separable across columns of $\mathbf{Y}$, therefore it is solved for each column separately. (16) and (17) are proximal operators of ℓ_1 and nuclear norm, respectively and have closed form solutions [15].

Convergence: $\mathbf{L}$-subproblem is convex and the proposed solution is a two-block ADMM, whose convergence can be shown using approaches in [16]. Similarly, $(\mathbf{Y}, \mathbf{S})$-subproblem is convex and the provided solution is a two-block ADMM with convergence guarantee. In our experiments, we observe the alternating minimization to converge. However, since the problem in (5) is non-convex, it is not guaranteed to converge to the global minimizer.

Parameter Selection: There are five hyperparameters in (5) that need to be tuned. In our experiments, we utilized the following observations to select these parameters. Following the literature on robust PCA [17], we set $\lambda_2 = \lambda_1 / \sqrt{max(n, m)}$. Next, $\lambda_3 = c\lambda_1$ where c is a constant that can be selected based on prior knowledge on how noisy the observed data is. Since β_2 controls the density of the learned graph, it can be set to a value that gives the desired edge density. For selection of β_1 and λ_1, we apply grid search and observe that there is a large range of values for which the algorithms perform well.

4. RESULTS

Proposed methods are compared to the state-of-art graph learning algorithms in [5] and [6] on simulated data. For implementation of [5], we use our algorithm since the optimization in (5) is equal to the one in [5] when $\beta_1 = \beta_2$, $\lambda_1 = \lambda_2 = 0$ and $f(\mathbf{d}) = \frac{1}{2}\mathbf{d}^\top \mathbf{d}$. For [6], we used GSPBOX [1]. We also report the performance of the methods on a real-world clustering problem. In the following, the proposed methods are referred to as $\mathrm{RoGL\text{–}MC}_1$ and $\mathrm{RoGL\text{–}MC}_2$ for $f(\mathbf{d}) = \mathbf{d}^\top \mathbf{d}/2$ and $f(\mathbf{d}) = -\mathbf{1}^\top \log(\mathbf{d})$, respectively.

Simulated Data: Synthetic graphs are generated from three different random graph models: Gaussian RBF (GRBF), Erdős–Rényi (ER) [18] and Barabási–Albert (BA) [19]. GRBF is constructed from 100 points uniformly sampled from $[0, 1] \times [0, 1]$. Each pair of sampled points (i, j) is connected with an edge whose weight is $w_{ij} = \exp(-d(i, j)^2 / 2\sigma^2)$ where $d(i, j)$ is the Euclidean distance between points i and j. We set $\sigma = 0.5$ and edges with weights smaller than 0.75 are removed. ER graphs are generated with edge probability 0.05. Finally, BA graph is constructed by growing an initial graph with 2 nodes and 1 edge. At each iteration, a new node is added to the graph by connecting it to 2 other nodes. From each random graph model, 1000 smooth graph signals are generated as described in [5]. In particular, $\mathbf{y}_i \sim \mathcal{N}(0, \mathbf{L}^\dagger)$ where $\mathbf{L}^\dagger$ is the pseudo-inverse of graph Laplacian of the generated random graph. To generate corrupted signals $\mathbf{x}_i$, we first add Gaussian noise with standard deviation $\sigma_\epsilon = 0.3$ to each signal $\mathbf{y}_i$. A mask matrix M is generated such that $M_{ij} = 0$ with probability p_m and 1, otherwise. Finally, $p_s\%$ of the observed entries are contaminated by additive binary sparse noise whose value is selected from $\{+\max(\operatorname{abs}(\mathbf{Y}))/2, -\max(\operatorname{abs}(\mathbf{Y}))/2\}$, where $\max(\cdot)$ and $\operatorname{abs}(\cdot)$ are elementwise operations.

Performance of the different graph learning algorithms is evaluated by the F-measure which is calculated by comparing the learned graphs to the ground truth graph structure. Each simulation is repeated 10 times and mean values of F-measure are reported in Table 1 for various p_s and p_m values. The performance of all methods

[1] https://github.com/epfl-lts2/gspbox

Table 1. Graph learning performances of various algorithms for different network structures and corruption levels.

			Gaussian RBF			Erdős–Rényi			Barabási–Albert		
	p_m \ p_s	Methods	5%	10%	15%	5%	10%	15%	5%	10%	15%
F-measure	0	[5]	0.784	0.712	0.653	0.734	0.587	0.490	0.796	0.727	0.694
		[6]	0.778	0.714	0.666	0.768	0.608	0.504	0.780	0.694	0.636
		$\mathrm{RoGL\text{–}MC}_1$	**0.823**	**0.775**	**0.743**	**0.876**	**0.814**	**0.770**	0.824	0.792	0.770
		$\mathrm{RoGL\text{–}MC}_2$	0.812	0.765	0.732	0.840	0.776	0.748	**0.828**	**0.806**	**0.780**
	0.25	[5]	0.725	0.657	0.580	0.596	0.481	0.353	0.706	0.669	0.597
		[6]	0.712	0.658	0.602	0.628	0.487	0.352	0.715	0.641	0.547
		$\mathrm{RoGL\text{–}MC}_1$	**0.775**	**0.726**	**0.669**	**0.794**	**0.731**	**0.629**	0.781	0.755	0.707
		$\mathrm{RoGL\text{–}MC}_2$	0.760	0.716	0.657	0.770	0.703	0.601	**0.788**	**0.766**	**0.709**
	0.5	[5]	0.638	0.573	0.554	0.492	0.335	0.259	0.587	0.511	0.436
		[6]	0.639	0.580	0.551	0.514	0.339	0.259	0.614	0.510	0.418
		$\mathrm{RoGL\text{–}MC}_1$	**0.703**	**0.623**	**0.604**	**0.657**	**0.536**	0.430	0.696	0.630	0.549
		$\mathrm{RoGL\text{–}MC}_2$	0.694	0.616	0.591	0.627	0.532	**0.436**	**0.721**	**0.641**	**0.561**

Table 2. Data recovery performances of various algorithms for different network structures and corruption levels.

	p_m \ p_s	Methods	Gaussian RBF 5%	Gaussian RBF 10%	Gaussian RBF 15%	Erdős–Rényi 5%	Erdős–Rényi 10%	Erdős–Rényi 15%	Barabási–Albert 5%	Barabási–Albert 10%	Barabási–Albert 15%
MSE	0.0	[5]	0.543	0.954	1.415	0.981	1.945	3.153	0.619	1.241	1.707
		RPCA-MC	**0.278**	0.431	0.635	**0.336**	0.555	0.870	**0.356**	0.583	0.841
		RoGL–MC$_1$	0.280	**0.420**	**0.600**	**0.336**	**0.532**	0.802	0.360	**0.566**	**0.796**
		RoGL–MC$_2$	0.281	**0.420**	**0.600**	0.338	0.533	**0.801**	0.361	**0.566**	**0.796**
	0.25	[5]	0.699	1.040	1.552	1.268	2.081	3.015	0.951	1.347	1.796
		RPCA-MC	0.504	0.650	0.973	0.650	0.866	1.148	0.704	0.891	1.143
		RoGL–MC$_1$	**0.503**	**0.638**	**0.951**	**0.645**	**0.841**	**1.083**	**0.701**	**0.873**	**1.100**
		RoGL–MC$_2$	**0.503**	0.639	**0.951**	0.646	0.842	1.084	**0.701**	**0.873**	**1.100**
	0.5	[5]	0.866	1.142	1.277	1.272	1.828	2.540	1.271	1.523	1.822
		RPCA-MC	0.744	0.870	0.973	0.966	1.146	1.410	1.096	1.249	1.446
		RoGL–MC$_1$	**0.741**	**0.859**	**0.951**	**0.958**	**1.124**	**1.359**	**1.086**	**1.230**	**1.411**
		RoGL–MC$_2$	**0.741**	**0.859**	**0.951**	0.959	1.125	1.361	**1.086**	**1.230**	**1.411**

deteriorates as the amount of missing data and the number of sparse outliers increase. However, the proposed framework performs better than the methods in [5] and [6] especially for larger values of p_s and p_m. Finally, it can be observed that RoGL–MC$_2$ results in better performance than RoGL–MC$_1$ in BA networks, while the latter performs better than the former in ER and GRBF networks. This is due to the fact that BA is a graph with power-law degree distribution, while the other two models do not. This means that log-barrier might be a better degree regularizer for power-law degree distribution, which is commonly observed in real world graphs.

Finally, the performance of the proposed methods in signal recovery is compared to those of [5] and robust PCA with matrix completion (RPCA-MC), which corresponds to (3) without smoothness term. The performance is quantified by mean-squared error (MSE) and reported in Table 2 for various p_s and p_m values. RoGL–MC$_1$ and RoGL–MC$_2$ have similar MSE values and are better than [5], which is not robust against sparse outliers and missing values. Moreover, the proposed methods perform better than RPCA-MC for larger values of p_s and p_m, which indicates that incorporating the graph structure results in better signal recovery. The proposed methods achieve this task while learning the graph, simultaneously.

Real Data: We also apply the proposed method to a real-world clustering problem considered in [6]. In particular, we select 100 images from each of the digits 1 and 2 of MNIST dataset[2]. Images are resized to 20×20 and pixel values are normalized to the range $[0, 1]$. By vectorizing each image, we construct the data matrix $\mathbf{Y} \in \mathbb{R}^{200\times 400}$. As in the simulations, missing values are added to $\mathbf{Y}$ with probability p_m and $p_s\%$ of the observed entries are contaminated with sparse noise whose value is selected from $\{+\max(\text{abs}(\mathbf{Y})), -\max(\text{abs}(\mathbf{Y}))\}$. We learn graphs for varying average degree k from corrupted data using RoGL–MC$_1$. We do not report the results for RoGL–MC$_2$ as its performance is observed to be similar. Fiedler vector [20] of the learned graphs is used to cluster the images such that the negative and positive entries of the Fiedler vector are used to bipartition the data. Classification accuracy is calculated as the ratio of correctly classified images to the total number. We compare the results with clustering on a kNN graph constructed from original data $\mathbf{Y}$ and the graph learned by [5]. Fig. 1a and 1b illustrate the classification accuracy as a function of the network density, i.e. average degree, for $p_s = 20\%$ and two different values of p_m. For both values of p_m, graph learning approaches achieve better accuracy than kNN graph, which indicates graph learning is better at revealing the relations between image samples. Moreover, RoGL–MC$_1$ performs better than [5] as it recovers the data from the corrupted samples and learns meaningful graphs.

[2]http://yann.lecun.com/exdb/mnist/

5. CONCLUSIONS

In this work, we proposed a comprehensive optimization framework to address graph learning from corrupted signals problem. Different from previous works, we addressed two inter-related problems simultaneously: graph learning from signals that are corrupted by sparse outliers, Gaussian noise and missing data; and graph signal recovery. The proposed problem extends two widely used graph learning approaches by rewriting the $\mathbf{L}$-subproblem in a general framework and providing an efficient solution with ADMM by exploiting its vectorized form.

Future work will consider faster implementations of $(\mathbf{Y}, \mathbf{S})$-subproblem by approximating the nuclear norm minimization with scalable algorithms. Different regularization functions for $f(\mathbf{d})$ and $g(\boldsymbol{\ell})$ will also be considered to tailor the proposed methods to a wide variety of graph structures.

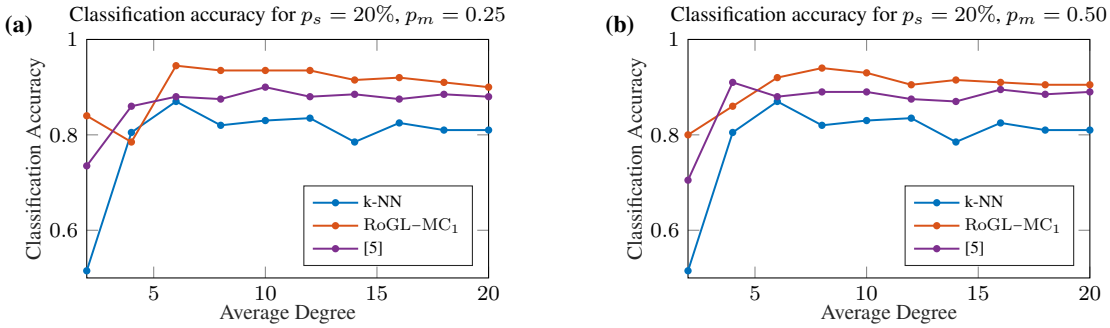

Fig. 1. Classification accuracy when Fiedler vector of learned graphs is used to cluster digits 1 and 2 of MNIST dataset.

978-1-7281-5767-2/21/$31.00 ©2021 IEEE

6. REFERENCES

[1] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[2] Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[3] Xiaowen Dong, Dorina Thanou, Michael Rabbat, and Pascal Frossard, "Learning Graphs From Data: A Signal Representation Perspective," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 44–63, 2019.

[4] Gonzalo Mateos, Santiago Segarra, Antonio Marques, and Alejandro Ribeiro, "Connecting the Dots: Identifying Network Structure via Graph Signal Processing," *IEEE Signal Processing Magazine*, p. 28, 2019.

[5] Xiaowen Dong, Dorina Thanou, Pascal Frossard, and Pierre Vandergheynst, "Learning Laplacian Matrix in Smooth Graph Signal Representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[6] Vassilis Kalofolias, "How to learn a graph from smooth signals," pp. 920–929, 2016.

[7] Junhui Hou, Lap-Pui Chau, Ying He, and Huanqiang Zeng, "Robust laplacian matrix learning for smooth graph signals," in *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 2016, pp. 1878–1882, IEEE.

[8] Peter Berger, Gabor Hannak, and Gerald Matz, "Efficient Graph Learning From Noisy and Incomplete Data," *IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS*, vol. 6, pp. 15, 2020.

[9] Sai Kiran Kadambari and Sundeep Prabhakar Chepuri, "Learning Product Graphs from Multidomain Signals," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, 2020, pp. 5665–5669, IEEE.

[10] Batiste Le Bars, Pierre Humbert, Laurent Oudre, and Argyris Kalogeratos, "Learning Laplacian Matrix from Bandlimited Graph Signals," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, United Kingdom, 2019, pp. 2937–2941, IEEE.

[11] Liu Rui, Hossein Nejati, Seyed Hamid Safavi, and Ngai-Man Cheung, "Simultaneous low-rank component and graph estimation for high-dimensional graph signals: Application to brain imaging," pp. 4134–4138, 2017.

[12] Shuyu Dong, "Graph learning for regularized low-rank matrix completion," p. 8.

[13] Tae-Hyun Oh, Yasuyuki Matsushita, Yu-Wing Tai, and In So Kweon, "Fast randomized singular value thresholding for nuclear norm minimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4484–4493.

[14] Siheng Chen, Aliaksei Sandryhaila, José MF Moura, and Jelena Kovačević, "Signal recovery on graphs: Variation minimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 17, pp. 4609–4624, 2015.

[15] Neal Parikh and Stephen Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[16] Stephen Boyd, Neal Parikh, and Eric Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Now Publishers Inc, 2011.

[17] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright, "Robust principal component analysis?," *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011.

[18] Paul Erdős and Alfréd Rényi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[19] Albert-László Barabási and Réka Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[20] Miroslav Fiedler, "Algebraic connectivity of graphs," *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973.

978-1-7281-5767-2/21/$31.00 ©2021 IEEE