2020 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 21-24, 2020, ESPOO, FINLAND

GRAPH LEARNING FROM MULTI-ATTRIBUTE SMOOTH SIGNALS

Jitendra K. Tugnait

Department of Electrical & Computer Engineering
Auburn University, Auburn, AL 36849, USA
tugnajk@auburn.edu

ABSTRACT

We consider the problem of estimating the structure of an undirected weighted graph underlying a set of smooth multi-attribute signals. Most existing methods for graph estimation are based on single-attribute models where one associates a scalar data variable with each node of the graph, and the problem is to infer the graph topology that captures the relationships between these variables. An example is image graphs for grayscale texture images for modeling dependence of a pixel on neighboring pixels. In multi-attribute graphical models, each node represents a vector, as for example, in color image graphs where one has three variables (RGB color components) per pixel node. In this paper, we extend the single attribute approach of Kalofolias (2016) to multi-attribute data. An alternating direction method of multipliers (ADMM) algorithm is presented to optimize the objective function to infer the graph topology. Numerical results based on synthetic as well as real data are presented.

Keywords: Sparse graph learning; graph estimation; graph Laplacian; undirected graph; multi-attribute data.

1. INTRODUCTION

An undirected simple weighted graph is denoted $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ where $V = \{1, 2, \cdots, p\} = [p]$ is the set of p nodes, $\mathcal{E} \subseteq [p] \times [p]$ is the set of undirected edges, and $\boldsymbol{W} \in \mathbb{R}^{p\times p}$ stores the nonnegative weights $W_{ij} \geq 0$ associated with the undirected edges. If there is an edge between nodes i and j, then edge $\{i, j\} \in \mathcal{E}$ and $W_{ij} > 0$. If there is no edge between nodes i and j, then edge $\{i, j\} \notin \mathcal{E}$ and $W_{ij} = 0$. In a simple graph there are no self-loops or multiple edges, so $\mathcal{E}$ consists of distinct pairs $\{i, j\}$, $i \neq j$ and $W_{ii} = 0$. In an undirected graph, if $\{i, j\} \in \mathcal{E}$, then $\{j, i\} \in \mathcal{E}$. In graphical models of data variables $x_1, x_1, \cdots, x_p$, ($\boldsymbol{x} = [x_1\ x_2\ \cdots\ x_p]^\top$), a weighted graph $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ (or unweighted $\mathcal{G} = (V, \mathcal{E})$) with $|V| = p$ is used to capture relationships between the p variables x_is [1–3]. If $\{i, j\} \in \mathcal{E}$, then x_i and x_j are related (similar or dependent) in some sense, with higher W_{ij} indicating stronger similarity or dependence.

Graphical models provide a powerful tool for analyzing multivariate data [1–3]. In a statistical graphical model, the conditional statistical dependency structure among p random variables $x_1, x_1, \cdots, x_p$, is represented using an undirected graph $\mathcal{G} = (V, \mathcal{E})$. The graph $\mathcal{G}$ then is a conditional independence graph (CIG) where there is no edge between nodes i and j (i.e., $\{i, j\} \notin \mathcal{E}$) iff x_i and x_j are conditionally independent given the remaining p-2 variables x_ℓ, $\ell \in [p]$, $\ell \neq i$, $\ell \neq j$. In particular, Gaussian graphical models (GGMs) are CIGs where $\boldsymbol{x}$ is multivariate Gaussian. Suppose $\boldsymbol{x}$ has positive-definite covariance matrix $\boldsymbol{\Sigma}$ with inverse covariance matrix $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. Then Ω_{ij}, the (i, j)-th element of $\boldsymbol{\Omega}$, is zero iff x_i and x_j are conditionally independent. Such models for $\boldsymbol{x}$ have been extensively studied. Given n samples of $\boldsymbol{x}$, in *high-dimensional settings* where $p \gg 1$ and/or n is of the order of p, one estimates $\boldsymbol{\Omega}$ under some sparsity constraints; see [4–8]. In these graphs each node represents a scalar random variable. In many applications, there may be more than one random variable associated with a node. This class of graphical models has been called multi-attribute graphical models in [9–11]. In [12, 13] image graphs for grayscale texture images are inferred for modeling dependence of a pixel on neighboring pixels; here one has one variable per pixel node. These approaches do not apply to color images where one has three variables (RGB color components) per pixel node. Image graphs for color images is an example of multi-attribute graphical models.

Graphical models for data variables have been inferred from consideration other than statistical, depending upon the intended application, nature of data and available prior information [1]. One class of graphical models are based on signal smoothness [1, 14, 15]. Suppose we are given n samples $\{\boldsymbol{x}(t)\}_{t=1}^n$ of the p data variables $x_1, x_1, \cdots, x_p$, with $\boldsymbol{x}(t) = [x_1(t)\ x_2(t)\ \cdots\ x_p(t)]^\top$. Define the $p \times n$ matrix

$$\boldsymbol{X} = \begin{bmatrix} x_1(1) & x_1(2) & \cdots & x_1(n) \\ x_2(1) & x_2(2) & \cdots & x_2(n) \\ \vdots & \vdots & \ddots & \vdots \\ x_p(1) & x_p(2) & \cdots & x_p(n) \end{bmatrix}. \tag{1}$$

A measure of smoothness of signal $\boldsymbol{x}(t)$ under which the signal takes "similar" values at "neighboring" vertices of a given weighted undirected graph, is the function [1, 14]

$$\frac{1}{2}\sum_{i,j=1}^{p} W_{ij}\|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_2^2 = \text{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X}) \tag{2}$$

where $\boldsymbol{X}_{i\cdot}$ denotes the ith row of $\boldsymbol{X}$, $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the (combinatorial) graph Laplacian (matrix), and $\boldsymbol{D}$ is the diagonal weighted degree matrix with $D_{ii} = \sum_{j=1}^p W_{ij}$. In the words of [14], if two vectors $\boldsymbol{X}_{i\cdot}$ and $\boldsymbol{X}_{j\cdot}$ from a smooth set of signals are associated with two well-connected nodes i and j (i.e., W_{ij} is large), they are expected to have a small distance $\|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_2$ so that $\text{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X})$ is small. In particular, if $\boldsymbol{X}_{i\cdot}$ does not vary with i, $\text{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X}) = 0$. Based on the smoothness constraint, graph learning from data $\boldsymbol{X}$ becomes equivalent to estimation of the graph Laplacian matrix $\boldsymbol{L}$ [1, 14]. The approaches given in [1, 14, 15] apply only to single attribute models. In this paper our objective is learn graphs from multi-attribute data.

Graph Laplacian matrix has been extensively used for embedding, manifold learning, clustering and semi-supervised learn-

This work was supported by NSF Grants CCF-1617610 and ECCS-2040536.

978-1-7281-6662-9/20/$31.00

ing [16, 17]; see [1, 14] for further references to applications to web page categorization with graph information, graph regularized sparse coding, and matrix completion. Graph-based transform coding and its potential application to signal/image compression is discussed in [18] where learning of the graph Laplacian matrix plays a key role.

Another set of approaches are based on statistical considerations under the graph Laplacian constraint [1, 12, 13] where Laplacian $\boldsymbol{L}$, after regularization, plays the role of inverse covariance $\boldsymbol{\Omega}$; $\boldsymbol{L}$ is a symmetric, nonnegative-definite matrix but with non-positive off-diagonal entries. These approaches too apply only to single attribute models (and when off-diagonal entries of inverse covariance are non-positive).

In this paper, we extend the single attribute approach of Kalofolias (2016) to multi-attribute data. An alternating direction method of multipliers (ADMM) algorithm is presented to optimize the objective function to infer the graph topology. Numerical results based on synthetic as well as real data are presented.

Notation: Given a matrix $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, tr$(\boldsymbol{A})$ denotes its trace. For a matrix $\boldsymbol{B} \in \mathbb{R}^{p\times q}$, we define its Frobenius norm as $\|\boldsymbol{B}\|_F = \sqrt{\text{tr}(\boldsymbol{B}^\top\boldsymbol{B})}$, and denote its (i,j)-th element by B_{ij}. We also denote B_{ij} by $[\boldsymbol{B}]_{ij}$. Also, $\boldsymbol{B}_{i\cdot}$ and $\boldsymbol{B}_{\cdot j}$ denote column vectors comprising ith row and jth column, respectively, of $\boldsymbol{B}$. Notation $\boldsymbol{1}_p$ denotes a column of p ones, $\boldsymbol{B}^+ = \max(\boldsymbol{B}, 0)$, $\boldsymbol{B}^- = \max(-\boldsymbol{B}, 0)$ and abs$(\boldsymbol{B}) = \boldsymbol{B}^+ + \boldsymbol{B}^-$, where "max" operation is elementwise.

2. BACKGROUND AND SYSTEM MODELS

2.1. Statistical Models

The conditional dependency structure among p random variables $x_1, x_1, \cdots, x_p$, components of $\boldsymbol{x} \in \mathbb{R}^p$, is represented using an undirected graph $\mathcal{G} = (V, \mathcal{E})$, where $V = [1, p]$ is the set of p nodes corresponding to the p random variables x_is, and $\mathcal{E} \subseteq V \times V$ is the set of undirected edges describing conditional dependencies among x_is. There is no edge between nodes i and j iff x_i and x_j are conditionally independent given the remaining p-2 variables x_ℓ, $\ell \in [1, p]$, $\ell \neq i$, $\ell \neq j$ [3, p. 60]. We will call $\mathcal{G}$ a **single-attribute graphical model** for $\boldsymbol{x}$.

Now consider p jointly Gaussian random vectors $\boldsymbol{z}_i \in \mathbb{R}^m$, $i = 1, 2, \cdots, p$. We associate $\boldsymbol{z}_i$ with the ith node of an undirected graph $\mathcal{G} = (V, \mathcal{E})$ where $V = [1, p]$ is the set of p nodes, and $\mathcal{E} \subseteq V \times V$ is the set of edges that describe the conditional dependencies among vectors $\{\boldsymbol{z}_i,\ i \in V\}$. As in the scalar case ($m = 1$), there is no edge between node i and node j in $\mathcal{G}$ (i.e., $\{i, j\} \notin \mathcal{E}$) iff random vectors $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ are conditionally independent given all the remaining random vectors $\boldsymbol{z}_\ell$ corresponding to the remaining $p - 2$ nodes in V, i.e., for $\ell \in V\backslash\{j, k\}$ [10, 11]. This is the **multi-attribute Gaussian graphical model**. The term multi-attribute Gaussian graphical model has been used in [9] for such models. In this paper, we will use a similar set-up under a smoothness constraint, but without statistical considerations.

Define the mp-vector

$$\boldsymbol{x} = [\boldsymbol{z}_1^\top\ \boldsymbol{z}_2^\top\ \cdots\ \boldsymbol{z}_p^\top]^\top \in \mathbb{R}^{mp}\,. \tag{3}$$

Suppose we have n i.i.d. observations $\boldsymbol{x}(t)$, $t = 1, 2, \cdots, n$, of $\boldsymbol{x}$. The objective in statistical multi-attribute graph learning is to estimate the inverse covariance matrix $(\mathbb{E}\{\mathbf{x}\mathbf{x}^\top\})^{-1}$ and to determine if $\{i, j\} \in \mathcal{E}$, given data $\{\boldsymbol{x}(t)\}_{t=1}^n$.

Let us associate $\boldsymbol{x}$ with an "enlarged" graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}})$, where $\bar{V} = [1, mp]$ and $\bar{\mathcal{E}} \subseteq \bar{V} \times \bar{V}$. Now $[\boldsymbol{z}_j]_\ell$, the ℓth component of $\boldsymbol{z}_j$ associated with node j of $\mathcal{G} = (V, \mathcal{E})$, is the random variable $x_q = [\boldsymbol{x}]_q$, where $q = (j-1)m + \ell$, $j = 1, 2, \cdots, p$ and $\ell = 1, 2, \cdots, m$. The random variable x_q is associated with node q of $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}})$. Corresponding to the edge $\{j, k\} \in \mathcal{E}$ in the multi-attribute $\mathcal{G} = (V, \mathcal{E})$, there are m^2 edges $\{q, r\} \in \bar{\mathcal{E}}$ specified by $q = (j-1)m+s$ and $r = (k-1)m+t$, where $s = 1, 2, \cdots, m$ and $t = 1, 2, \cdots, m$. The graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}})$ is a single-attribute graph. In order for $\bar{\mathcal{G}}$ to reflect the conditional independencies encoded in $\mathcal{G}$, we must have the equivalence

$$\{j, k\} \notin \mathcal{E} \Leftrightarrow \bar{\mathcal{E}}^{(jk)} \cap \bar{\mathcal{E}} = \emptyset\,, \text{ where} \tag{4}$$

$$\bar{\mathcal{E}}^{(jk)} = \big\{\{q, r\}\,\big|\, q = (j-1)m + s,\ r = (k-1)m + t,\ s, t = 1, 2, \cdots, m\big\}\,. \tag{5}$$

Let $\boldsymbol{R}_{xx} = \mathbb{E}\{\boldsymbol{x}\boldsymbol{x}^\top\} \succ \boldsymbol{0}$ and $\boldsymbol{\Omega} = \boldsymbol{R}_{xx}^{-1}$. Define the (j, k)th $m \times m$ subblock $\boldsymbol{\Omega}^{(jk)}$ of $\boldsymbol{\Omega}$ as

$$[\boldsymbol{\Omega}^{(jk)}]_{st} = [\boldsymbol{\Omega}]_{(j-1)m+s,(k-1)m+t}\,,\ s, t = 1, 2, \cdots, m\,. \tag{6}$$

It is established in [11, Sec. 2.1] that

$$\begin{aligned}\boldsymbol{\Omega}^{(jk)} = \boldsymbol{0} &\Leftrightarrow \boldsymbol{z}_j \text{ and } \boldsymbol{z}_k \text{ are conditionally independent}\\ &\Leftrightarrow \{j, k\} \notin \mathcal{E}\,.\end{aligned} \tag{7}$$

Since $\boldsymbol{\Omega}^{(jk)} = \boldsymbol{0}$ is equivalent to $[\boldsymbol{\Omega}]_{qr} = 0$ for every $\{q, r\} \in \bar{\mathcal{E}}^{(jk)}$, and since, by [2, Proposition 5.2], $[\boldsymbol{\Omega}]_{qr} = 0$ iff x_q and x_r are conditionally independent, hence, iff $\{q, r\} \notin \bar{\mathcal{E}}$, it follows that equivalence (4) holds true.

2.2. Smoothness-Based Graph Learning: Single-Attribute Models

Here we review [14]. With reference to (1) and (2), it is established in [14] that

$$\text{tr}(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X}) = \frac{1}{2}\text{tr}(\boldsymbol{W}\boldsymbol{Z}) \tag{8}$$

$$\text{where } \boldsymbol{W}, \boldsymbol{Z} \in \mathbb{R}^{p\times p},\ Z_{ij} = \|\boldsymbol{X}_{i\cdot} - \boldsymbol{X}_{j\cdot}\|_2^2 \tag{9}$$

and $\boldsymbol{W}$ is the weight matrix (or the weighted adjacency matrix) with $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, $\boldsymbol{W} = \boldsymbol{W}^\top$, $W_{ij} \geq 0$ and $W_{ii} = 0$ for $1 \leq i, j \leq p$. Instead of performing a penalized minimization of tr$(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X})$ to estimate $\boldsymbol{L}$, [14] minimizes a penalized tr$(\boldsymbol{W}\boldsymbol{Z})$ w.r.t. $\boldsymbol{W}$ for graph learning. Given $\boldsymbol{W}$, one has unique $\boldsymbol{L}$ and the edge-set $\mathcal{E}$. [14] shows that minimization of tr$(\boldsymbol{W}\boldsymbol{Z})$ is computationally more efficient than minimizing tr$(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X})$ (as is done in [15]).

Define the space $\mathcal{W}_p$ of all valid $p \times p$ weight matrices $\boldsymbol{W}$

$$\begin{aligned}\mathcal{W}_p = \Big\{&\boldsymbol{W} \in \mathbb{R}^{p\times p}\,:\, \boldsymbol{W} = \boldsymbol{W}^\top,\ W_{ij} \geq 0,\\ &W_{ii} = 0,\ 1 \leq i, j \leq p\Big\}\end{aligned} \tag{10}$$

The following optimization problem is solved in [14]:

$$\min_{\boldsymbol{W}\in\mathcal{W}_p} \text{tr}(\boldsymbol{W}\boldsymbol{Z}) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2 - \alpha\sum_{i=1}^p \ln\Big(\sum_{j=1}^p W_{ij}\Big) \tag{11}$$

with parameters $\alpha > 0$ and $\beta \geq 0$ controlling the "shape" (number and weights) of the edges. In (11), tr$(\boldsymbol{W}\boldsymbol{Z})$ is the main cost but minimizing it alone w.r.t. $\boldsymbol{W}$ is ill-posed ($\boldsymbol{W} = \boldsymbol{0}$ minimizes it). The sum $\sum_{j=1}^p W_{ij}$ is the node degree at node i and the term $\ln\Big(\sum_{j=1}^p W_{ij}\Big)$ is the logarithmic barrier on the node degrees. This

implies that the degrees are forced to be positive but do not prevent individual edges to vanish. It is noted in [14] that using only the logarithmic barrier ($\beta = 0$) leads to very sparse graphs, and changing α only changes the scale of the solution, not the sparsity pattern. Therefore, the term $\frac{\beta}{2}\|\boldsymbol{W}\|_F^2$ is added to control the graph sparsity: larger values of β lead to more dense connectivity. In [14], optimization is carried for fixed $\alpha = 1$ (search over β to a get a desired edge density) and then one scales $\boldsymbol{W}$ to obtain a desired $\|\boldsymbol{W}\|$; we refer the reader to [14] for further details.

In the next section we modify optimization problem (11) to apply to multi-attribute data using the enlarged graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}})$ discussed in Sec. 2.1.

3. GRAPH LEARNING FROM MULTI-ATTRIBUTE DATA

Consider (similar to Sec. 2.1, but without any probabilistic modeling) p data vector sequences $\boldsymbol{z}_i(t) \in \mathbb{R}^m$, $i = 1, 2, \cdots, p$, each vector with n samples $t = 1, 2, \cdots, n$. We associate $\boldsymbol{z}_i$ with the ith node of a weighted undirected graph $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ where $V = [1, p]$, $\mathcal{E} \subseteq V \times V$ and $\boldsymbol{W} \in \mathcal{W}_p$ (defined in (10)). Define $\boldsymbol{x}(t) \in \mathbb{R}^{mp}$ based on $\boldsymbol{z}_i(t)$s, as in (3). Then we have n samples $\{\boldsymbol{x}(t)\}_{t=1}^n$. Based on $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$, construct an enlarged weighted undirected graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}}, \bar{\boldsymbol{W}})$ with $\bar{\boldsymbol{W}} \in \mathcal{W}_{mp}$, similar to the enlarged graph $(\bar{V}, \bar{\mathcal{E}})$ based on $(V, \mathcal{E})$ as discussed in Sec. 2.1. Then the ℓth component $[\boldsymbol{z}_j(t)]_\ell$ of the data vector $\boldsymbol{z}_j(t)$ associated with node j of graph $\mathcal{G} = (V, \mathcal{E}, \boldsymbol{W})$ is now the scalar data variable $x_q(t) = [\boldsymbol{x}(t)]_q$ associated with node q of the enlarged graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}}, \bar{\boldsymbol{W}})$, where $q = (j-1)m + \ell$, $j = 1, 2, \cdots, p$ and $\ell = 1, 2, \cdots, m$. Corresponding to each edge $\{j, k\}$ in $\mathcal{G}$, there are m^2 edges in $\bar{\mathcal{G}}$ (see (4) and (5)).

Define the $mp \times n$ matrix

$$\bar{\boldsymbol{X}} = \begin{bmatrix} x_1(1) & x_1(2) & \cdots & x_1(n) \\ x_2(1) & x_2(2) & \cdots & x_2(n) \\ \vdots & \vdots & \ddots & \vdots \\ x_{mp}(1) & x_{mp}(2) & \cdots & x_{mp}(n) \end{bmatrix} \tag{12}$$

$$= \begin{bmatrix} \boldsymbol{z}_1(1) & \boldsymbol{z}_1(2) & \cdots & \boldsymbol{z}_1(n) \\ \boldsymbol{z}_2(1) & \boldsymbol{z}_2(2) & \cdots & \boldsymbol{z}_2(n) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{z}_p(1) & \boldsymbol{z}_p(2) & \cdots & \boldsymbol{z}_p(n) \end{bmatrix}. \tag{13}$$

Define $\bar{\boldsymbol{Z}} \in \mathbb{R}^{(mp)\times(mp)}$ with (i,j)th elements $\bar{Z}_{ij} = \|\bar{\boldsymbol{X}}_{i.} - \bar{\boldsymbol{X}}_{j.}\|_2^2$. Following (8) and (9), and using the graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}}, \bar{\boldsymbol{W}})$, consider

$$\mathrm{tr}(\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{L}} \bar{\boldsymbol{X}}) = \frac{1}{2}\mathrm{tr}(\bar{\boldsymbol{W}}\bar{\boldsymbol{Z}}) = \frac{1}{2}\sum_{i=1}^{mp}\sum_{j=1}^{mp} \bar{W}_{ij}\bar{Z}_{ij} \tag{14}$$

$$= \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{p}\left[\sum_{s=1}^{m}\sum_{k=1}^{m} \bar{W}_{qr}\bar{Z}_{qr}\right], \quad \begin{matrix} q = (i-1)m + s, \\ r = (j-1)m + k\,. \end{matrix} \tag{15}$$

Now observe that the underlying graph for multi-attribute data is a p-node graph $\mathcal{G}$, not the enlarged mp-node graph $\bar{\mathcal{G}}$, implying that the strength of edge $\{i, j\}$ between nodes i and j of $\mathcal{G}$ which is W_{ij}, should be the strength of connection of all m^2 edges between nodes $q = (i-1)m + s$ and $r = (j-1)m + k$, $(s, k = 1, 2, \cdots, m)$, of the enlarged graph $\bar{\mathcal{G}}$, corresponding to nodes i and j of $\mathcal{G}$. We enforce this by requiring that for any $\{i, j\}$,

$$\bar{W}_{qr} = W_{ij} \text{ for } \begin{matrix} q = (i-1)m + s,\ r = (j-1)m + k, \\ s, k = 1, 2, \cdots, m\,. \end{matrix} \tag{16}$$

Then we have

$$\mathrm{tr}(\bar{\boldsymbol{W}}\bar{\boldsymbol{Z}}) = \sum_{i=1}^{p}\sum_{j=1}^{p} W_{ij}\left[\sum_{s=1}^{m}\sum_{k=1}^{m}\bar{Z}_{qr}\right] = \mathrm{tr}(\boldsymbol{W}\tilde{\boldsymbol{Z}}) \tag{17}$$

where $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{p\times p}$ with (i,j)th component

$$\tilde{Z}_{ij} = \sum_{s,k=1}^{m} \|\bar{\boldsymbol{X}}_{q.} - \bar{\boldsymbol{X}}_{r.}\|_2^2 = \sum_{s,k=1}^{m}\left(\frac{1}{n}\sum_{t=1}^{n}(x_q(t) - x_r(t))^2\right) \tag{18}$$

and q, r are as in (16).

Following [14], we have the regularized/penalized optimization problem (as in (11))

$$\min_{\boldsymbol{W}\in\mathcal{W}_p} \mathrm{tr}(\boldsymbol{W}\tilde{\boldsymbol{Z}}) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2 - \alpha\sum_{i=1}^{p}\ln\Big(\sum_{j=1}^{p} W_{ij}\Big) \tag{19}$$

We provide a solution in Sec. 3.1.

3.1. Optimization

We will use the alternating direction method of multipliers (ADMM) approach [20] with variable splitting. The solution of [14] is different. Using variable splitting, consider (recall $W_{ii} = 0$)

$$\min_{\boldsymbol{W}\in\mathcal{W}_p, \boldsymbol{\theta}\in\mathbb{R}^p} \Big\{\mathrm{tr}(\boldsymbol{W}\tilde{\boldsymbol{Z}}) - \alpha\sum_{i=1}^{p}\ln(\theta_i) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2\Big\} \tag{20}$$

$$\text{subject to} \quad \theta_i = \sum_{j=1, j\neq i}^{p} W_{ij}\,, i = 1, 2, \cdots, p. \tag{21}$$

The scaled augmented Lagrangian for this problem is [20] ($\boldsymbol{1}_p$ is a column of all ones)

$$L_\rho = \mathrm{tr}(\boldsymbol{W}\tilde{\boldsymbol{Z}}) - \alpha\sum_{i=1}^{p}\ln(\theta_i) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{\theta} - \boldsymbol{W}\boldsymbol{1}_p + \boldsymbol{u}\|_2^2 \tag{22}$$

where $\boldsymbol{u} \in \mathbb{R}^p$ is the dual variable, and $\rho > 0$ is the penalty parameter. Given the results $\boldsymbol{\theta}^{(k)}, \boldsymbol{W}^{(k)}, \boldsymbol{u}^{(k)}$ of the kth iteration, in the $(k+1)$th iteration, an ADMM algorithm executes the following three updates:

(a) With $L_a(\boldsymbol{W}) := \mathrm{tr}(\boldsymbol{W}\tilde{\boldsymbol{Z}}) + \frac{\beta}{2}\|\boldsymbol{W}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{\theta}^{(k)} - \boldsymbol{W}\boldsymbol{1}_p + \boldsymbol{u}^{(k)}\|_2^2$, let $\boldsymbol{W}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{W}\in\mathcal{W}_p} L_a(\boldsymbol{W})$.

(b) With $L_b(\boldsymbol{\theta}) := -\alpha\sum_{i=1}^p \ln(\theta_i) + \frac{\rho}{2}\|\boldsymbol{\theta} - \boldsymbol{W}^{(k+1)}\boldsymbol{1}_p + \boldsymbol{u}^{(k)}\|_2^2$, let $\boldsymbol{\theta}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\theta}} L_b(\boldsymbol{\theta})$.

(c) $\boldsymbol{u}^{(k+1)} \leftarrow \boldsymbol{u}^{(k)} + \boldsymbol{\theta}^{(k+1)} - \boldsymbol{W}^{(k+1)}\boldsymbol{1}_p$.

In update (a) above, cost $L_a(\boldsymbol{W})$ is separable row-wise in $\boldsymbol{W}$: $L_a(\boldsymbol{W}) = \sum_{i=1}^p L_{ai}(\check{\boldsymbol{W}}_{i.})$ where $\check{\boldsymbol{W}}_{i.} \in \mathbb{R}^{p-1}$ is obtained from $\boldsymbol{W}_{i.}$ by deleting its ith row (recall that $W_{ii} = 0$), similarly, $\check{\tilde{\boldsymbol{Z}}}_{i.} \in \mathbb{R}^{p-1}$ is obtained from $\tilde{\boldsymbol{Z}}_{i.}$ by deleting its ith row, and

$$L_{ai}(\check{\boldsymbol{W}}_{i.}) = \check{\boldsymbol{W}}_{i.}^\top \check{\tilde{\boldsymbol{Z}}}_{i.} + \frac{\beta}{2}\check{\boldsymbol{W}}_{i.}^\top\check{\boldsymbol{W}}_{i.} + \frac{\rho}{2}(\theta_i^{(k)} - \check{\boldsymbol{W}}_{i.}^\top\boldsymbol{1}_{p-1} + u_i^{(k)})^2$$

$$= \check{\boldsymbol{W}}_{i.}^\top\left(\frac{\beta}{2} + \frac{\rho}{2}\boldsymbol{1}_{p-1}\boldsymbol{1}_{p-1}^\top\right)\check{\boldsymbol{W}}_{i.} + \frac{\rho}{2}(\theta_i^{(k)} - u_i^{(k)})^2$$

$$+ \left(\check{\tilde{\boldsymbol{Z}}}_{i.} - \rho(\theta_i^{(k)} + u_i^{(k)})\boldsymbol{1}_{p-1}\right)^\top\check{\boldsymbol{W}}_{i.} \tag{23}$$

We need to minimize $L_{ai}(\check{\boldsymbol{W}}_{i\cdot})$ w.r.t. $\check{\boldsymbol{W}}_{i\cdot}$ subject to $W_{ij} \geq 0$. An iterative solution is given in [19]. [19] minimizes $(1/2)\boldsymbol{y}^\top \boldsymbol{Q}\boldsymbol{y} - \boldsymbol{y}^\top \boldsymbol{h}$ w.r.t. $\boldsymbol{y} \in \mathbb{R}^q$ subject to $y_\ell \geq 0\ \forall \ell$, where $\boldsymbol{Q}$ is symmetric, positive-definite and $\boldsymbol{h}$ is arbitrary. The monotonically convergent iterative solution of [19] is

$$y_\ell \leftarrow y_\ell \left[\frac{2[\boldsymbol{Q}^- \boldsymbol{y}]_\ell + h_\ell^+ + \delta}{[\text{abs}(\boldsymbol{Q})\boldsymbol{y}]_\ell + h_\ell^- + \delta} \right] \tag{24}$$

where $0 < \delta \ll 1$. In our problem of minimization of $L_{ai}(\check{\boldsymbol{W}}_{i\cdot})$ w.r.t. $\check{\boldsymbol{W}}_{i\cdot}$ subject to $W_{ij} \geq 0$, we have $\boldsymbol{Q} = \frac{\beta}{2} + \frac{\rho}{2}\mathbf{1}_{p-1}\mathbf{1}_{p-1}^\top$ independent of row i of $\boldsymbol{W}$, and $\boldsymbol{h} = \check{\boldsymbol{Z}}_{i\cdot} - \rho(\theta_i^{(k)} + u_i^{(k)})\mathbf{1}_{p-1}$ which depends upon row i of $\boldsymbol{W}$. Since all elements of $\boldsymbol{Q}$ for our problem are positive, $\boldsymbol{Q}^- = \mathbf{0}$ and $\boldsymbol{Q}^+ = \boldsymbol{Q}$.

Now we turn to update (b). Notice that $L_b(\boldsymbol{\theta})$ is completely separable w.r.t. each component of $\boldsymbol{\theta}$: $L_b(\boldsymbol{\theta}) = \sum_{i=1}^p L_{bi}(\theta_i)$, where $L_{bi}(\theta_i) = -\alpha \ln(\theta_i) + (\rho/2)(\theta_i - \boldsymbol{W}_{i\cdot}^{(k+1)}\mathbf{1}_p + u_i^{(k)})^2$. Setting

$$\begin{aligned} 0 =& \frac{\partial L_{bi}(\theta_i)}{\partial \theta_i} = -\frac{\alpha}{\theta_i} + \rho\,(\theta_i - c_i) \\ \text{where } \ c_i =& \boldsymbol{W}_{i\cdot}^{(k+1)}\mathbf{1}_p - u_i^{(k)}\,. \end{aligned} \tag{25}$$

This leads to a quadratic equation in θ_i. Since the ith node degree θ_i must be positive, we take

$$\theta_i^{(k+1)} = \frac{1}{2}\left(c_i + \sqrt{c_i + (4\alpha/\rho)}\right) \tag{26}$$

where c_i is specified in (25).

Finally, update (c) follows from [20].

We initialize as $\boldsymbol{u}^{(0)} = \mathbf{0}$, $\boldsymbol{W}^{(0)} = \mathbf{0}$ and $\boldsymbol{\theta} = \mathbf{1}_p$. For all numerical results presented later, we used $\rho = 2$ and set $\alpha = 1$.

As noted in [14], the objective function (19) is proper, convex, and lower-semicontinuous. Hence the ADMM algorithm is guaranteed to converge [20, Sec. 3.2], in the sense that we have primal residual convergence to 0, dual residual convergence to 0, and objective function convergence to the optimal value.

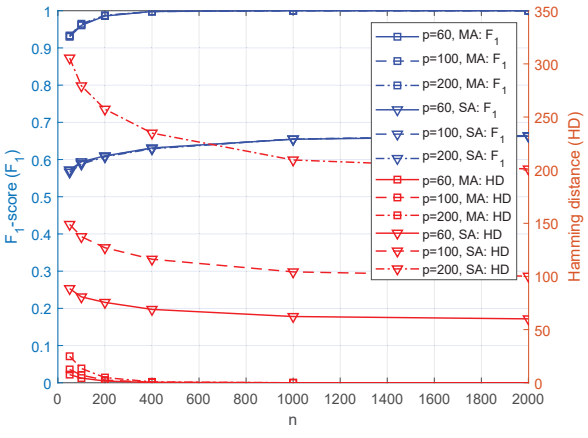


Fig. 1: F_1-score and Hamming distance for the chain graph with $m = 3$ attributes per node. The label MA refers to multi-attribute data, and SA refers to single attribute data using just the first attribute at each node. $\alpha = 1$, and $\beta = 10$ for MA graph learning and $\beta = 1$ for SA graph learning.

4. NUMERICAL EXAMPLES

4.1. Synthetic Data

We consider a chain graph (an example in [11]) where p nodes are connected in succession. We use a probabilistic/statistical model to generate multi-attribute data, but apply signal smoothness-based graph learning. Using the notation of (6), we set $[\boldsymbol{\Omega}^{(jk)}]_{st} = -0.5^{|s-t|}$ for $j = k = 1, \cdots, p$ and $s \neq t = 1, \cdots, m$ (off-diagonal terms are non-positive). For $j \neq k$, if the two nodes are not connected, we have $\boldsymbol{\Omega}^{(jk)} = \mathbf{0}$, and if nodes j and k are connected in the chain graph, then $[\boldsymbol{\Omega}^{(jk)}]_{st} = -0.2$ if $s \neq t$, and $[\boldsymbol{\Omega}^{(jk)}]_{st} = 0$ if $s = t$. Then we fill in the diagonal terms of $\boldsymbol{\Omega}$ to make it a valid graph Laplacian. This is simulation example 3 in [11, Sec. 5.1] except that off-diagonal terms of $\boldsymbol{\Omega}$ are non-positive. Now add $\gamma \boldsymbol{I}$ to $\boldsymbol{\Omega}$ with γ picked to make minimum eigenvalue of $\boldsymbol{\Omega} + \gamma \boldsymbol{I}$ equal to 0.001. With $\boldsymbol{\Phi}\boldsymbol{\Phi}^\top = (\boldsymbol{\Omega} + \gamma \boldsymbol{I})^{-1}$, we generate $\boldsymbol{x} = \boldsymbol{\Phi}\boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^{mp}$ as Gaussian $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. We generate n i.i.d. observations from $\boldsymbol{x}$, with $m = 3$, $p \in \{60, 100, 200\}$, $n \in \{50, 100, 200, 400, 1000, 2000\}$. The true value of $|\mathcal{E}| = p-1$.

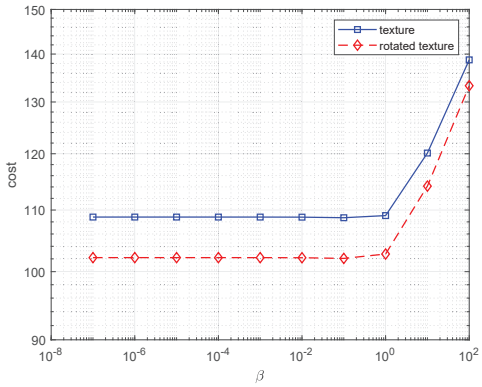


Fig. 2: Cross-validation cost (19) versus β, for β selection for image graph fitting

Simulation results based on 100 runs are shown in Fig. 1. We applied our proposed multi-attribute (labeled MA in figures) graph learning approach to estimate $\boldsymbol{W}$, hence the edgeset $\mathcal{E}$, along with the restriction of this approach to a single attribute (labeled SA in figures; we used only the first attribute at each node, results for other attributes were very similar). We fixed $\alpha = 1$ and selected β over a grid of values to maximize the F_1-score*. The selected values were $\beta = 10$ for multi-attribute and $\beta = 1$ single attribute; the results are not very sensitive to the choice of β. Fig. 1 shows the F_1-score and the Hamming distance between the true and estimated edge sets $\mathcal{E}_0$ and $\hat{\mathcal{E}}$, respectively. It is seen that multi-attribute based graph learning is significantly better than single attribute graph learning, since for same number of nodes and sample size, the former yields significantly higher F_1-score and significantly lower Hamming distance. As sample size n increases, the Hamming distance tends to zero for multi-attribute based graph learning, indicating that the the edgeset $\mathcal{E}$ is learned perfectly.

Since computation the F_1-score requires knowledge of true $\mathcal{E}_0$, in practice, one would select β via cross-validation or some infor-

*It is defined as $F_1 = 2 \times \text{precision} \times \text{recall}/(\text{precision} + \text{recall})$ where precision $= |\hat{\mathcal{E}} \cap \mathcal{E}_0|/|\hat{\mathcal{E}}|$, recall $= |\hat{\mathcal{E}} \cap \mathcal{E}_0|/|\mathcal{E}_0|$, and $\mathcal{E}_0$ and $\hat{\mathcal{E}}$ denote the true and estimated edge sets, respectively.

mation criterion-based approach. This is illustrated for the real data set later.

Algorithm 1 ADMM Algorithm for Multi-Attribute Graph learning

Input: Number of samples n, number of nodes p, number of attributes m, data $\{\boldsymbol{x}(t)\}_{t=1}^{n}$, $\boldsymbol{x} \in \mathbb{R}^{mp}$, regularization parameters α and β, ADMM penalty parameter ρ, and $\delta = 10^{-16}$.

Output: estimated weight matrix $\boldsymbol{W}$ and edge-set $\mathcal{E}$

Calculate the (squared) distance matrix $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{p\times p}$ with (i,j)th component $\tilde{Z}_{ij} = \sum_{s,k=1}^{m} \left(\frac{1}{n}\sum_{t=1}^{n}(x_q(t) - x_r(t))^2\right)$, where $q = (i-1)m + s$ and $r = (j-1)m + k$.

$\boldsymbol{Q} = \frac{\beta}{2} + \frac{\rho}{2}\mathbf{1}_{p-1}\mathbf{1}_{p-1}^{\top}$, $\boldsymbol{P} = \boldsymbol{Q}^{-1}$.

Initialize: $\boldsymbol{u}^{(0)} = \mathbf{0} \in \mathbb{R}^p$, $\boldsymbol{W}^{(0)} = \mathbf{0} \in \mathbb{R}^{p\times p}$ and $\boldsymbol{\theta} = \mathbf{1}_p$= column of p ones.

for $k = 0, 1, 2, \ldots$, until convergence, **do**

  for $i = 1, 2, \cdots, p$ **do**

    Define $\tilde{\tilde{\boldsymbol{Z}}}_{i\cdot} \in \mathbb{R}^{p-1}$ from $\tilde{\boldsymbol{Z}}_{i\cdot}$ by deleting its ith row:

$$\tilde{\tilde{\boldsymbol{Z}}}_{ij} = \begin{cases} \tilde{\boldsymbol{Z}}_{ij}, & 1 \le j \le i-1 \\ \tilde{\boldsymbol{Z}}_{i(j+1)}, & i \le j \le p-1\,. \end{cases}$$

    $\boldsymbol{h} = \tilde{\tilde{\boldsymbol{Z}}}_{i\cdot} - \rho(\theta_i^{(k)} + u_i^{(k)})\mathbf{1}_{p-1}$

    $\boldsymbol{y}^{(0)} = \max(\boldsymbol{P}\boldsymbol{h}, \mathbf{0}) \in \mathbb{R}^{p-1}$

    for $q = 1, 2, \ldots$, until convergence, **do**

      for $\ell = 1, 2, \cdots, p-1$ **do**

        $y_\ell^{(q)} = y_\ell^{(q-1)} \left[\frac{h_\ell^{+} + \delta}{[\boldsymbol{Q}\boldsymbol{y}^{(q-1)}]_\ell + h_\ell^{-} + \delta}\right]$

      end for

    end for

    Update ith row of $\boldsymbol{W}$ as

$$\boldsymbol{W}_{ij}^{(k+1)} = \begin{cases} y_j, & 1 \le j \le i-1 \\ 0, & j = i \\ y_{j-1}, & i+1 \le j \le p\,. \end{cases}$$

  end for

  for $i = 1, 2, \cdots, p$ **do**

    $c_i = \boldsymbol{W}_{i\cdot}^{(k+1)}\mathbf{1}_p - u_i^{(k)}$

    $\theta_i^{(k+1)} = \frac{1}{2}\left(c_i + \sqrt{c_i + (4\alpha/\rho)}\right)$

  end for

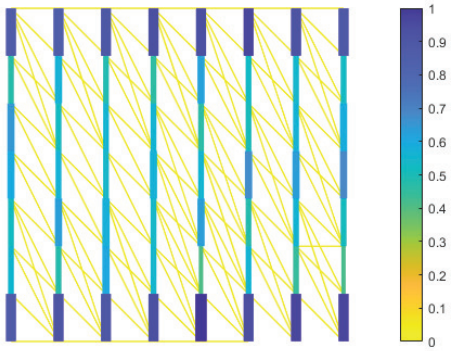
  $\boldsymbol{u}^{(k+1)} \leftarrow \boldsymbol{u}^{(k)} + \boldsymbol{\theta}^{(k+1)} - \boldsymbol{W}^{(k+1)}\mathbf{1}_p$

end for

4.2. Real Data Example: Graphs of Color Texture Images

Following [12, 13] where grayscale texture images from a USC database are considered, we now consider color textures from the Amsterdam Library of Textures (ALOT) [21]. We use two versions of the image 108 from [21], c111.png and c111r60.png (400×400 patches shown in Figs. 3 and 4), photographed from different angles. The 400×400 patches were partitioned into non-overlapping 8×8 blocks, vectorized into 64-pixel columns, 3 colors associated with each pixel. Thus, we have $m = 3$, $p = 64$ and $n = 2500$. The data were centered and mean-square value normalized to one before processing. To select β (α=1 throughout), we use cross-validation with criterion C_{cv}= the objective function (19), as follows. We random sample 70% of n (=2500) samples as training data for model fitting and use the remaining 30% samples as test data to compute C_{cv}. We average C_{cv} over 30 random samples for β values over a grid; the value of β is selected to minimize average C_{cv}. With the selected $\hat{\beta}$, we perform model fitting over the entire dataset (with $\alpha = 1$). The average cost C_{cv} for graph learning for the two color images displayed in Fig. 3(a) and Fig. 4(a), are shown in Fig. 2. For both textures, we get $\hat{\beta} = 0.1$. These values were used for final graph model fitting. We take the final estimated $\boldsymbol{W}$ resulting from (multi-attribute) ADMM algorithm as edge weights, normalize maximum value to one, and show the resulting graphs (arranged as 8×8 nodes) with colored edge weights and link thickness also reflecting edge weight. Compare Figs. 3(a) and 3(b), and Figs. 4(a) and 4(b), respectively, to note that the strong link weights follow the texture orientation: vertical in Figs. 3(a) and 3(b), and horizontal in Figs. 4(a) and 4(b). This provides significant "visual" support for the fitted graphs.

(a) a texture (from [21])



(b) image graph of (a): number of edges =458

Fig. 3: Color texture graph example

5. CONCLUSIONS

The problem of how to learn the graph combinatorial Laplacian matrix defining the structure of an undirected weighted graph underlying a set of smooth multi-attribute signals, was addressed. We extended the single attribute approach of Kalofolias (2016) to multi-attribute data. In a typical single-attribute graph modeling problem, one associates a scalar data variable with each node of the graph, whereas in multi-attribute graphical models, each node represents a

vector. Image graphs for grayscale images for modeling dependence of a pixel on neighboring pixels is an example of single-attribute graph modeling. These approaches do not apply to color images where one has three variables (RGB color components) per pixel node. Image graphs for color images is an example of multi-attribute graphical models. An ADMM algorithm was presented to optimize the objective function to infer the graph topology. We tested the proposed approach on synthetic as well as real data (color image graphs).

6. REFERENCES

[1] X. Dong, D. Thanou, M. Rabbat and P. Frossard, "Learning graphs from data," *IEEE Signal Process. Mag.*, pp. 44-63, May 2019.

[2] S.L. Lauritzen, *Graphical models*. Oxford, UK: Oxford Univ. Press, 1996.

[3] J. Whittaker, *Graphical Models in Applied Multivariate Statistics*. New York: Wiley, 1990.

[4] P. Danaher, P. Wang and D.M. Witten, "The joint graphical lasso for inverse covariance estimation across multiple classes," *J. Royal Statistical Society, Series B (Methodological)*, vol. 76, pp. 373-397, 2014.

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

[6] K. Mohan, P. London, M. Fazel, D. Witten and S.I. Lee, "Node-based learning of multiple Gaussian graphical models," *J. Machine Learning Research*, vol. 15, 2014.

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

[8] O. Banerjee, L.E. Ghaoui and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *J. Machine Learning Research*, vol. 9, pp. 485-516, 2008.

[9] J. Chiquet, G. Rigaill and M. Sundquist, "A multiattribute Gaussian graphical model for inferring multiscale regulatory networks: an application in breast cancer." In: Sanguinetti G., Huynh-Thu V. (eds), *Gene Regulatory Networks. Methods in Molecular Biology*, vol 1883. Humana Press, New York, NY, 2019.

[10] M. Kolar, H. Liu and E.P. Xing, "Markov network estimation from multi-attribute data," in *Proc. 30th Intern. Conf. Machine Learning (ICML)*, Atlanta, GA, 2013.

[11] M. Kolar, H. Liu and E.P. Xing, "Graph estimation from multi-attribute data," *J. Machine Learning Research*, vol. 15, pp. 1713-1750, 2014.

[12] E. Pavez and A. Ortega, "Generalized Laplacian precision matrix estimation for graph signal processing," in *Proc. IEEE ICASSP 2016*, Shanghai, China, March 2016, pp. 6350-6354.

[13] E. Pavez, H.E. Egilmez and A. Ortega, "Learning graphs with monotone topology properties and multiple connected components," *IEEE Trans. Signal Process.*, vol. 66, no. 9, pp. 2399-2413, May 1, 2018.

[14] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. 19th Intern. Conf. Artificial Intelligence & Statistics (AISTATS)*, Cadiz, Spain, 2016.

[15] X. Dong, D. Thanou, P. Frossard and P. Vandergheynst "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160-6173, Dec. 1, 2016.

[16] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering.," in *Proc. NIPS*, vol. 14, pp. 585-591, 2001.

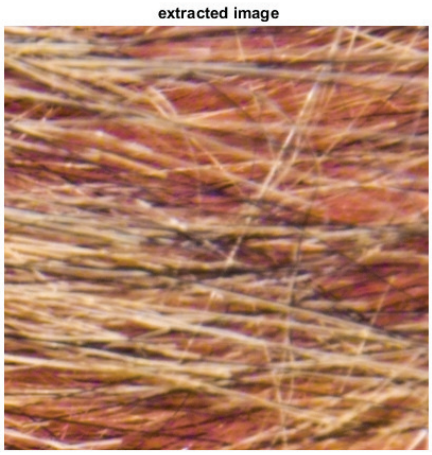
[17] M. Belkin, P. Niyogi and V. Sindhwani "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Machine Learning Research*, vol. 7, pp. 2399-2434, 2006.

[18] G. Fracastoro, D. Thanou and P. Frossard, "Graph transform optimization with application to image compression," *IEEE Trans. Image Process.*, vol. 29, pp. 419-432, 2020.

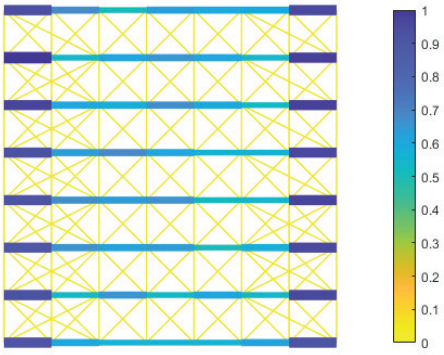
[19] X. Xiao and D. Chen, "Multiplicative iteration for nonnegative quadratic programming," *arXiv:1406.1008v1 [math.NA]*, 4 June 2014.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1-122, 2010.

[21] Amsterdam Library of Textures (ALOT), `http://aloi.science.uva.nl/public_alot`.



(a) "rotated" texture (from [21])



(b) image graph of (a): number of edges =640

Fig. 4: Another color texture graph example