Kernel-based Joint Multiple Graph Learning and Clustering of Graph Signals

Mohamad H. Alizade, Aref Einizade, and Jhony H. Giraldo

***Abstract*—Within the context of Graph Signal Processing (GSP), Graph Learning (GL) is concerned with the inference of the graph's underlying structure from nodal observations. However, real-world data often contains diverse information, necessitating the simultaneous clustering and learning of multiple graphs. In practical applications, valuable node-specific covariates, represented as kernels, have been underutilized by existing graph signal clustering methods. In this letter, we propose a new framework, named Kernel-based joint Multiple GL and clustering of graph signals (KMGL), that leverages a multi-convex optimization approach. This allows us to integrate node-side information, construct low-pass filters, and efficiently solve the optimization problem. The experiments demonstrate that KMGL significantly enhances the robustness of GL and clustering, particularly in scenarios with high noise levels and a substantial number of clusters. These findings underscore the potential of KMGL for improving the performance of GSP methods in diverse, real-world applications.**

***Index Terms*—Graph signal processing, graph learning, clustering, kernel subspace.**

I. Introduction

The emerging field of Graph Signal Processing (GSP) has introduced a plethora of analytical techniques [1]–[3]. GSP focuses on the manipulation and analysis of data represented as signals associated with the nodes of a meaningful graph. While some datasets, like traffic data, naturally exhibit graph-like structures, many others lack a known graph topology [4]. This has stimulated the growing popularity of Graph Learning (GL) within GSP [5], [6]. GL encompasses various approaches, including those that employ physical processes like diffusion for data interpretation [7], [8] and methods that assume neighboring nodes exhibit similar values, promoting global smoothness within the graph [4], [9]–[11].

Previous GL methods have primarily dealt with homogeneous datasets, where the data is associated with a single graph [4], [9]. However, many real-world datasets are heterogeneous, comprising clusters with diverse underlying structures. This heterogeneity results in the partition of graph signals, where each partition corresponds to a distinct, often unknown, graph. For example, in fMRI datasets, brain imaging reveals various cognitive processes across different parts of the brain [12]. Each graph signal in such datasets may correspond to a separate cognitive process and, consequently, a distinct functional network [13]. Additionally, there is often node-specific information, such as spatial coordinates in a sensor network, or non-numeric data like categories or text [14]. Recent efforts have emerged to address the simultaneous GL and clustering of graph signals in these complex scenarios [12], [15]–[17]. Yet, none of these methods have effectively exploited node-side information to enhance their performance. Moreover, they typically do not reconstruct the filtered (noiseless) graph signals, which represents a significant practical limitation.

In this letter, we introduce a new framework that combines node-specific information to simultaneously cluster the graph signals and learn the graph's underlying topology. To achieve this, we map the node-specific information into elements of a kernel's Hilbert space. The kernel matrix represents the covariates of the relationship between nodes in the Hilbert space, and we create low-pass filters by combining the Laplacian matrix with the inverse of the kernel matrices. Thus, we introduce an iterative approach called the Kernel-based joint Multiple Graph Learning and clustering of graph signals (KMGL[1]) algorithm, inspired by the K-means clustering framework [18] and kernel-based GL methods [14]. To efficiently solve our optimization problem, we employ the Block Coordinate Descent (BCD) method [19]. We link our GL task to a widely studied least squares problem, enabling us to optimize and solve our framework effectively.

Our work brings several contributions to the field: i) This is the first study to incorporate node-specific information into multiple GL and signal clustering while also obtaining denoised graph signals. ii) We demonstrate the convergence of the KMGL algorithm by exploiting the multi-convexity of the optimization problem. iii) Our experiments reveal that leveraging node-specific information significantly enhances the robustness of GL and clustering, particularly when dealing with high levels of noise and a large number of clusters. In contrast, existing methods struggle with severe performance deterioration. iv) We extend our framework to handle cases where data is missing, providing further flexibility and practicality (see Appendix A for details).

II. Preliminaries

1) Notation: Vectors, matrices, and sets are denoted by boldface lowercase, boldface capital, and calligraphic capital letters, respectively. The notations $(\cdot)^\top$, $\mathrm{tr}(\cdot)$, $\|\cdot\|_p$, and $\|\cdot\|_F$ stand for the transpose operator, the trace operator, the p-norm of a vector, and the Frobenious norm of a matrix, respectively. The matrix $\mathrm{diag}(\mathbf{a})$ is a diagonal matrix with the elements of the vector $\mathbf{a}$ on its principal diagonal. The (i,j)th and ith

M. H. Alizade (corresponding author) is with the Electrical Engineering Department of Sharif Uni. of Tech., Iran. E-mail: mhmd.h.alizade98@gmail.com.

A. Einizade and J. H. Giraldo are with LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France. E-mail: {aref.einizade, jhony.giraldo}@telecom-paris.fr.

This work was partially supported by the center Hi! PARIS.

[1] https://github.com/mohamad-h-alizade/KMGL

elements of a matrix $\mathbf{M}$ and a vector $\mathbf{x}$ are denoted as $\mathbf{M}_{ij}$ and x_i, respectively. The cardinality of set $\mathcal{I}$ is stated by $|\mathcal{I}|$.

2) Graph Signals: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ be a weighted undirected graph without self-loops, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the node set with $|\mathcal{V}| = n$, the edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, and $\mathbf{W}$ is the symmetric adjacency matrix. The entity $\mathbf{W}_{ij}$ has a positive value if there is an edge between vertices v_i and v_j but zero otherwise. Let $\mathbf{D} = \mathrm{diag}(\mathbf{W}\mathbf{1})$ be the diagonal degree matrix, where $\mathbf{1}$ is the all-one vector of size n. The Laplacian matrix of $\mathcal{G}$ given by $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is symmetric and positive semi-definite with eigendecomposition $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ [2]. The Graph Fourier Transform (GFT) is defined in terms of $\mathbf{U}$. Formally, a graph signal is a function $x : \mathcal{V} \to \mathbb{R}$ isomorphic to $\mathbb{R}^n$, and forms the graph signal $\mathbf{x} \in \mathbb{R}^n$ consisting of node real values. Therefore, the GFT of $\mathbf{x}$ is given by $\tilde{\mathbf{x}} = \mathbf{U}^\top \mathbf{x}$, where $\tilde{x}_i$ is the spectral component of the ith eigenvector [2].

A graph signal is smooth if connected nodes with a larger weight have more similar values [4]. This is measured via the Laplacian's quadratic form $\mathbf{x}^\top \mathbf{L} \mathbf{x} = \sum_{i,j \in \mathcal{E}} \mathbf{W}_{ij}(x_i - x_j)^2$. Equivalently, this quadratic form can be expressed in terms of graph spectral components:

$$\mathbf{x}^\top \mathbf{L} \mathbf{x} = \mathbf{x}^\top \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top \mathbf{x} = \tilde{\mathbf{x}}^\top \boldsymbol{\Lambda} \tilde{\mathbf{x}} = \sum_{i=1}^{n} \lambda_i \tilde{x}_i^2, \tag{1}$$

where $\lambda_i = \boldsymbol{\Lambda}_{ii}$ (the ith eigenvalue of $\mathbf{L}$) has a frequency-like interpretation [5]. With this notion of frequency, $h(\lambda_i)$ forms a graph filter that either amplifies or attenuates each spectral component. The filtered graph signal

$$\mathbf{y} = \mathbf{U} h(\boldsymbol{\Lambda}) \mathbf{U}^\top \mathbf{x} = h(\mathbf{L})\mathbf{x} \tag{2}$$

can be characterized by applying a linear operator in terms of $\mathbf{L}$. Common choices for low-pass filtering includes $\mathbf{L}^{-1/2}$ or $(\mathbf{I} + \gamma \mathbf{L})^{-1}$ (for some scalar $\gamma > 0$), and conversely $\mathbf{L}$ for high-pass filtering [2], [3].

III. Kernel Multiple Graph Learning (KMGL)

1) Problem Statement: We are given a (normalized) dataset $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^n\}_{i=1}^m$ of graph signals residing on the shared vertex set $\mathcal{V}$ and a set of node-side information $\mathcal{P} = \{\mathcal{P}_1, \cdots, \mathcal{P}_n\}$ with $\mathcal{P}_i$ denoting a prior covariate for $v_i \in \mathcal{V}$. The objective of this study is to partition $\mathcal{X}$ into K clusters $\{\mathcal{X}_k\}_{k=1}^K$ (where $|\mathcal{X}_k| = m_k$) and learn their associated graphs $\{\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k, \mathbf{W}_k)\}_{k=1}^K$ that best fit their partition in terms of graph signal smoothness and the node-side information.

2) KMGL: To solve this problem, we select kernels [20] to separate the representation of information from the algorithm. The selection of kernels also permits the processing of different features in each cluster and consequently captures various types of relationships in each graph. This allows us to implicitly transform each node-side information to a high-dimensional feature vector without conducting any direct computation.

Let $\mathcal{K} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ be a symmetric positive definite kernel on the node-side information. Based on the Aronszajn theorem [21] there is a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ and a feature map $\phi : \mathcal{P} \to \mathcal{H}$ such that $\mathcal{K}(\mathbf{p}_i, \mathbf{p}_j) = \langle \phi(\mathbf{p}_i), \phi(\mathbf{p}_j) \rangle_{\mathcal{H}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product in the kernel space $\mathcal{H}$. This maps (possibly infinite) features to each node. The set of node features $\mathcal{F} = \{\mathbf{f} \in \mathbb{R}^n \mid f_i = \phi(\mathbf{p}_i)_d, d \in dims(\mathcal{H})\}$ represents all the n-dimensional feature vectors associated with the kernel. The graph signal $\mathbf{x}$ is expected to match the node-side information and may not deviate from the set of points in $\mathcal{F}$. Thus, the first few principal components of $\mathcal{F}$ approximate $\mathbf{x}$ well. The kernel matrix $\mathbf{K}_{ij} = \mathcal{K}(\mathbf{p}_i, \mathbf{p}_j)$ is the sample covariance of features and its eigenvectors capture these components, then the deviation is evaluated via $h(\mathbf{K})\mathbf{x} = \mathbf{K}^{-1}\mathbf{x}$ as in (2) for the node side [14]. This transformation gets a larger effect when the graph signal has a significant projection in a direction that the feature set is less spread, *i.e.*, a small eigenvalue of $\mathbf{K}$. Alternatively, $h(\mathbf{K})$ can be viewed as a filter based on $\mathcal{P}$ that amplifies the signal in atypical directions that the feature set is spread.

The fitness of a graph signal $\mathbf{x}$ to the underlying graph $\mathcal{G}$ and the node-side information $\mathcal{P}$ is measured via applying a filter in terms of $\mathbf{L}$ and $\mathbf{K}$, then comparing the two signals. We use the inner product as the similarity function as follows:

$$s(\mathbf{x}, \hat{\mathbf{x}}) = \langle \mathbf{x}, \hat{\mathbf{x}} \rangle = \mathbf{x}^\top \hat{\mathbf{x}}, \tag{3}$$

where $\hat{\mathbf{x}}$ is the filtered (denoised) version of $\mathbf{x}$ such that:

$$\mathbf{x} - \hat{\mathbf{x}} = \alpha \mathbf{K}^{-1} \hat{\mathbf{x}} + \beta \mathbf{L} \hat{\mathbf{x}} \tag{4}$$

$$\Rightarrow \hat{\mathbf{x}} = \overbrace{(\mathbf{I} + \alpha \mathbf{K}^{-1} + \beta \mathbf{L})^{-1}}^{h(\mathbf{K},\mathbf{L})} \mathbf{x} \tag{5}$$

for some positive scalars α and β. Eq. (4) states that the difference between $\mathbf{x}$ and $\hat{\mathbf{x}}$ lies in the linear cone of $\mathbf{K}^{-1}\hat{\mathbf{x}}$ and $\mathbf{L}\hat{\mathbf{x}}$. The former means $\hat{\mathbf{x}}$ aligns more with the prior information, and the latter means $\hat{\mathbf{x}}$ is smoother on the underlying graph. Thus, the filter $h(\mathbf{K}, \mathbf{L})$ has a low-pass behavior on the combination of the $\mathcal{G}$ and $\mathcal{P}$. Specifically, it becomes a typical low-pass graph filter when $\alpha = 0$.

We propose to jointly cluster the graph signals and learn multiple graphs consistently with prior node-side information. Our KMGL algorithm expresses the problem as finding the partition sets $\{\mathcal{X}_k\}_{k=1}^K$ and the Laplacian matrices $\{\mathbf{L}_k\}_{k=1}^K$ as follows:

$$\begin{aligned} \max_{\{\mathcal{X}_k, \mathbf{L}_k \in \mathcal{L}\}_{k=1}^K, \{\hat{\mathbf{x}}_i\}_{i=1}^m} \quad & \sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{X}_k} \mathbf{x}^\top \hat{\mathbf{x}} - \gamma \|\mathbf{L}_k\|_F^2 \\ \text{s.t.} \quad & \mathbf{x} - \hat{\mathbf{x}} = \alpha \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \mathbf{L}_k \hat{\mathbf{x}};\ \forall \mathbf{x} \in \mathcal{X}_k,\ \forall k \\ & \mathrm{tr}(\mathbf{L}_k) = n;\ \forall k \end{aligned} \tag{6}$$

where $\mathcal{L}$ is the set of valid graph Laplacians

$$\mathcal{L} = \{\mathbf{L} \in \mathbb{R}^{n \times n} \mid \mathbf{L} = \mathbf{L}^\top, \mathbf{L}\mathbf{1} = \mathbf{0}, \mathbf{L}_{ij} \le 0\ \forall i \ne j\}. \tag{7}$$

The first term in (6) promotes the similarity of the graph signal and its filtered version. This term helps in assigning graph signals that are more similar to the filtered version w.r.t. the underlying graph. The hyperparameters α and β control the trade-off between matching the side information and the smoothness of graph signals, respectively. The second term in (6) regularizes the graphs to have a smaller Frobinius norm. Combined with the second constraint, they affect the sparsity of the learned graphs and avoid trivial solutions [4], [14]. Graphs are sparser as $\gamma \in \mathbb{R}_+$ gets larger.

A. Algorithm

The problem formulated in (6) is NP-hard. This is because the selection of partitions affects the optimal graphs and, consequently, the objective function. To avoid solving the problem for every possible partitioning, an iterative solution similar to K-means [18] is proposed that increases the objective at each step. The algorithm first partitions the dataset randomly and then iterates between two steps: i) learning a graph for each cluster, and ii) reassigning the graph signals. This is repeated until the partitions remain the same.

1) Fixing the cluster assignments and learning the underlying graphs and filtered signals: Firstly, for the kth cluster, given the initial (and possibly noisy) graph signals $\{\mathbf{x} \in \mathcal{X}_k\}$ and by fixing their assignments, we solve the following optimization problem for learning their associated filtered versions and also the kth graph Laplacian:

$$\{\mathbf{L}_k, \hat{\mathcal{X}}_k\} = \underset{\mathbf{L}_k \in \mathcal{L}, \hat{\mathbf{x}} \in \hat{\mathcal{X}}_k}{\operatorname{argmax}} \sum_{\mathbf{x} \in \mathcal{X}_k} \mathbf{x}^\top \hat{\mathbf{x}} - \gamma \|\mathbf{L}_k\|_F^2$$
$$\text{s.t. } \forall \mathbf{x} \in \mathcal{X}_k : \mathbf{x} - \hat{\mathbf{x}} = \alpha \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \mathbf{L}_k \hat{\mathbf{x}}, \;\; \operatorname{tr}(\mathbf{L}_k) = n, \quad (8)$$

where the set $\hat{\mathcal{X}}_k$ contains the filtered versions of graph signals for $\mathcal{X}_k$. Although it is less obvious, the above problem can be formulated more similarly to the typical GL objective functions [4], [14] as follows:

Theorem 1. *The maximization problem in (8) is equivalent to a joint kernel ridge regression from set $\mathcal{P}$ to $\mathcal{X}_k$ and a GL problem.*

Proof. We show that our problem is equivalent to:

$$\begin{aligned} \min_{\mathbf{L}_k \in \mathcal{L}, \mathbf{c} \in \mathcal{C}_k} \sum_{\mathbf{x} \in \mathcal{X}_k} & \|\mathbf{x} - \mathbf{K}_k \mathbf{c}\|_2^2 + \alpha \mathbf{c}^\top \mathbf{K}_k \mathbf{c} \\ & + \beta \mathbf{c}^\top \mathbf{K}_k \mathbf{L}_k \mathbf{K}_k \mathbf{c} + \gamma \|\mathbf{L}_k\|_F^2 \\ \text{s.t.} \quad & \operatorname{tr}(\mathbf{L}_k) = n, \end{aligned} \quad (9)$$

where for any $\mathbf{c} \in \mathcal{C}_k$ there exist only one filtered signal $\hat{\mathbf{x}} \in \hat{\mathcal{X}}_k$ such that $\hat{\mathbf{x}} = \mathbf{K}_k \mathbf{c}$ as proposed in [14]. We start by writing (9) in terms of the filtered signals $\{\hat{\mathbf{x}} \in \hat{\mathcal{X}}_k\}$. Note that since $\mathbf{K}_k$ is positive definite, $\mathbf{c}^\top \mathbf{K}_k \mathbf{c} = \hat{\mathbf{x}}^\top \mathbf{K}_k^{-1} \hat{\mathbf{x}}$ and we have:

$$\begin{aligned} \min_{\mathbf{L}_k \in \mathcal{L}, \hat{\mathcal{X}}_k} \sum_{\mathbf{x} \in \mathcal{X}_k} & \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \alpha \hat{\mathbf{x}}^\top \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \hat{\mathbf{x}}^\top \mathbf{L}_k \hat{\mathbf{x}} + \gamma \|\mathbf{L}_k\|_F^2 \\ \text{s.t.} \quad & \operatorname{tr}(\mathbf{L}_k) = n. \end{aligned} \quad (10)$$

The above problem is separable in each $\hat{\mathbf{x}}$ and we start by minimizing over them as follows:

$$\hat{\mathbf{x}} = \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \underbrace{\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \alpha \hat{\mathbf{x}}^\top \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \hat{\mathbf{x}}^\top \mathbf{L}_k \hat{\mathbf{x}}}_{f(\hat{\mathbf{x}})} \quad (11)$$

that is convex and differentiable with a gradient:

$$\nabla_{\hat{\mathbf{x}}} f(\hat{\mathbf{x}}) = -(\mathbf{x} - \hat{\mathbf{x}}) + \alpha \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \mathbf{L}_k \hat{\mathbf{x}}. \quad (12)$$

Putting the gradient in (12) to zero results in (4). Then, by substituting (4) into (11), one can write:

$$\begin{aligned} f(\hat{\mathbf{x}}) &= \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \hat{\mathbf{x}}^\top \left(\alpha \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \mathbf{L}_k \hat{\mathbf{x}} \right) \\ &= ||\mathbf{x} - \hat{\mathbf{x}}||_2^2 + \hat{\mathbf{x}}^\top (\mathbf{x} - \hat{\mathbf{x}}) \\ &= \mathbf{x}^\top (\mathbf{x} - \hat{\mathbf{x}}) = ||\mathbf{x}||_2^2 - \mathbf{x}^\top \hat{\mathbf{x}}. \end{aligned}$$

Algorithm 1 : KMGL

```
Input: Graph signals $\mathcal{X}$, number of clusters $K$, Kernel matrices $\{\mathbf{K}_k\}_{k=1}^K$, hyperparameters $\alpha, \beta, \gamma$, tolerance $\epsilon$
Output: Partition set $\{\mathcal{X}_k\}_{k=1}^K$, graph Laplacians $\{\mathbf{L}_k\}_{k=1}^K$, filtered signals $\{\forall\ \mathbf{x} \in \mathcal{X} : \hat{\mathbf{x}}\}$
Initialization: Randomly partition $\mathcal{X}$ into $K$ clusters.
repeat
    for $k = 1 : K$ do                ▷ GL and filtering (denoising)
        repeat
            Filter every $\mathbf{x} \in \mathcal{X}_k$ with $h(\mathbf{K}_k, \mathbf{L}_k)$ in (5)
            Learn the graph Laplacian $\mathbf{L}_k$ via Eq. (14)
        until $\mathbf{L}_k$ converges w.r.t. a tolerance $\epsilon$
    end for
    for $\mathbf{x} \in \mathcal{X}$ do                ▷ Refining the clusters
        Update the cluster of $\mathbf{x}$ via Eq. (15)
    end for
until clusters are unchanged
```

Next, objective (10) turns to:

$$\begin{aligned} \min_{\mathbf{L}_k \in \mathcal{L}, \hat{\mathcal{X}}_k} \sum_{\mathbf{x} \in \mathcal{X}_k} & \|\mathbf{x}\|_2^2 - \mathbf{x}^\top \hat{\mathbf{x}} + \gamma \|\mathbf{L}_k\|_F^2 \\ \text{s.t.} \quad & \forall \mathbf{x} \in \mathcal{X}_k : \mathbf{x} - \hat{\mathbf{x}} = \alpha \mathbf{K}_k^{-1} \hat{\mathbf{x}} + \beta \mathbf{L}_k \hat{\mathbf{x}} \\ & \operatorname{tr}(\mathbf{L}_k) = n. \end{aligned} \quad (13)$$

The term $\|\mathbf{x}\|_2^2$ has no bearing on the minimization, and a sign change in objective results in the maximization (8). □

Based on Theorem 1 and the approach in [14], we solve (8) by applying a BCD scheme on objective (10) that iteratively filters the graph signals by Eq. (5) and then solves for:

$$\begin{aligned} \mathbf{L}_k = \underset{\mathbf{L}_k \in \mathcal{L}}{\operatorname{argmin}} & \sum_{\forall \hat{\mathbf{x}} \in \hat{\mathcal{X}}_k} \beta \hat{\mathbf{x}}^\top \mathbf{L}_k \hat{\mathbf{x}} + \gamma \|\mathbf{L}_k\|_F^2 \\ \text{s.t.} \quad & \operatorname{tr}(\mathbf{L}_k) = n, \end{aligned} \quad (14)$$

which is a GL problem in the typical Laplacian quadratic form [4], [9] and can be solved by convex optimization techniques [22].

2) Assigning the graph signals to their associated clusters by fixing the underlying graphs and filtered signals.: In the second step of the algorithm, we refine the partitions by fixing the graphs and assigning each graph signal to the most compatible cluster. Let $\hat{\mathcal{I}}(\mathbf{x}) = \{\hat{\mathbf{x}}_k \mid \hat{\mathbf{x}}_k = h(\mathbf{K}_k, \mathbf{L}_k)\mathbf{x}, k = 1, \ldots, K\}$ be the set of filtered graph signals of $\mathbf{x}$ over all graphs. We specify the assignment of $\mathbf{x}$, *i.e.*, $i(\mathbf{x})$, as follows:

$$i(\mathbf{x}) = \underset{k:\ \hat{\mathbf{x}}_k \in \hat{\mathcal{I}}(\mathbf{x})}{\operatorname{argmax}} \ \mathbf{x}^\top \hat{\mathbf{x}}_k. \quad (15)$$

Then, the partitions are refined such that $\mathcal{X}_k = \{\mathbf{x} \in \mathcal{X} \mid i(\mathbf{x}) = k\}$ for $k = 1, \ldots, K$.

These two steps continue alternatively until getting convergence, *e.g.* clusters are unchanged. The proposed KMGL algorithm is summarized in Algorithm 1.

The next theorem states some properties about the convergence of the proposed KMGL method.

Theorem 2. *The KMGL algorithm converges in a finite number of iterations.*

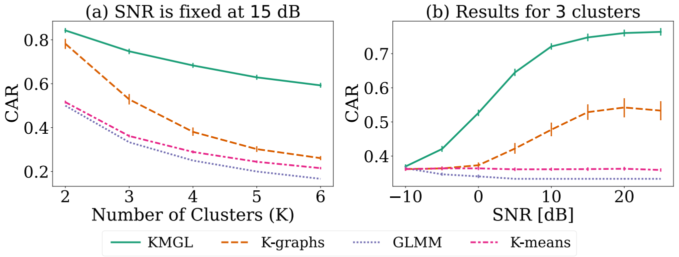

Fig. 1: Clustering performance of KMGL compared to K-graphs, GLMM, and K-means based on CAR when (a) K and (b) SNR increases. $\alpha = \beta = 10^{-2}$, $\gamma = 10^{-4}$.

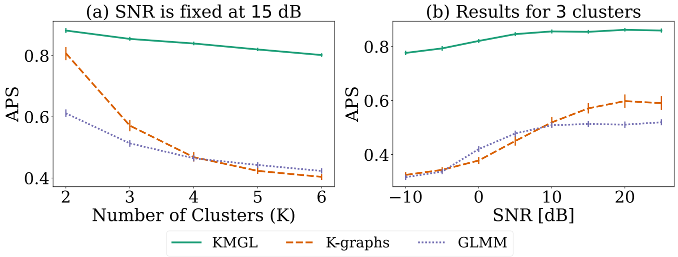

Fig. 2: GL performance of KMGL compared to K-graphs, and GLMM based on APS when (a) K and (b) the SNR increases. $\alpha = \beta = 10^{-2}$, $\gamma = 10^{-4}$.

Proof. For each algorithm step, the objective function in (6) is non-decreasing. This is because in the first step when the graphs are updated via Eq. (8), the maximization is over the same terms, and thus the new graphs will not decrease the objective. Precisely, the optimization problem (8) is biconvex [14], and therefore, utilizing BCD reaches unique solutions for each subproblem which guarantees to reach a stationary point [19]. In the second step, when the clusters are refined, we directly increase the term $\mathbf{x}^\top \hat{\mathbf{x}}$ for each graph signal, or it remains the same. Since $\|\mathbf{L}_k\|_F^2$ is fixed, the objective is also non-decreasing here. Lastly, there are finite assignments of m graph signals to K clusters, and consequently, the algorithm has to converge [18]. □

IV. Experiments

In this section, the performance of the KMGL algorithm on numerical data is evaluated and compared to the GLMM [12], and K-graphs [15] methods. The K-means algorithm is also added as a baseline to represent a model without knowledge of the underlying graphs. We draw random Erdos–Renyi graphs of n nodes with a (binary) connection probability of $p((v_i, v_j) \in \mathcal{E}) = 0.3$ for $i \neq j$. Edge weights are normalized such that the sum of weights is n. Similar to [14], graph signals of the kth cluster are generated according to $\forall \mathbf{x} \in \mathcal{X}_k : \mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_k + \sigma_\epsilon \mathbf{I})$, where $\mathbf{K}_k = (\mathbf{I} + \eta \mathbf{L}_k)^{-1}$ is the kernel matrix corresponding to the kth cluster and σ_ϵ is the noise-level. This common choice of kernel leads to globally smooth signals [14]. We select $\eta = 10$ for the experiments by performing a grid search on the training data. The models are examined on different σ_ϵ and number of clusters K. We select the noise level such that the Signal-to-Noise Ratio

$$\mathrm{SNR} = 10 \log_{10} \left(\frac{1}{K n \sigma_\epsilon^2} \sum_{c=1}^{K} \mathrm{tr}(\mathbf{K}_c) \right) \tag{16}$$

is varied uniformly.

Models are evaluated based on their clustering performance and the quality of learned graphs. The former is measured via Clustering Accuracy Ratio (CAR), which finds the best map between the cluster indices of samples and the partitions, then measures the number of correctly clustered signals to their total number [15]. The quality of learned graphs is evaluated by Average Precision Score (APS), where the ability of the model to detect the presence of edges is considered [14].

The datasets have $m = 500$ graph signals, sampled equally from $K \in \{2, 3, 4, 5\}$ different graphs with $n = 20$ nodes. We perform preliminary tests for each model to tune the hyperparameters and then keep them intact through the experiments. Furthermore, we restrict our model so that $\alpha = \beta = 10^{-2}$ equally prioritizes smoothness and side information. In the results, each data point consists of 50 independent realizations. The compared models are applied 10 times for each realization and are evaluated on their best try, determined via their objective function, while the proposed algorithm is applied only once.

Fig. 1 displays the effective clustering performance of the KMGL algorithm. Fig. 1(a) shows that KMGL is more robust to a high number of clusters, and Fig. 1(b) shows KMGL's robustness against noise compared to the previous methods. Fig. 2 shows the ability of the compared models to recover and learn the graphs, even in a high number of clusters in Fig. 2(a) and a high amount of noise in Fig. 2(b). Similar to Fig. 1, KMGL outperforms the compared models both in Fig. 2(a) when the number of clusters is increased and Fig. 2(b) when the noise-rejection behavior is studied. It is worth mentioning that the ability of the model to recover the graphs even in high noise levels is an effective advantage of exploiting kernel metrics that was also seen in [14]. This further shows the benefits of incorporating side information.

V. Conclusion

In this letter, we introduced the KMGL algorithm, which incorporates node-side information in clustering graph signals and learning multiple graphs. We used kernels to represent this node-side information and built a framework that uses filters to model the relationship between the data and the underlying graphs. We solved the optimization problem associated with KMGL using the BCD method, and we proved its convergence. Our experiments have shown that KMGL outperforms existing methods, especially when dealing with high levels of noise and a large number of clusters. The theoretical guarantees and experiments underscore the potential value of KMGL in real-world applications.

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, 2013.

[2] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[3] G. Leus, A. G. Marques, J. M. Moura, A. Ortega, and D. I. Shuman, "Graph signal processing: History, development, impact, and outlook," *IEEE Signal Processing Magazine*, vol. 40, no. 4, pp. 49–60, 2023.

[4] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[5] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 44–63, 2019.

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

[7] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Identifying the topology of undirected networks from diffused non-stationary graph signals," *IEEE Open Journal of Signal Processing*, vol. 2, pp. 171–189, 2021.

[8] A. Einizade, S. H. Sardouie, and M. B. Shamsollahi, "Simultaneous graph learning and blind separation of graph signal sources," *IEEE Signal Processing Letters*, vol. 28, pp. 1495–1499, 2021.

[9] V. Kalofolias, "How to learn a graph from smooth signals," in *Artificial intelligence and statistics*, pp. 920–929, PMLR, 2016.

[10] J. Guo, S. Moses, and Z. Wang, "Graph learning from signals with smoothness superimposed by regressors," *IEEE Signal Processing Letters*, 2023.

[11] G. Fatima, A. Arora, P. Babu, and P. Stoica, "Learning sparse graphs via majorization-minimization for smooth node signals," *IEEE Signal Processing Letters*, vol. 29, pp. 1022–1026, 2022.

[12] H. P. Maretic and P. Frossard, "Graph laplacian mixture model," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 261–270, 2020.

[13] S. S. Saboksayr and G. Mateos, "Accelerated graph learning from smooth signals," *IEEE Signal Processing Letters*, vol. 28, pp. 2192–2196, 2021.

[14] X. Pu, S. L. Chau, X. Dong, and D. Sejdinovic, "Kernel-based graph learning from smooth signals: A functional viewpoint," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 192–207, 2021.

[15] H. Araghi, M. Sabbaqi, and M. Babaie-Zadeh, "k-graphs: An algorithm for graph signal clustering and multiple graph learning," *IEEE Signal Processing Letters*, vol. 26, no. 10, pp. 1486–1490, 2019.

[16] A. Karaaslanli and S. Aviyente, "Simultaneous graph signal clustering and graph learning," in *International Conference on Machine Learning*, pp. 10762–10772, PMLR, 2022.

[17] Y. Yuan, X. Yang, K. Guo, T. Q. Quek, *et al.*, "Gracge: Graph signal clustering and multiple graph estimation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 2015–2030, 2022.

[18] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Information Sciences*, 2022.

[19] A. Beck and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.

[20] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," 2008.

[21] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[22] S. Diamond and S. Boyd, "CVXPY: A python-embedded modeling language for convex optimization," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2909–2913, 2016.

[23] S. K. Narang, A. Gadde, E. Sanou, and A. Ortega, "Localized iterative methods for interpolation in graph structured data," in *2013 IEEE Global Conference on Signal and Information Processing*, pp. 491–494, IEEE, 2013.

Appendix

A. Clustering and Learning with Missing Data

In this section, we extend the KMGL algorithm to support partially observed graph signals. Specifically, we change how each graph signal is filtered by first recovering its missing values. To this end, we adopt an iterative approach for graph signal reconstruction. The asymptotic behavior of the resulting algorithm is studied and we conduct numerical experiments to measure its effectiveness.

Let $\mathcal{B}$ denote the observed subspace of the graph signal $\mathbf{x}$. Then, the downsampling operator $\mathbf{J} : \mathbb{R}^n \to \mathcal{B}$ maps the partially observed signal to this space. Moreover, $\mathbf{M} = \mathbf{J}^\top \mathbf{J}$ represents the downsampling and then upsampling operation where $\mathbf{M}$ is a diagonal masking matrix with $M_{ii} = 1$ if we observe the ith component of $\mathbf{x}$ and zero otherwise. $\mathbf{Mx}$ interpolates the missing values of $\mathbf{x}$ with zeroes, however, this incorporates neither the graph structure nor the side information.

To extend the KMGL algorithm, we need to revisit how each graph signal $\mathbf{x}$ relates to its filtering $\hat{\mathbf{x}}$. Previously, the low-pass filter $\mathbf{S} = h(\mathbf{K}, \mathbf{L})$ related the two in (5). Since $\mathbf{x}$ is partially observed,

$$\begin{aligned}\mathbf{x}^1 &= \mathbf{Mx}\\ \hat{\mathbf{x}}^t &= \mathbf{Sx}^t\\ \mathbf{x}^{t+1} &= \hat{\mathbf{x}}^t + \mathbf{M}(\mathbf{x}^1 - \hat{\mathbf{x}}^t)\end{aligned} \tag{17}$$

iteratively recovers its missing values as suggested in [23]. The first line initially interpolates $\mathbf{x}$ with zero. The second line applies a low-pass filter to update the missing nodes based on the others. The structure of the graph and the side information dictate how missing values relate to others. The last line ensures that $\mathbf{x}^t$ remains unchanged in the observed nodes. Lastly, the similarity of the reconstructed graph signal $\mathbf{x}^t$ with its low-passed filtering $\hat{\mathbf{x}}^t$ is compared in $\mathcal{B}$:

$$s(\mathbf{x}, \hat{\mathbf{x}}^t) = (\mathbf{Jx}^t)^\top(\mathbf{J}\hat{\mathbf{x}}^t) = \mathbf{x}^\top \mathbf{M}\hat{\mathbf{x}}^t. \tag{18}$$

Theorem 3. *As the number of iterations t increases, $\hat{\mathbf{x}}^t$ has the asymptotic solution of $(\mathbf{M} + \alpha\mathbf{K}^{-1} + \beta\mathbf{L})^{-1}\mathbf{Mx} = \hat{\mathbf{x}}$.*

Proof. The iterations in (17) converge to a fixed point if $\mathbf{S}$ is a non-expansive operator [23]. Since $\alpha\mathbf{K}^{-1} + \beta\mathbf{L}$ is positive definite, all the eigenvalues of $\mathbf{S}$ have the absolute value of less than one. Consequently, $\mathbf{S}$ is non-expansive and $\hat{\mathbf{x}}_t$ converges. In the converged point $\hat{\mathbf{x}} = \hat{\mathbf{x}}^t = \hat{\mathbf{x}}^{t+1}$ we have

$$\begin{aligned}&\hat{\mathbf{x}} = \mathbf{S}(\hat{\mathbf{x}} + \mathbf{M}(\mathbf{x}_1 - \hat{\mathbf{x}})) = \mathbf{S}(\hat{\mathbf{x}} + \mathbf{Mx} - \mathbf{M}\hat{\mathbf{x}}) \Rightarrow\\ &\mathbf{M}(\mathbf{x} - \hat{\mathbf{x}}) = \mathbf{S}^{-1}\hat{\mathbf{x}} - \hat{\mathbf{x}} = \alpha\mathbf{K}^{-1}\hat{\mathbf{x}} + \beta\mathbf{L}\hat{\mathbf{x}} \Rightarrow\\ &(\mathbf{M} + \alpha\mathbf{K}^{-1} + \beta\mathbf{L})^{-1}\mathbf{Mx} = \hat{\mathbf{x}}\end{aligned} \tag{19}$$

which completes the proof. □

We remark that (19) also results from solving the following convex problem as shown in [14]:

$$\min_{\hat{\mathbf{x}}} \|\mathbf{M}(\mathbf{x} - \hat{\mathbf{x}})\|_2^2 + \alpha\hat{\mathbf{x}}^\top\mathbf{K}^{-1}\hat{\mathbf{x}} + \beta\hat{\mathbf{x}}^\top\mathbf{L}\hat{\mathbf{x}}. \tag{20}$$

To summarize, the Alg. 1 is generalized to support partially observed graph signals as follows. In line 5, every signal is now filtered via (19), where $\mathbf{M}$ is the diagonal masking matrix associated with $\mathbf{x}$. The Laplacian and the kernel also relate to each signal's cluster. In line 10, the signals are reassigned based on:

$$i(\mathbf{x}) = \underset{k:\hat{\mathbf{x}}_k \in \hat{\mathcal{I}}(\mathbf{x})}{\operatorname{argmax}} \quad \mathbf{x}^\top\mathbf{M}\hat{\mathbf{x}}_k \tag{21}$$

where

$$\hat{\mathcal{I}} = \{\hat{\mathbf{x}}_k \mid (\mathbf{M} + \alpha\mathbf{K}_k^{-1} + \beta\mathbf{L}_k)^{-1}\mathbf{Mx} = \hat{\mathbf{x}}_k, 1 \le k \le |\mathcal{C}|\}$$

is similar to the previous set but with the filters in (19).

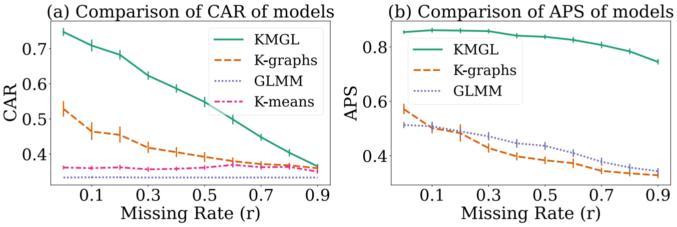

Fig. 3: Performance of KMGL algorithm on partially observed signals compared to K-graphs, GLMM, and K-means in terms of (a) clustering and (b) learning graphs. SNR=15 and 3 clusters.

The rest of this section discusses the validity of the proposed algorithm in numerical experiments. Each entry of each graph signal has a $(1 - r)$ probability of missing where r is the missing rate. Hence, the element of the diagonal mask matrix is generated via $M_{ii} \sim \text{Bernoulli}(1 - r)$. For the compared models, the missing values are set to their statistical expectation, *i.e.*, zero, as this is a natural and unbiased setting in practice.

The performance of models is evaluated based on their Clustering Accuracy Ratio (CAR) and Average Precision Score (APS) of the recovered graphs. The metrics are plotted against the missing rate as it changes uniformly between zero and one. Fig. 3 summarizes the results and shows the higher robustness of KMGL. Each data point in the figures is the average of 20 independent realizations with different graphs and data.