Signal Processing Over Multilayer Graphs: Theoretical Foundations and Practical Applications

Songyang Zhang, Qinwen Deng, Member, IEEE, and Zhi Ding, Fellow, IEEE

Abstract—Signal processing over single-layer graphs has become a mainstream tool owing to its power in revealing obscure underlying structures within data signals. However, many real-life data sets and systems, including those in Internet of Things (IoT), are characterized by more complex interactions among distinct entities, which may represent multilevel interactions that are harder to be captured with a single-layer graph, and can be better characterized by multilayers graph connections. Such multilayer or multilevel data structures can be more naturally modeled by high-dimensional multilayer graphs (MLGs). To generalize traditional graph signal processing (GSP) over MLGs for analyzing multilevel signal features and their interactions, this work proposes a tensor-based framework of MLG signal processing (M-GSP). Specifically, we introduce core concepts of M-GSP and study properties of MLG spectral space, followed by fundamentals of MLG-based filter design. To illustrate novel aspects of M-GSP, we further explore its link with traditional signal processing and GSP. We provide example applications to demonstrate the efficacy and benefits of applying MLGs and M-GSP in practical scenarios.

Index Terms—Data analysis, graph signal processing (GSP), multilayer graph (MLG), tensor.

I. INTRODUCTION

Geometric signal processing tools have found broad applications in data analysis to uncover obscure or hidden structures from complex data sets [1]. Various data sources, such as social networks, Internet of Things (IoT) intelligence, traffic flows and biological images, often feature complex structures that pose challenges to traditional signal processing tools. Recently, graph signal processing (GSP) emerges as an effective tool over the graph signal representation [2]. For a signal with \( N \) samples, a graph of \( N \) nodes can be formed to model their underlying interactions [3]. In GSP, a graph Fourier space is also defined from the spectral space of the representing matrix (adjacency/Laplacian) for signal processing tasks [4], such as denoising [5], resampling [6], and classification [7]. Generalization of the more traditional GSP includes signal processing over hypergraphs [8] and simplicial complexes [9], which are suitable to model high-degree multilateral node relationships.

Traditional GSP tools generally describe signals as graph nodes connected by one type of edges. However, real-life systems and data sets may feature multifacet interactions [10]. For example, in a video data set modeled by spatial–temporal graph shown in Fig. 1(a), the nodes may exhibit different types of spatial connections at different temporal steps. It is harder for single-layer graphs to model such multifacet connections. To model multiple layers of signal connectivity, we explore a high-dimensional graph representation known as multilayer graphs (MLGs) [11].

MLG, also named as multilayer network, is a geometric model containing correlated layers with different structures and physical meanings, unlike traditional single-layer graphs [10]. A typical example is smart grid consisting of two layers shown as Fig. 1(b): 1) the power grid and 2) the computation network. These two layers have different physical connectivity and rules [12]. Still, signal interactions across the multiple layers in MLG can be strongly correlated. Thus, separate representations by multiple single layer graphs may fail to capture such characteristics. Consider a network consisting of a physical power layer and a cyber layer, the failure of one layer could trigger the failure of the other [13]. One example was the power line damage caused by a storm on September 28, 2003. Not only did it lead to the failure of several power stations, but also disrupted communications as a result of power station breakdowns that eventually affected 56 million people in Europe [14].

The complexity and multilevel interactions of MLG make the data reside on the irregular and high-dimensional structures, which do not directly lend themselves to standard GSP tools. For example, even though one can represent MLG by a supra-graph unfolding all the layers [15], traditional GSP would treat interlayer and intralayer interactions equivalently in one spectral space without differentiating the spectra of interlayer and intralayer signal correlations. Recently, there has been growing interest in developing advanced GSP tools to process such multilevel structures. In [16], a joint time-vertex Fourier transform (JFT) is defined to process spatial–temporal graphs by applying graph Fourier transform (GFT) and discrete Fourier transform (DFT). Although JFT can process time-varying data sets, it does not provide more general temporal (interlayer) connectivity in a generic MLG. For flexible interlayer structure, a two-way GFT, proposed in [17], defines different graph Fourier spaces for interlayer and intralayer connections, respectively. However, its spatial interactions all lie

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
within a single graph structure, thereby limiting the generalization of MLG. Expanding the works of [17], the tensor-based multiway GSP framework (MWGSP) of [18] relies on product graph. In MWGSP, separate factor graphs are constructed for each mode of a tensor-represented signal, and a joint spectrum combines factor graph spectra. However, both JFT and MWGSP use the same intralayer connections for all layers. They do not admit different layers with heterogeneous graph structures, thereby limiting the ability to represent a general MLG.

Another challenge in MLG signal processing (M-GSP) lies in the need for a suitable mathematical representation. Traditional methods start with connectivity matrices. For example, in [19], a supra-adjacency matrix is defined to represent all layers equivalently while ignoring the nature of layers. One can also represent each layer with an individual adjacency matrix [20]. However, such matrix-based representations mainly focus on the intralayer connections and lack representation for interlayer interactions. A more natural and general way may start with tensor representation [11], which is particularly attractive in handling complex MLG graph analysis.

Our goal is to generalize GSP for MLGs to model, analyze, and process signals based on the intralayer and interlayer signal interactions. To address the aforementioned challenges and to advance MLG processing, we present a novel tensor framework for M-GSP. We summarize the main contributions of this work as follows.

1) Leveraging tensor representation of MLG, we introduce M-GSP from a spatial perspective, in which MLG signals and shifting of signals are defined.
2) Taking a spectrum perspective, we introduce new concepts of spectral space and spectrum transform for MLG. For interpretability of spectral space, we analyze the resulting MLG spectral properties and their distinction from existing GSP tools.
3) We also present fundamentals of filter design in M-GSP, and suggest several practical applications based on the proposed framework, including those in IoT systems.
4) Our application examples, together with experimental results, demonstrate the strong potentials of M-GSP in analyzing complex multilayer systems and processing multilevel data in IoT intelligence and data analysis.

We organize the technical presentation as follows. Section II first summarizes preliminaries of traditional GSP and tensor analysis, before presenting representations of MLGs within M-GSP in Section III. We then introduce the fundamentals of M-GSP spatial and spectral analysis in Sections IV and V, respectively. We next discuss MLG filter design in Section VI. We develop the physical insights and spectrum interpretation of M-GSP concepts in Section VII. With the newly proposed M-GSP framework, we provide several example applications to demonstrate its potential in Section VIII, before concluding this article in Section IX.

II. PRELIMINARIES

A. Overview of Graph Signal Processing

Signal processing on graphs [1], [2], [3] studies signals that are discrete in some dimensions by representing the irregular signal structure using a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is a set of $N$ nodes, and $\mathcal{E}$ is the representing matrix (e.g., adjacency/Laplacian) describing the geometric structure of the graph $\mathcal{G}$. Graph signals are the attributes of nodes that underlie the graph structure. A graph signal can be written as vector $s = [s_1, s_2, \ldots, s_N]^T \in \mathbb{R}^N$ where the superscript $(\cdot)^T$ denotes matrix/vector transpose.

With a graph representation $\mathcal{F}$ and a signal vector $s$, the basic graph filtering (shifting) is defined via

$$ s' = \mathcal{F}s $$ \hspace{1cm} (1) 

The graph spectral space, also known as the graph Fourier space is defined based on the eigenspace of the representing matrix. Let the eigendecomposition of $\mathcal{F}$ be given by $\mathcal{F} = \mathcal{V}\Lambda\mathcal{V}^{-1}$, where $\mathcal{V}$ is the matrix with eigenvectors of $\mathcal{F}$ as columns, and diagonal matrix $\Lambda$ consists of the corresponding eigenvalues. The GFT is defined as

$$ \hat{s} = \mathcal{V}^{-1}s $$ \hspace{1cm} (2) 

whereas the inverse GFT is given by $s = \mathcal{V}\hat{s}$.

From definitions of GFT, other concepts, such as sampling theory [21], filter design [22], and frequency analysis [4], can be developed for signal processing and data analysis tasks.

B. Introduction of Tensor Basics

Before introducing the fundamentals of M-GSP, we first review some basics on tensors that are useful for MLG analysis. Tensors can be viewed as multidimensional arrays. The order of a tensor is the number of indices needed to label a component of that array [23]. For example, a third-order tensor has three indices. More specially, a scalar is a zeroth-order tensor; a vector is a first-order tensor; a matrix is a second-order tensor; and an $M$-dimensional array is an $M$th-order tensor. For convenience, we use bold letters to represent the
tensors excluding scalars, i.e., $A \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ represents an $N$th-order tensor with $I_d$ being the dimension of the $k$th order, and use $A_{i_1, \ldots, i_N}$ to represent the entry of $A$ at position $(i_1, i_2, \ldots, i_N)$ with $1 \leq i_k \leq I_k$ in this work. If $A_f$ has a subscript $f$, we use $[A_f]_{i_1, \ldots, i_N}$ to denote its entries.

We now start with some useful definitions and tensor operations for the M-GSP framework [23].

1) **Super-Diagonal Tensor**: An $N$th-order tensor $A \in \mathbb{R}^{I_1 \times I_2 \times \cdots I_N}$ is super-diagonal if its entries $A_{i_1, i_2, \ldots, i_N} \neq 0$ only for $i_1 = i_2 = \cdots = i_N$.

2) **Symmetric Tensor**: A tensor is super-symmetric if its elements remain constant under index permutation. For example, a third-order tensor $A \in \mathbb{R}^{I \times I \times I}$ is super-symmetric if $A_{ijk} = A_{kij} = A_{ijk} = A_{jki}$. In addition, tensors can be partially symmetric in two or more modes as well. For example, a third-order tensor $A \in \mathbb{R}^{I \times I \times J}$ is symmetric in the order one and two if $A_{ijk} = A_{ikj}$, for $1 \leq i, j \leq I$ and $1 \leq k \leq J$.

3) **Tensor Outer Product**: The tensor outer product between a $P$th-order tensor $U \in \mathbb{R}^{I_1 \times \cdots \times I_P}$ with entries $U_{i_1, \ldots, i_P}$ and a $Q$th-order tensor $V \in \mathbb{R}^{J_1 \times \cdots \times J_Q}$ with entries $V_{j_1, \ldots, j_Q}$ is denoted by

$$W = U \odot V. \quad (3)$$

The result $W \in \mathbb{R}^{I_1 \times \cdots \times I_P \times J_1 \times \cdots \times J_Q}$ is a $(P+Q)$th-order tensor, whose entries are calculated by

$$W_{i_1, \ldots, i_P, j_1, \ldots, j_Q} = U_{i_1, \ldots, i_P} \cdot V_{j_1, \ldots, j_Q}. \quad (4)$$

The tensor outer product is useful to construct a higher order tensor from several lower order tensors.

4) **n-Mode Product**: The $n$-mode product between a tensor $U \in \mathbb{R}^{I_1 \times \cdots \times I_P}$ and a matrix $V \in \mathbb{R}^{I_n \times I_n}$ is denoted by

$$W = U \times_n V \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_P}. \quad (5)$$

Each element in $W$ is given by

$$W_{i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_P} = \sum_{i_n=1}^{I_n} U_{i_1, \ldots, i_p} V_{i_n, i_{n+1}, \ldots, i_P}. \quad (6)$$

Note that the $n$-mode product is a different operation from matrix product.

5) **Tensor Contraction**: In M-GSP, the contraction (inner product) between a forth order tensor $A \in \mathbb{R}^{M \times N \times M \times N}$ and a matrix $x \in \mathbb{R}^{M \times N}$ in the third and forth order is defined as

$$y = A \circ x \in \mathbb{R}^{M \times N} \quad (7)$$

where $y_{ai} = \sum_{\beta=1}^{M} \sum_{\alpha=1}^{N} A_{a_i \beta} x_{\alpha \beta}$.

In addition, the contraction between two fourth-order tensor $U, V \in \mathbb{R}^{M \times N \times M \times N}$ is defined as

$$W = U \odot V \in \mathbb{R}^{M \times N \times M \times N} \quad (8)$$

whose entries are

$$W_{aijp} = \sum_{\beta, \gamma} U_{a_i \beta} V_{\gamma j p}. \quad (9)$$

where integer $R$ denotes the rank, i.e., the lowest number of rank-one tensors in the decomposition. We illustrate CP decomposition for a third-order tensor in Fig. 2(a), which could be viewed as factorization of the tensor.

Another important decomposition is the Tucker decomposition, which is in the form of higher order PCA. More specifically, Tucker decomposition decomposes a tensor into a core tensor multiplied by a matrix along each mode [23]. Defining a core tensor $S = [S_{pq}] \in \mathbb{R}^{P \times Q \times R}$. Defining $A \in \mathbb{R}^{I \times P}, B \in \mathbb{R}^{J \times Q}, C \in \mathbb{R}^{K \times R}$, the Tucker decomposition of a third-order tensor $T \in \mathbb{R}^{I \times J \times K}$ is

$$T \approx S \times_1 A \times_2 B \times_3 C \quad (10)$$

where $A = [a_1 \cdots a_P], B = [b_1 \cdots b_Q], C = [c_1 \cdots c_K]$. The Tucker decomposition for a third-order tensor is illustrated in Fig. 2(b). Tucker decomposition reduces to CP decomposition if the core tensor is limited to be super-diagonal.

Other typical decompositions include higher order SVD (HOSVD) [25], orthogonal CP-decomposition [26], and Tensor-Train decomposition [27]. Interested readers are referred to the tutorial [23] for more details.

### III. REPRESENTATIONS OF MLG IN M-GSP

In this section, we introduce the fundamental representations of MLG within the M-GSP framework.

#### A. Multilayer Graphs

MLG, also referred to as multilayer network, is an important geometric model in complex systems [10]. In this work,
we refrain from using the “network” terminology because of its diverse meanings in various fields ranging from communication networking to deep learning. From here onward, unless otherwise specified, we shall use the less ambiguous term of MLG.

We first provide definitions of MLGs.

**Definition 1 (Multilayer Graph):** An MLG with $K$ nodes and $M$ layers is defined as $\mathcal{M} = \{\mathcal{V}, \mathcal{L}, \mathcal{F}\}$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_K\}$ is the set of nodes, $\mathcal{L} = \{l_1, l_2, \ldots, l_M\}$ denotes the set of layers with each layer $l_i = \{v_{i1}, \ldots, v_{in}\}$ being the subsets of $\mathcal{V}$, whereas $\mathcal{F}$ is the algebraic representation describing node interactions.

Note that, we mainly focus on the layer-disjoint MLG [10] where each node exists exactly in one layer, since layers denote different phenomena. For example, in a smart grid, a station with functions in both power grid and communication network, is usually modeled as two nodes in a two-layer graph for the network analysis [28].

In MLGs, edges connect nodes in the same layer (intralayer edges) or nodes of different layers (interlayer edges) [29]. There are two main types of MLGs: 1) multiplex graph and 2) interconnected graph [30]. In a multiplex graph, each layer has the same number of nodes, and each node only connects with its 1-to-1 matching counterparts in other layers to form interlayer connections. Typically, multiplex graphs characterize different types of interactions among the same (or a similar) set of physical entities. For example, the spatial-temporal connections among a set of nodes can be intuitively modeled as a multiplex graph [30]. In the interconnected graphs, each layer may have different numbers of nodes without a 1-to-1 counterpart. Their interlayer connections could be more flexible. Examples of a three-layer multiplex graph and a three-layer interconnected graph are shown in Fig. 3, where different colors represent different layers, solid lines represent intralayer connections, and dashed lines indicate interlayer connections.

**B. Algebraic Representation**

To capture the high-dimensional “multilayer” interactions between different nodes, we use tensor as algebraic representation of MLG for the proposed M-GSP framework [11].

1) **MLG With Same Number of Nodes in Each Layer:** To better interpret the tensor representation of an MLG, we start from a simpler type of MLG, in which each layer contains the same number of nodes. For an MLG $\mathcal{M} = \{\mathcal{V}, \mathcal{L}\}$ with $|\mathcal{L}| = M$ layers and $N$ nodes in each layer, i.e., $|l_i| = N$ for $1 \leq i \leq M$, it could be interpreted as embedding the interactions between a set of $N$ “entities” (not nodes) into a set of $M$ layers. The nodes in different layers can be viewed as the projections of the entities. For example, the video data sets could be modeled by the spatial connections between objects (entities) into different temporal frames (layers).

Mathematically, the process of embedding (projecting) entities can be viewed as a tensor product, and graph connections can be represented by tensors [11]. For convenience, we use Greek letters $\alpha, \beta, \ldots$ to indicate each layer and Latin letters $i, j, \ldots$ to indicate each interpretable “entity” with corresponding node in each layer. Given a set of entities $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, one can construct a vector $e_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ whose sole nonzero element is its $i$-th element (equal to 1) to characterize each entity $i$. Thus, interactions of two entities can be represented by a second-order tensor $A_{\alpha} = \sum_{i,j=1}^{N} a_{ij} e_i \circ e_j \in \mathbb{R}^{N \times N}$, where $a_{ij}$ is the intensity of the relationship between entity $i$ and entity $j$. Similarly, given a set of layers $\mathcal{L} = \{l_1, l_2, \ldots, l_M\}$, a vector $e_i \in \mathbb{R}^M$ can capture the properties of the layer $\alpha$, and the connectivity between two layers could be represented by $A_{L} = \sum_{\alpha,\beta=1}^{M} b_{\alpha\beta} e_\alpha \circ e_\beta \in \mathbb{R}^{M \times M}$. Following this approach, connectivity between the projected nodes of the entities in the layers can be represented by a fourth-order tensor

$$A = \sum_{\alpha,\beta=1}^{M} \sum_{i,j=1}^{N} w_{\alpha\beta ij} e_\alpha \circ e_\beta \circ e_i \circ e_j \in \mathbb{R}^{M \times N \times M \times N} \quad (11)$$

where $w_{\alpha\beta ij}$ is the weight of connection between the entity $i$’s projected node on layer $\alpha$ and the entity $j$’s projected node on layer $\beta$. More specifically, the fourth-order tensor becomes the adjacency tensor of the MLG, where each entry $A_{\alpha\beta ij} = w_{\alpha\beta ij}$ characterizes the edge between the entity $i$’s projected node on layer $\alpha$ and the entity $j$’s projected node on layer $\beta$. Thus, similar to the adjacency matrix whose 2-D entries indicate whether and how two nodes are pairwise connected by a simple edge in the normal graphs, we adopt an adjacency tensor $A \in \mathbb{R}^{M \times N \times M \times N}$ to represent the MLG with the same number of nodes in each layer as follows.

**Definition 2 (Adjacency Tensor):** An MLG $\mathcal{M}$, with $|\mathcal{L}| = M$ layers and $|l_i| = N$ nodes in each layer $i$, can be represented by a fourth-order adjacency tensor $A \in \mathbb{R}^{M \times N \times M \times N}$ defined as

$$A = (A_{\alpha\beta ij}), \quad 1 \leq \alpha, \beta \leq M, 1 \leq i, j \leq N. \quad (12)$$

Here, each entry $A_{\alpha\beta ij}$ of the adjacency tensor $A$ indicates the intensity of the edge between the entity $j$’s projected node on layer $\beta$ and entity $i$’s projected node on layer $\alpha$.

Clearly, for a single layer graph, $e_\alpha$ in (11) is a scalar 1 and the fourth-order tensor degenerates to the adjacency matrix of a normal graph. Similar to $A_{ij}$ in an adjacency matrix which indicates the direction from the node $v_j$ to $v_i$, $A_{\alpha\beta ij}$ also indicates the direction from the node $v_{ij}$ to the node $v_{ai}$ in an MLG. Note that, vectors $e_i$ and $e_\alpha$ are not eigen-vectors of the adjacency tensor. They are merely the vectors.
characterizing features of the entities and layers, respectively. We shall discuss the MLG-based spectral space in Section V.

Given an adjacency tensor, we can define the Laplacian tensor of the MLG similar to that in a single-layer graph. Denoting the degree (or multistrength) of the entity’s $i$’s projected node $v_{ai}$ on layer $\alpha$ as $d(v_{ai})$ which is a summation over weights of different natures (inter- and intra-layer edges), the degree tensor $D \in \mathbb{R}^{M \times N \times M \times N}$ is defined as a diagonal tensor with entries $D_{ai\alpha} = d(v_{ai})$ for $1 \leq i \leq N, 1 \leq \alpha \leq M$, whereas its other entries are zero. The Laplacian tensor can be defined as follows.

**Definition 3 (Laplacian Tensor):** An MLG $\mathcal{M}$, with $|\mathcal{L}| = M$ layers and $|\mathcal{I}| = N$ nodes in each layer $i$, can be represented by a fourth-order Laplacian tensor $L \in \mathbb{R}^{M \times N \times M \times N}$ defined as $L = D - A$, where $A$ is the adjacency tensor and $D$ is the degree tensor.

The Laplacian tensor can be useful to analyze propagation processes, such as diffusion or random walk [11]. Both adjacency and Laplacian tensors are important algebraic representations of the MLG depending on data sets and user objectives. For convenience, we use a tensor $F \in \mathbb{R}^{M \times N \times M \times N}$ as a general representation of a given MLG $\mathcal{M}$. As the adjacency tensor is more general, the representing tensor $F$ refers to the adjacency tensor in this article unless specified otherwise.

1) **Representation of General MLG:** Representing a general MLG with different number of nodes in each layer always remains a challenge if one aims to distinguish the interlayer and intralayer connection features. In JFT [16] and MWGSP [18], all layers must reside on the same underlying graph structure which restrict the number of nodes to be the same in each layer. Similarly, a reconstruction is also needed to represent a general MLG by the fourth-order tensor in M-GSP. Note that, although M-GSP also needs a reconstruction to represent a general MLG, we allow different layers with heterogeneous graph structures, which provides additional flexibility than JFT and MWGSP.

There are mainly two ways to reconstruct: 1) add isolated nodes to layers with fewer nodes to reach $N$ nodes [12] and set the augmented signals as zero and 2) aggregate several nodes into super-nodes for layers with $|I_{i}| > N$ [31] and merge the corresponding signals. Since isolated nodes do not interact with any other nodes, it does not change the topological structure of the original multilayer architecture in the sense of signal shifting while the corresponding spectral space could still be changed. The aggregation method depends on how efficiently we can aggregate redundant or similar nodes. Different methods can be applied depending on specific tasks. For example, if one wants to explore the cascading failure in a physical system, the method based on isolated nodes is more suitable. For the applications, such as video analysis, where pixels can be intuitively merged as superpixels, the aggregation method can be also practical.

In addition, although the fourth-order representing tensor can be viewed as the projection of several entities into different layers in (11), the entities and layers can be virtual and not necessarily physical to capture the underlying structures of the data sets. The information within the MLGs, together with the definitions of the underlying virtual entities and layers, should only depend on the structure of the MLGs. We will illustrate this further in Section VII-B.

2) **Flattening and Analysis**

In this part, we introduce the flattening of the MLG, which could simplify some operations in the tensor-based M-GSP. For an MLG $\mathcal{M} = (\mathcal{V}, \mathcal{L}, \mathcal{F})$ with $M$ layers and $N$ nodes in each layer, its fourth-order representing tensor $F \in \mathbb{R}^{M \times N \times M \times N}$ can be flattened into a second-order matrix to capture the overall edge weights. There are two main flattening schemes in the sense of entities and layers, respectively.

1) **Layerwise Flattening:** The representing tensor $F$ can be flattened into $F_{FL} \in \mathbb{R}^{MN \times MN}$ with each element

   \[ [F_{FL}]_{N(i-1)+i,N(j-1)+j} = F_{ai\beta}. \]

2) **Entitywise Flattening:** The representing tensor $F$ can be flattened into $F_{FN} \in \mathbb{R}^{NM \times NM}$ with each element

   \[ [F_{FN}]_{M(i-1)+\alpha,M(j-1)+\beta} = F_{ai\beta}. \]

These two flattening methods provide two ways to interpret the graph structure. In the first method, the flattened MLG has $M$ clusters with $N$ nodes in each cluster. The nodes in the same cluster have the same function (belong to the same layer). In the second method, the flattened graph has $N$ clusters with $M$ nodes in each cluster. Here, the nodes in the same cluster are from the same entity. Examples of the tensor flattening of a two-layer graph with three nodes in each layer are shown in Fig. 4. From the examples, we can see that the diagonal blocks in $\mathbb{R}^{N \times N}$ are the intralayer connections for each layer and other blocks describe the interlayer connections through layerwise flattening; and the diagonal block in $\mathbb{R}^{M \times M}$ describe the “intra-entity” connections and other elements represent the “inter-entity” connections in entitywise flattening. Although these two flattening schemes define the same MLG with a different indexing of vertices, they are still helpful to analyze the MLG spectral space.

IV. SPATIAL DEFINITIONS IN M-GSP

Based on the tensor representation, we now define signals and signal shifting over the MLGs. In GSP, each signal sample is the attribute of one node. Typically, a graph signal can be represented by an $N$-length vector for a graph with $N$ nodes. Recall that in traditional GSP [3], basic signal shifting is defined with the representing matrix as the shifting filter. Thus, in M-GSP, we can also define the signals and signal shifting based on the filter implementation.
In M-GSP, each signal sample is also related to one node in the MLG. Intuitively, if there are \( K = MN \) nodes, there are \( MN \) signal samples in total. Similar to GSP, we use the representing (adjacency/Laplacian) tensor \( F \in \mathbb{R}^{M \times N \times M \times N} \) as the basic MLG-filter. Since the input signal and the output signal of the MLG-filter should be consistent in the tensor size, we define a special form of M-GSP signals to work with the representing tensor as follows.

**Definition 4 (Signals Over MLGs):** For an MLG \( \mathcal{M} = \{V, \mathcal{L}, F\} \), with \(|\mathcal{L}| = M \) layers and \(|V| = N \) nodes in each layer \( i \), the definition of MLG signals is a second-order tensor

\[
s = (s_{ai}) \in \mathbb{R}^{M \times N}, \quad 1 \leq \alpha \leq M, 1 \leq i \leq N
\]  

where the entry \( s_{ai} \) is the signal sample in the projected node of entity \( i \) on layer \( \alpha \).

Note that, if the MLG degenerates to a single-layer graph with \( M = 1 \), the MLG signal becomes an \( N \)-length vector, which is similar to that in the traditional GSP. Similar to the representing tensor, the tensor signal \( \mathbf{s} \in \mathbb{R}^{M \times N} \) can also be flattened as a vector in \( \mathbb{R}^{MN} \).

1) **Layerwise Flattening:** \( s_L \in \mathbb{R}^{MN} \) whose entries are calculated as \([s_{L}]_{M(\alpha-1)+i} = s_{ai}\).

2) **Entitywise Flattening:** \( s_N \in \mathbb{R}^{NM} \) whose entries are calculated as \([s_N]_{N(\alpha-1)+j} = s_{ai}\).

Given the definitions of MLG signals and filters, we now introduce the definitions of signal shifting in M-GSP. In traditional GSP, the signal shifting is defined as product between signal vectors and representing matrix. Similarly, we define the shifting in the MLG based on the contraction (inner product) between the representing tensor and tensor signals.

**Definition 5 (Signal Shifting Over MLGs):** Given the representing tensor \( F \in \mathbb{R}^{M \times N \times M \times N} \) and the tensor signal \( \mathbf{s} \in \mathbb{R}^{M \times N} \) defined over an MLG \( \mathcal{M} \), the signal shifting is defined as product between the representing tensor and tensor signals. Since tensor decomposition is less stable when exploring the factorization of a specific order or when extracting the separate features in the asymmetric tensors, we will mainly focus on spectral properties of undirected MLGs in this section for simplicity and clarity of presentation. For directed MLG, we provide alternative spectrum definitions in the Appendix and will explore detailed analysis in future works.

V. MULTILAYER GRAPH SPECTRAL SPACE

In traditional GSP, graph spectral space is defined according to the eigenspace of the representing matrix [3]. Similarly, we define the MLG spectral space based on the decomposition of the representing tensor. More specifically, if \( F \) is the adjacency tensor, signals shift in directions of edges. To better illustrate the signal shifting based on the adjacency tensor, we use a two-layer directed network shown in Fig. 5 as an example. In this MLG, the original signal \( \mathbf{s} \) is defined as

\[
s = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 5 & 4 \end{bmatrix}
\]

and the adjacency tensor \( A \) is defined as

\[
A_{(1,:;1,:)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad A_{(2,:;2,:)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]

for each fiber. Then, the shifted signal is calculated as

\[
s' = \begin{bmatrix} 6 & 1 & 2 \\ 5 & 4 & 3 \end{bmatrix}
\]

From (19), we can see that the signal shift one step following the direction of the links.

Meanwhile, if \( F \) is the Laplacian tensor, (17) can be written as

\[
s'_{ai} = \sum_{\beta=1}^{M} \sum_{j=1}^{N} A_{ai\beta} (s_{ai} - s_{lj})
\]

which is the weighted average of difference with neighbors.

A. Joint Spectral Analysis in M-GSP

For an MLG \( \mathcal{M} = \{V, \mathcal{L}, F\} \) with \( M \) layers and \( N \) nodes, the eigentensor \( V \in \mathbb{R}^{M \times N} \) of the representing tensor \( F \) is defined in the tensor-based MLG theory [11] as

\[
F \odot V = \lambda V.
\]

More specifically, \( F \in \mathbb{R}^{M \times N \times M \times N} \) can be decomposed as

\[
F = \sum_{k=1}^{MN} \lambda_k V_k \odot V_k
\]

and the adjacency tensor \( A \in \mathbb{R}^{M \times N \times M \times N} \) can be decomposed as

\[
A = \sum_{\alpha=1}^{M} \sum_{i=1}^{N} \lambda_{\alpha i} V_{\alpha i} \odot V_{\alpha i}
\]
where \( \lambda_k \) is the eigenvalues and \( V_k \in \mathbb{R}^{M \times N} \) is the corresponding eigentensor. Note that \( V_{ai} \) just relabels the index of \( V_k \), and there is no specific order for \( V_{ai} \) here.

Similar to the traditional GSP where the graph Fourier space is defined by the eigenvectors of the representing matrix, we define the joint MLG Fourier space as follows.

**Definition 6 (Joint MLG Fourier Space):** For an MLG \( \mathcal{M} = \{V, L, F\} \) with \( M \) layers and \( N \) nodes, the MLG Fourier space is defined as the space consisting of all spectral tensors \( \{V_1, \ldots, V_{MN}\} \), which characterizes the joint features of entities and layers.

Recall that in GSP, the GFT is defined based on the inner product of \( V^{-1} \) and the signals \( s \) defined in (2). Similarly, we can define the M-GFT based on the spectral tensors of the representing tensor \( F \) to capture joint features of inter- and intra-layer interactions as follows.

**Definition 7 (Joint M-GFT):** Let \( U_F = (V_{ai}) \in \mathbb{R}^{M \times N \times M \times N} \) consist of spectral tensors of the representing tensor \( F \), where \( [U_F]_{ai}\beta] = [V_{ai}]_{\beta]}. \)

The joint M-GFT (M-JGFT) can be defined as the contraction between \( U_F \) and the tensor signal \( s \in \mathbb{R}^{M \times N} \), i.e.,

\[
\hat{s} = U_F \odot s.
\] (24)

Now, we show how to obtain the eigentensors. Implementing the flattening analysis, we have the following properties.

**Property 1:** The two types of flattened tensor in (13) and (14) lead to the same eigenvalues.

**Proof:** Suppose \((\lambda, x)\) is an eigenpair of \( A_{FL} \), i.e.,

\[
A_{FL} \cdot x = \lambda x.
\] (25)

Let \( x_{N(a-1)}+i = y_{M(i-1)+\alpha} \). Since

\[
A_{ai}\beta] = [A_{FL}]_{\beta]N(a-1)+i,N(\beta-1)+j} = [A_{FN}]_{\beta]M(i-1)+\alpha,M(j-1)+\beta}
\] (26)

we have

\[
A_{FN} \cdot y = \lambda y.
\] (27)

Thus, \( \lambda \) is also an eigenvalue of \( A_{FN} \).

**Property 2:** Given the eigenpair \((\lambda_{FL}, x)\) of the layerwise flattened tensors, the eigenpair \((\lambda, V)\) of the original representing tensor can be calculated as \( \lambda = \lambda_{FL} \), and \( V_{ai} = x_{N(a-1)}+i \). Similarly, given the eigenpair \((\lambda_{FN}, y)\) of the entitywise flattened tensor, the eigenpair \((\lambda, V)\) of the original representing tensor can be calculated as \( \lambda = \lambda_{FN} \), and \( V_{ai} = y_{M(i-1)+\alpha} \).

The Property 2 shows that we can calculate the Eigentensor from the flattened tensor to simplify the decomposition operations. Moreover, the M-JGFT is the bijection of GFT in the flattened MLG, with vertices indexed by both the layers and the entities. However, such M-JGFT analyzes the inter- and intra-layer connections jointly while ignoring the individual features of entities and layers. Next, we will show how to implement the orderwise frequency analysis in M-GSP based on tensor decomposition.

### B. Orderwise Spectral Analysis in M-GSP

In an undirected MLG, the representing tensor (adjacency/Laplacian) \( F \) is partially symmetric between orders one and three, and between orders two and four, respectively. Then, the representing tensor can be written with the consideration of the MLG structure under orthogonal CP-decomposition [26] as follows:

\[
F \approx \sum_{a=1}^{M} \sum_{i=1}^{N} \lambda_{ai} \circ f_a \circ e_i \circ e_i
\] (28)

\[
= \sum_{a=1}^{M} \sum_{i=1}^{N} \lambda_{ai} \tilde{V}_{ai} \circ \tilde{V}_{ai}
\] (29)

where \( f_a \in \mathbb{R}^M \) are orthonormal, \( e_i \in \mathbb{R}^N \) are orthonormal, and \( \tilde{V}_{ai} \circ \tilde{V}_{ai} \in \mathbb{R}^{M \times N} \).

The CP decomposition factorizes a tensor into a sum of component rank-one tensors, which describe the underlying features of each order. Although approximated algorithms are implemented to obtain the optimal decomposition, CP decomposition still achieves great success in real scenarios, such as feature extraction [32] and tensor-based PCA analysis [33].

A detailed discussion of tensor decomposition and its implementation in M-GSP is provided in Section VII-D. In (29), \( f_a \) and \( e_i \) capture the features of layers and entities, respectively, which can be interpreted as the subspaces of the MLG. More discussions about the frequency interpretation of orderwise M-GSP spectrum and connections to MWGSP spectrum are presented in Section VII-C.

Note that, if there is only one layer in the MLG, (29) reduces to the eigendecomposition of a normal single-layer graph, i.e.,

\[
F = \sum_{i=1}^{N} \lambda_i e_i \circ e_i
\] (30)

With the decomposed representing tensor in (29), the orderwise MLG spectrum is defined as follows.

**Definition 8 (Orderwise MLG Spectral Pair):** For an MLG \( \mathcal{M} = \{V, L, F\} \) with \( M \) layers and \( N \) nodes, the orderwise MLG spectral pairs are defined by \( \{\lambda_{ai}, f_a, e_i\} \), where \( \{f_1, \ldots, f_M\} \) and \( \{e_1, \ldots, e_N\} \) characterize features of layers and entities, respectively.

With the definition of orderwise MLG spectral pair, we now explore their properties. Considering \( \tilde{V}_{ai} = f_a \circ e_i \), we have the following property, which indicates the availability of a joint MLG analysis based on orderwise spectrum.

**Property 3:** The factor tensor \( \tilde{V}_{ai} \) of the representing tensor \( F \) is the approximated eigentensor of \( F \).

**Proof:** Suppose that \( \tilde{V}_{ai} \) is one factor tensor of \( F \) obtained from (29).

Let \( \delta[k] \) denote the Kronecker delta. Since \( f_a \) forms an orthonormal basis, then the inner product would satisfy

\[
<f_a, f_\beta> = \sum_k [f_a]_k \cdot [f_\beta]_k = \delta[\alpha - \beta].
\]

Similarly

\[
<e_i, e_j> = \delta[i - j].
\]

\[
F \circ \tilde{V}_{ai} = \sum_{\alpha=1}^{M} \sum_{k=1}^{N} F_{\alpha} \circ \tilde{V}_{ai} \circ \tilde{V}_{ai}
\] (30)
\[
\sum_{\sigma=1}^{M} \sum_{k=1}^{N} \sum_{\gamma=1}^{M} \sum_{\gamma'=1}^{N} \lambda_{\gamma \gamma'} f_{\gamma k} (e_{\gamma})_{k} \approx \sum_{\sigma=1}^{M} \sum_{k=1}^{N} \sum_{\gamma=1}^{M} \sum_{\gamma'=1}^{N} \lambda_{\gamma \gamma'} (f_{\gamma k} (e_{\gamma})_{k})_{\sigma k}. \tag{31}
\]

Then, we have
\[
\sum_{\sigma=1}^{M} \sum_{k=1}^{N} \left[ f_{\gamma} \right]_{\beta} (e_{\gamma})_{k} V_{\alpha i} \approx \sum_{\sigma=1}^{M} \sum_{k=1}^{N} \left[ f_{\gamma} \right]_{\beta} (e_{\gamma})_{k} \left[ V_{\alpha i} \right]_{\sigma k}. \tag{32}
\]

Thus,
\[
\sum_{\sigma=1}^{M} \sum_{k=1}^{N} \left[ f_{\gamma} \right]_{\beta} (e_{\gamma})_{k} V_{\alpha i} \approx \sum_{\sigma=1}^{M} \sum_{k=1}^{N} \left[ f_{\gamma} \right]_{\beta} (e_{\gamma})_{k} \left[ V_{\alpha i} \right]_{\sigma k}. \tag{33}
\]

Let \( T = [f_{1}, \ldots, f_{M}] \in \mathbb{R}^{M \times M} \) and \( E_{e} = [e_{1}, \ldots, e_{N}] \in \mathbb{R}^{N \times N} \). We then have
\[
\tilde{s}' = E_{e}^{T} s E_{e}. \tag{41}
\]

Then, we have
\[
\tilde{s}' = \sum_{j, \beta} \left[ f_{\beta} \right]_{\beta} \cdot (e_{\beta})_{j} \cdot s_{\beta j}. \tag{42}
\]

Clearly, the M-GFT can be obtained as
\[
\hat{s} = \tilde{s}' = E_{e}^{T} s E_{e}. \tag{43}
\]

Then, we have the following definition of M-GFT based on orderwise spectrum.

**Definition 9 (Orderwise M-GFT):** Given the spectral vectors \( T = [f_{1}, \ldots, f_{M}] \in \mathbb{R}^{M \times M} \) and \( E_{e} = [e_{1}, \ldots, e_{N}] \in \mathbb{R}^{N \times N} \), the layerwise M-GFT can be defined as
\[
\hat{s}_{L} = E_{e}^{T} \tilde{s} E_{e}. \tag{44}
\]

and the entitywise M-GFT can be defined as
\[
\hat{s}_{N} = s E_{e} E_{e}^{T} \in \mathbb{R}^{M \times N}. \tag{45}
\]

The general M-GFT based on orderwise spectrum is defined as
\[
\hat{s} = E_{e}^{T} s E_{e} \in \mathbb{R}^{M \times N}. \tag{46}
\]

Different from joint MLG Fourier space in Section V-A, the orderwise MLG spectrum provides an individual analysis on layers and entities separately, and a reliable approximated analysis on the underlying MLG structures jointly. To minimize confusion, we abbreviate joint M-GFT in (24) as M-JGFT. M-GFT refers to the orderwise transform in (46) in the remaining parts if there is no specification.

**C. MLG Singular Tensor Analysis**

In addition to the eigendecomposition, the singular value decomposition (SVD) is another important decomposition to factorize a matrix. In this part, we provide the HOSVD [25] of the representing tensor as an alternative definition of spectrum for the MLGs.

Given the MLG \( \mathcal{M} = (\mathcal{V}, \mathcal{L}, \mathcal{F}) \) with \( M \) layers and \( N \) nodes in each layer, its representing tensor \( F \in \mathbb{R}^{M \times N \times N \times N} \) can be decomposed via HOSVD as
\[
F = S_{1} \times \mathbb{U}^{(1)}_{1} \times \mathbb{U}^{(2)}_{2} \times \mathbb{U}^{(3)}_{3} \times \mathbb{U}^{(4)}_{4}, \tag{48}
\]

where \( \mathbb{U}^{(n)} = [U_{1}^{(n)} \quad U_{2}^{(n)} \quad \cdots \quad U_{n}^{(n)}] \) is a unitary \((I_{N} \times I_{N})\) matrix, with \( I_{1} = I_{3} = M \) and \( I_{2} = I_{4} = N \). \( S \) is a complex \((I_{1} \times I_{2} \times I_{3} \times I_{4})\)-tensor of which the subtensor \( S_{n} \) obtained by fixing \( n \)th index to \( \alpha \) have as follows.

1) \( (\mathbf{U}_{n} = \alpha, \mathbf{S}_{n} = \beta) > 0 \) where \( \alpha \neq \beta \).
2) \( \| \mathbf{S}_{n=1} \| \geq \| \mathbf{S}_{n=2} \| \geq \cdots \geq \| \mathbf{S}_{n=I_{n}} \| \geq 0 \).
The Frobenius-norms $\sigma_i^{(n)} = ||S_{n-i}||$ is the $n$-mode singular value, and $U^{(i)}$ are the corresponding $n$-mode singular vectors. For an undirected MLG, the representing tensor is symmetric for every 2-D combination. Thus, there are two modes of singular spectrum, i.e., $(\gamma_i, \ell_0)$ for modes 1, 3, and $(\sigma_i, e_i)$ for modes 2, 4. More specifically, $U^{(1)} = U^{(3)} = (f_\alpha)$ and $U^{(2)} = U^{(4)} = (e_i)$. Since the joint singular tensor captures the consistent information of entities and layers, it can be calculated as
\[
(\hat{\lambda}_{ai}, \hat{V}_{ai}) = (\gamma_a \cdot \sigma_i, f_\alpha \odot e_i).
\]
(49)

Note that the diagonal entries of $S$ are not the eigenvalues or frequency coefficients of the representing tensor in general. The MLG singular space is defined as follows.

Definition 10 (MLG Singular Space): For an MLG $\mathcal{M} = \{V, L, F\}$ with $M$ layers and $N$ nodes, the MLG singular space is defined as the space consisting of all singular tensors $\{V_1, \ldots, V_{MN}\}$ obtained from (49). The singular vectors $\{f_1, \ldots, f_M\}$ and $\{e_1, \ldots, e_N\}$ in (48) characterize layers and entities, respectively.

Similar to orderwise spectral analysis in Section V-B, we can define the MLG singular tensor transform (M-GST) based on the singular tensors as follows.

Definition 11 (M-GST): Suppose that $U_i = (f_i \odot e_i) \in \mathbb{R}^{M \times N \times M \times N}$ consists of the singular vectors of the representing tensor $F$ in (48), where $[U_{i\alpha}]_{\beta j} = [f_i]_{\alpha} \cdot [e_i]_{\beta j}$. The M-GST can be defined as the contraction between $U_i$ and the tensor signal $s \in \mathbb{R}^{M \times N}$, i.e.,
\[
\hat{s} = U_i \odot s.
\]
(50)

If the singular vectors are included in $W_f = [f_1, \ldots, f_M] \in \mathbb{R}^{M \times M}$ and $W_e = [e_1, \ldots, e_N] \in \mathbb{R}^{N \times N}$, the layerwise M-GST can be defined as
\[
\hat{s}_L = W_f^T s \in \mathbb{R}^{M \times N}
\]
(51)
and the entitywise M-GST can be defined as
\[
\hat{s}_N = s W_e \in \mathbb{R}^{M \times N}.
\]
(52)

Inverse M-GST can be defined similarly as in (47) with unitary $W_f$ and $W_e$.

Compared to the eigentensors in (22), the singular tensors come from the combinations of the singular vectors, thus are capable of capturing information of layers and entities more efficiently. Eigendecomposition, however, focuses more on the joint information and approximate the separate information of layers and entities. We shall provide further discussion on the performance of different decomposition methods in Section VII-D. The intuition of applying HOSVD in MLG analysis and its correlations to GSP are also provided in Section VII-A3.

D. Spectrum Ranking in the Multilayer Graph

In traditional GSP, the frequencies are defined by the eigenvalues of the shift, whereas the total variation is an alternative measurement of the order of the graph frequencies [3]. Similarly, we use the total variation of $\lambda_{ai}$ based on the spectral tensors to rank the MLG frequencies. Let $|\lambda|_{max}$ be the joint eigenvalue of the adjacency tensor $A$ with the largest magnitude. The M-GSP total variation is defined as follows:
\[
\text{TV}(V_{ai}) = \left\| V_{ai} - \frac{1}{|\lambda|_{max}} A \odot V_{ai} \right\|_1
\]
(53)
\[
= 1 - \frac{\lambda_{ai}}{|\lambda|_{max}} \cdot \|V_{ai}\|_1
\]
(54)
where $\|\cdot\|_1$ is the elementwise $l_1$ norm. Other norms could also be used to define the total variation. For example, the $l_2$ norm could be efficient in signal denoising [3]. The graph frequency related to $\lambda_{ai}$ is said to be a higher frequency if its total variation $\text{TV}(V_{ai})$ is larger, and its corresponding spectral tensor $V_{ai}$ is a higher frequency spectrum. If the representation tensor refers to Laplacian tensor, i.e., $L = D - A$, the frequency order is in contrast to the adjacency tensor $A$ as GSP [3]. We shall provide more details on interpretation of MLG frequency in Section VII-A.

VI. FILTER DESIGN

In this section, we introduce an MLG filter design together with its properties based on signal shifting.

A. Polynomial Filter Design

Polynomial filters are basic filters in GSP [7], [22]. In M-GSP, first-order filtering consists of basic signal filtering, i.e.,
\[
s' = f_1(s) = F \odot s.
\]
(55)
Similarly, a second order filter can be defined as additional filtering on first-order filtered signal, i.e.,
\[
s'' = f_2(s) = F \odot (F \odot s)
\]
(56)
whose entries $s_{ai}''$ are calculated as
\[
\hat{s}_{ai}'' = \sum_{\beta=1}^{M} \sum_{\gamma=1}^{N} F_{ai\beta} s_{\beta j}
\]
(58)
\[
= \sum_{\beta,j} F_{ai\beta} s_{\beta j}
\]
(59)
\[
= \sum_{\beta,j} F_{ai\beta} F_{\beta jep} s_{ep}
\]
(60)
\[
= (F \odot F) \odot s
\]
(61)
where $\odot$ is the contraction defined in (8).

Let $F_{[2]} = F \odot F$. From (22), we have
\[
F_{[2]}_{ai\beta j} = \sum_{\theta,p} F_{ai\theta p} F_{\theta j ep}
\]
(62)
\[
= \sum_{\theta,p} \left( \sum_k \lambda_k [V_{k\theta p}]_{ai} [V_{k\theta p}]_{aj} \right) \left( \sum_l \lambda_l [V_{l\beta p}]_{aj} [V_{l\beta p}]_{aj} \right)
\]
(63)
Similarly, for $r$th-order term $F^{(r)}$, its entry $F_{ai bj}^{(r)}$ can be calculated as $F_{ai bj}^{(r)} = \sum_k \lambda_k^r [V_k]_a [V_k]_b$.

Now we have the following property.

Property 5: The $r$th order basic shifting filter $f_r(s)$ can be calculated as

$$f_r(s) = F^{(r)} \circ s$$

$$= \left( \sum_{k=1}^{MN} \lambda_k^r V_k \circ V_k \right) \circ s.$$  

This property is the M-GSP counterpart to the traditional linear system interpretation that complex exponential signals are eigenfunctions of linear systems [3], and provide a quicker implementation of higher order shifting. With the $k$-order polynomial term, the adaptive polynomial filter is defined as

$$h(s) = \sum_{k} \alpha_k f^{(k)} \circ s$$  

where $\{\alpha_k\}$ are parameters to be estimated from data.

Adaptive polynomial filter is useful in semi-supervised classification [34] and exploits underlying geometric topologies. We will illustrate further and provide application examples based on MLG polynomial filtering in Section VIII.

B. Spectral Filter Design

Filtering in the graph spectral space is useful in GSP frequency analysis. For example, ordering the Laplacian graph frequency analysis. For example, ordering the Laplacian graph

$$B_{i j}$$

based on MLG polynomial filtering in Section VIII. We will illustrate further and provide application examples based on MLG polynomial filtering in Section VIII.

C. Discussion

We briefly discuss the interpretation of polynomial filters and spectral filters. From the spatial perspective, MLG polynomial filter is a weighted sum of messaging passing on the MLG in different orders, shown as (66). Each node collects information from both inter- and intra-layer neighbors, before combining them with its own information. From the spectrum perspective, M-GSP polynomial filters are eigenfunctions of linear systems, which are special cases of M-GSP spectral filters shown in (69). The M-GSP spectral filters assign different weights to each M-GSP spectrum via functions $f(\cdot)$ and $g(\cdot)$ depending on specific tasks. Both M-GSP polynomial filters and spectral filters can be useful for high-dimensional IoT signal processing. More discussions and examples of M-GSP filters are presented in Section VIII.

VII. DISCUSSION AND INTERPRETATIVE INSIGHTS

A. Interpretation of M-GSP Frequency

1) Interpretation of Graph Frequency: To better understand its physical meaning, we start with total variation in digital signal processing (DSP), and provide the relationship between GSP/M-GSP and traditional GSP. The total variation in DSP is defined as differences among signals over time [35]. Moreover, the total variations of frequency components have a 1-to-1 correspondence to frequencies in the order of their values. If the total variation of a frequency component is larger, the corresponding frequency with the same index is higher. This means that, a higher frequency component changes faster over time and exhibits a larger total variation. Interested readers could refer to [3] and [8] for a detailed interpretation of total variation in DSP.

Now, let us elaborate the graph frequency motivated by the cyclic graph. Rewrite the finite signals in DSP as vectors, i.e., $s = [s_1, \ldots, s_N] \in \mathbb{R}^N$, the signal shifting can be interpreted as the shift filtering corresponding to a cyclic graph shown in Fig. 6. Suppose that its adjacency matrix is written as

$$C_N = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 \\ 0 & \cdots & 1 & 0 & 0 \end{bmatrix}.$$  

Then, the shifted signal over the cyclic graph is calculated as $s' = C_N s = [s_N, s_1, \ldots, s_{N-1}]^T$, which shifts the signal at each node to its next node. More specifically, $C_N$ can be decomposed as $C_N = V \Lambda V^{-1}$ with eigenvalues $\lambda_n = e^{-j2\pi n/N}$ and discrete Fourier matrix $V^{-1} = (1/\sqrt{N})[e^{-j2\pi kn/N}]$ as eigenspace. Inspired by the DSP, the eigenvectors in $V$ are the spectral components (spectrum) of the cyclic graph and the eigenvalues are related to the graph.
frequencies [3]. Note that the traditional GSP would degrade to DSP if the graph structure is the cyclic graph.

Generalizing the adjacency matrix of the cyclic graph to the representing matrix \( F_M \) of an arbitrary graph, the graph Fourier space consists of the eigenvectors of \( F_M \) and the graph frequencies are related to the eigenvalues. More specifically, the graph Fourier space can be interpreted as the manifold or spectral space of the representing matrix. As aforementioned, the total variations of frequency components reflect the order of frequencies, we can also use the total variation, i.e., \( TV(e) = ||e - (1/||\lambda_{\text{max}}||)\lambda||_1 \), to rank the graph frequencies, where \( e_i \) is the spectral component related to the eigenvalue \( \lambda_i \) in \( F_M \). Similar to DSP, the graph frequency indicates the oscillations over the vertex set, i.e., how fast the signals change over the graph shifting.

2) Interpretation of MLG Frequency: Now, return to M-GSP. Given spectral tensors \( V_k \in \mathbb{R}^{M \times N} \) of an MLG, a signal \( s \in \mathbb{R}^{M \times N} \) can be written in a weighted sum of the spectrum, i.e., \( s = \sum_k a_k V_k \). Viewing the spectral tensor as a signal component, the total variation is in the form of differences between the original signal and its shifted version in (53). If the signal component changes faster over the MLG, the corresponding total variation is larger. Since we relate higher frequency component with a larger total variation, the MLG frequency indicates how fast the signal propagates over the MLG under the representing tensor. If a signal \( s \) contains more high-frequency components, it changes faster under the representing tensor. Similar to GSP, M-GSP degenerates to classic DSP if the MLG is a single-layer cyclic graph.

3) Interpretation of MLG Singular Tensors: As discussed in Section VII-A1, the name of graph Fourier space arises from the adjacency matrix of the cyclic graph. However, when the algebra representation is generalized to an arbitrary graph, especially the Laplacian matrix, the definition of graph spectrum is less related to the Fourier space in DSP but can be interpreted as the manifold or subspace of the representing matrix instead. In literature, SVD is an efficient method to obtain the spectrum for signal analysis, such as spectral clustering [36] and PCA analysis [37]. It is straightforward to generalize graph spectral analysis to graph singular space, especially for the Laplacian matrix. In M-GSP, the orderwise singular vectors can be interpreted as subspaces characterizing features of layers and entities, respectively. Since HOSVD is robust and efficient, transforming signals to the MLG singular space (M-GST) for the analysis of underlying structures can be a useful alternative for M-GFT.

B. Interpretation of Entities and Layers

To gain better physical insight of entities and layers, we discuss two categories of data sets.

1) In most of the physical systems and data sets, signals can be modeled with a specific physical meaning in terms of layers and entities. In smart grid, for example, each station can be an entity, connected in two layers of computation and power transmission, respectively. Another example is video in which each geometric pixel point is an entity and each video frame form a layer. Each layer node denotes the pixel value in that video frame. M-GSP can be intuitive tool for these data sets and systems.

2) In some scenarios, however, the data sets usually only has a definition of layers without meaningful entities. In particular, for MLGs with different numbers of nodes, we may insert some isolated artificial nodes to augment the MLG. Often in such applications, it may be harder to identify the physical meaning of entities. Here, the entities may be virtual and are embedded in the underlying structure of the MLG. Although definition of a virtual entity may vary with the chosen adjacency tensor, it relates to the topological structure in terms of global spectral information. For example, in Fig. 7, we can use two different definitions of virtual entities. Although the representing tensors for these two definitions differ, their eigenvalues remain the same. Considering also layer-wise flattening, the two supra-matrices are related by reshaping, by exchanging the fourth and fifth columns and rows. They still have the same eigenvalues, whereas the eigentensors can also be the same by implementing the reshaping operations. Note that, to capture distinct information from entities, their spectra would change with different definitions of virtual entities.

C. Distinction From and Connection to Existing GSP Works

1) Graph Signal Processing: Generally, M-GSP extends traditional GSP into MLGs. To process an MLG with \( M \) layers and \( N \) nodes in each layer via GSP, one may stack all MLG layers to represent them with a supra-matrix \( A_F \in \mathbb{R}^{MN \times MN} \), which flattens the M-GSP adjacency tensor \( A \in \mathbb{R}^{M \times N \times M \times N} \). As illustrated in Section V-A, the joint M-GSP Fourier basis \( V_i \in \mathbb{R}^{M \times N} \) reorganizes eigenvectors \( v_i \in \mathbb{R}^{MN \times 1} \) of \( A_F \), i.e., graph Fourier bases. However, such matrix representation makes GSP inefficient in extracting features of layers and entities separately. Given a supra-matrix of the MLG, the layers of nodes cannot be identified directly from its index since all the nodes are treated equivalently. Different from matrix-based GSP, the tensor representation provides clear identifications on layers in its index. Moreover, in GSP, we can only implement a joint transform to process inter- and intra-layer connections together, while the M-GSP provide a more flexible choice on joint and orderwise analysis. In Section V-A, the joint M-GSP analysis introduced can be viewed as the bijection of GFT in the flattened MLG, with vertices indexed by both layers and entities. Beyond that, we flexibly provide orderwise spectral analysis based on tensor decompositions, which allow the orderwise analysis on layers and nodes. One can select suitable MLG-based tools depending on tasks. The joint spectral analysis can be implemented if we aim to explore layers and entities.
fully, whereas the orderwise spectral and singular analysis are more efficient in characterizing layers and entities separately.

2) Joint Time-Vertex Fourier Analysis: In [16], a JFT is defined by implementing GFT and DFT consecutively. As discussed in Section VII-A1, the time sequences can be interpreted under a cyclic graph, and thus reside on an MLG structure. However, JFT assumes that all the layers have the same intralayer connections, which limits the generalization of the MLG analysis. Differently, the tensor-based representation allows heterogeneous structures for the intralayer connections, which makes M-GSP more general.

3) Multiway Graph Signal Processing: In [18], MWGSP has been proposed to process high-dimensional signals. Given $K$th-order high-dimensional signals, one can decompose the tensor signal in different orders, and construct one graph for each. Graph signal is said to reside on a high-dimensional product graph obtained by the product of all individual factor graphs. Although the MW-GFT is similar to M-GFT for $K=2$, there still are notable differences in terms of spectrum. First, MWGSP can only process signals without exploiting a given structure since multiple graph spectra would arise from each order of the signals. For an MLG with a given structure, such as physical networks with heterogeneous intralayer connections, MWGSP does not naturally process it efficiently and cohesively. The orderwise spectra come from factor graphs of each order in MWGSP while M-GSP spectra are calculated from the tensor representation of the whole MLG. Second, MWGSP assumes all the layers residing on a homogeneous factor graph and restricts the types of manageable MLG structure. For example, in a spatial temporal data set, a product graph, formed by the product of spatial connections and temporal connections, assumes the same topology in each layer. However, many practical systems and data sets feature more complex geometric interactions. M-GSP provide a more intuitive and natural framework for such MLG. In summary, despite some shared similarities between MW-GFT and M-GFT in some scenarios, they serve different purposes and are suitable for different underlying data structures.

D. Comparison of Different Decomposition Methods

To compare recovery accuracy of representing tensor using different tensor decomposition methods, we examine Eigen tensor decomposition, HOSVD, optimal CP decomposition, and Tucker decomposition in MLGs randomly generated from the Erdős – Rényi (ER) random graph $ER(p, q, M, N)$. Here, $M$ is the number of layers with $N$ nodes in each layer, $p$ is the intralayer connection probability and $q$ is the interlayer connection probability. We apply different decomposition methods of similar complexity, and compute errors between decomposed and original tensors. From Table I, we can see that the eigentensor decomposition and HOSVD exhibit better overall accuracy. Generally, eigentensor decomposition is better suited for applications emphasizing joint features of layers and entities. On the other hand, HOSVD is effective at separating individual features of layers and entities. Note that, in addition to recovery accuracy, different decompositions may have different performance advantages when capturing different data features that can be better measured with different metrics.

E. Complexity of Spectrum Calculation

To apply M-GSP in practical scenarios, the major computational complexity resides within the tensor decomposition for spectrum calculation. Here, we provide a complexity analysis of M-GSP in comparison with existing GSP approaches.

1) Supra-Matrix-Based GSP: To analyze the flattened MLG via M-GSP, one may represent the geometric structure as an $M \times MN$ supra-adjacency matrix. Next, eigendecomposition can be used to calculate the $M$-dimensional graph Fourier basis vectors, i.e., $v_i \in \mathbb{R}^{MN}$. The complexity of classic eigendecomposition is $O(M^3N^3)$, and the required memory size of Fourier spectra is $M^2N^2$. Note that, supra-based GSP can only analyze the joint layer-entity properties, but is unable to further decompose individual layer- and entitywise characterizations.

2) MWGSP Based on Individual Matrices: Other GSP-based approaches, such as JFT and multiway GSP, would construct MLG from two factor graphs with size $N \times N$ and $M \times M$. Eigendecomposition is applied to each factor graph to obtain the spectra for both factor graphs. The complexity here is $O(M^3N^3)$ and memory size of spectra is $N^2 + M^2$. However, such individual matrix approach has a critical drawback by assuming the same structure for all layers, thereby limiting its generalization.

3) M-GSP Singular Analysis: In M-GSP singular analysis, HOSVD is applied for spectra calculation. To calculate singular tensors, one may first flatten it into an $M \times (NMN)$ matrix for orders 1, 3 and an $N \times (NMN)$ matrix for orders 2, 4. Since the adjacency tensor is partially symmetric, SVD is applied to these two matrices, of complexity $O(M^2N^2 + M^3N^2)$. Memory size required for the two modes of singular vectors is $M^2 + N^2$. Note that here we only consider the classic SVD [38]. More advanced HOSVD can further lower complexity, including fast-HOSVD [39] and Quantum-HOSVD [40].

4) M-GSP Fourier Analysis: For M-GSP Fourier analysis, we propose a joint analysis and approximate individual analysis based on orthogonal CP decomposition. The joint analysis has the same complexity and storage need as the supra-matrix GSP analysis. For individual analysis, the decomposition usually relies on optimization formulation and weighs a tradeoff between accuracy and complexity. Interested readers could refer to literature, such as [26], for more details. The storage size for approximated Fourier spectra is $M^2 + N^2$.

| Table I |
|---------|
| Graph Structure | Error of Decomposing the Representing Tensor |
|------------|----------------|
| Eigen-tensor | ER(0.3,0.3,5) | ER(3,0.7,1,15) | ER(0,0.7,2,15) |
| HOSVD      | 8.3893e-15    | 1.6001e-13    | 3.3347e-13    |
| CP         | 1.011e-14     | 1.9563e-13    | 1.9056e-13    |
| Tucker     | 9.22e-01      | 8.82e-01      | 9.24e-01      |
| M-GSP      | 3.37e-03      | 9.70e-05      | 9.40e-03      |
In summary, our proposed M-GSP has similar or lower complexity than traditional GSP, and the same memory size as MWGSP. Compared to MWGSP, although M-GSP may need more computation, M-GSP is more powerful in processing MLG with heterogeneous structures, which cannot be achieved in MWGSP. New advances in fast tensor decomposition can make M-GSP more efficient for spectra calculation in big data analysis.

VIII. APPLICATION EXAMPLES

We now provide some illustrative application examples within our M-GSP framework.

A. Analysis of Cascading Failure in IoT Systems

Analysis of cascading failure in IoT systems based on the spreading processes in MLGs has recently attracted significant interests [41]. Modeling the failure propagation in complex systems by shifting over MLG, M-GSP spectrum theory can help the analysis of system stability. In this part, we introduce an M-GSP analysis for cascading failure over multilayer cyber-physical systems (CPS) based on epidemic model [12]. Shown in Fig. 1(b), a CPS with $M$ layers and $N$ nodes in each layer can be intuitively modeled by an MLG with adjacency tensor $A \in \mathbb{R}^{M \times N \times M \times N}$.

Here, we consider the susceptible-infectious-susceptible (SIS) model [42] for the failure propagation. In the SIS model, each node has two possible states: susceptible (not fail) or infectious (fail). At each time slot, the infectious node may cause failure to other nodes through directed links at certain infection rates, or it may heal itself spontaneously at a self-cure rate. The initial attack make several nodes infectious.

Since the nodes in the same layer correspond to the same functionality, e.g., power transmission, nodes on the same layer have the same self-cure rate and infection rate. The notations of the epidemic model are given as follows.

1) $\mu_\alpha$: Self-cure rate for nodes on layer $\alpha$.
2) $\theta_{\alpha \beta}$: Infection rate describing failure propagation probability from nodes on layer $\beta$ to those on layer $\alpha$.
3) $P_{i,\alpha}(t)$: Failure probability of the projected node of entity $i$ on layer $\alpha$ at time $t$.
4) $\epsilon_{i,\alpha}(t)$: Transition probability that the projected node of entity $i$ on layer $\alpha$ shifts from infectious state to susceptible state at time $t$.
5) $\sigma_{i,\alpha}(t)$: Transition probability that the projected node of entity $i$ on layer $\alpha$ remains susceptible at time $t$.

Since an infectious node becomes susceptible if it cures itself without being infected by its neighbors, we have

$$\epsilon_{i,\alpha}(t) = \mu_\alpha \prod_{j,\beta} \left[1 - \theta_{\alpha \beta} A_{i \alpha j} P_{j,\beta}(t)\right].$$

Similarly, a susceptible node remains susceptible without being infected by its neighbors. Thus

$$\sigma_{i,\alpha}(t) = \prod_{j,\beta} \left[1 - \theta_{\alpha \beta} A_{i \alpha j} P_{j,\beta}(t)\right].$$

The state transition forms a Markov chain, for which we derive the failure probability as

$$P_{i,\alpha}(t) = 1 - \prod_{j,\beta} \left[1 - T_{i \alpha j} P_{j,\beta}(t)\right].$$

where $T_{i \alpha j} = (1 - \mu_\alpha) \delta_{i \alpha j} + \theta_{i \alpha j} A_{i \alpha j}$, with $\delta_{i \alpha j} = 1$ if $(j, \beta) = (i, \alpha)$; otherwise, $\delta_{i \alpha j} = 0$.

We can define a transition tensor $T \in \mathbb{R}^{M \times N \times M \times N}$ with elements $T_{i \alpha j} = \delta_{i \alpha j}$ to characterize the failure propagation in (73). In steady state, $P_{i,\alpha}(\tau) = P_{i,\alpha}(\tau - 1)$. Moreover

$$\tilde{P}_{i,\alpha} = 1 - \prod_{j,\beta} \left[1 - T_{i \alpha j} P_{j,\beta}\right].$$

where $\tilde{P}_{i,\alpha}$ is the failure probability of the projected node of entity $i$ on layer $\alpha$ in steady state. Following [12], we can arrive at $\tilde{P}_{i,\alpha} = 0$ if the spectral radius of the transition tensor $\rho(T) < 1$, which indicates no failed nodes in steady state. Thus, $\rho(T)$ could serve as an indicator for system robustness.

To evaluate the effect of $\rho(T)$, we simulate the cascading failure in a seven-layer MLG, with intralayer connections in layers 2, 4 generated by $\text{Erd\'os} - \text{R\'enyi}$ models $\text{ER}(N, p)$. Here, $N = 33$ is the number of nodes in each layer and $p_i$ is the connection probability to generate the intralayer graph structure in layer $i$. The interlayer connections follow the multiplex structure. We first test node failures over time with different $\rho(T)$, where $p_2 = 0.5$, $p_4 = 0.4$ and other $p_i = 0$ for the intralayer connections. We initialize 10 failed nodes in the MLG system, and repeat the test over 1000 times. We change the value $\rho(T)$ through different epidemic parameters. Fig. 8(a) shows the averaged results, in which the fraction of failed nodes approaches zero if $\rho(T) < 1$ which matches our theoretical conclusion. We also test the fraction of failed nodes in different systems at the stable states in Fig. 8(b). From these results, the fraction of failed nodes is approximately zero for $\rho(T) < 1$, further illustrating the power of $\rho(T)$ as a indicator to evaluate system robustness. Here, to avoid being too repetitive, we merely introduce a simple example of MLG-based cascading failure analysis. Interested readers may refer to another work [12] for more details. With a better understanding of M-GSP spatial shifting, one can develop more general analysis for various failure models in multilayer IoT systems.

![Fig. 8. Influences of Spectral Radius: fraction of failed nodes with different $\rho(T)$. (a) Fraction of failed nodes with different spectral radius over time. (b) Fraction of failed nodes at stable state.](image-url)
B. Spectral Clustering

Clustering is a widely used tool in a variety of applications, such as social network analysis, computer vision, and IoT. Spectral clustering is popular and effective among many variants. Modeling data set by a normal graph before spectral tions, such as social network analysis, computer vision, and MLG-Based Unsupervised Image Segmentation

Algorithm 1 MLG-Based Unsupervised Image Segmentation

1: Input: RGB Image $I \in \mathbb{R}^{P \times Q \times 3}$
2: Build $N$ superpixels for the image $I$ and calculate the value of superpixel based on the mean of all pixels inside that superpixel, i.e., $s \in \mathbb{R}^{N \times 3}$
3: Construct a multilayer graph $A \in \mathbb{R}^{M \times N \times M \times N}$
4: Find entity-wise spectrum $E = [e_1, \ldots, e_N] \in \mathbb{R}^{N \times N}$
5: Select the first $K$ important leading spectrum based on the eigenvalues (singular values) of $E$ as $C \in \mathbb{R}^{N \times K}$
6: Cluster each row of $C$, and assign the $i$th superpixel into $j$th cluster if the $i$th row of $C$ is clustered into $j$th group
7: Assign all pixels inside one superpixel to the cluster of that superpixel
8: Output: The segmented image

$k$-means clustering by using a public BCCD blood cell data set shown in Fig. 9(a). In this data set, there are mainly three types of objects, i.e., white blood cell (WBC) versus red blood cell (RBC) versus Platelet (P). We set the number of clusters to 3 and $N = 1000$. For GSP-based spectral clustering, we construct graphs based on the Gaussian model by using information from all 3 color values $\sum_{\ell=1}^{2} |s_{i}(\ell) - s_{j}(\ell)|^2$ to form edge connections in a single graph. There is only a single $\delta_\ell$ and $t_\ell$ in the Gaussian model. For M-GSP, we use the MLG singular vectors (MLG-SVD), and tensor factorization (MLG-FZ) for spectral clustering, separately. Their respective results are compared in Fig. 9. WBCs are marked yellow, and RBCs are marked green. Platelet (P) is marked blue.

To test the proposed Algorithm 1, we first compare its results with those from the GSP-based method and traditional

$k$-means clustering by using a public BCCD blood cell data set shown in Fig. 9(a). In this data set, there are mainly three types of objects, i.e., white blood cell (WBC) versus red blood cell (RBC) versus Platelet (P). We set the number of clusters to 3 and $N = 1000$. For GSP-based spectral clustering, we construct graphs based on the Gaussian model by using information from all 3 color values $\sum_{\ell=1}^{2} |s_{i}(\ell) - s_{j}(\ell)|^2$ to form edge connections in a single graph. There is only a single $\delta_\ell$ and $t_\ell$ in the Gaussian model. For M-GSP, we use the MLG singular vectors (MLG-SVD), and tensor factorization (MLG-FZ) for spectral clustering, separately. Their respective results are compared in Fig. 9. WBCs are marked yellow, and RBCs are marked green. Platelet (P) is marked blue. From the illustrative results, MLG methods exhibit better robustness and are better in detecting regions under noise. Comparing results from different MLG-based methods, we find MLG-FZ to be less stable than HOSVD, partly due to approximation algorithms used for tensor factorization. Overall, MLG-based methods show reliable and robust performance over the GSP-based method and $k$-means.

In addition to visual inspection of results for such images, we are further interested to numerically evaluate the performance of the proposed methods against some state-of-art
methods for several more complex data sets that contain more classes. For this purpose, we test our methods on the BSD300 and BSD500 data sets [43]. BSD300 contains 300 images with labels, and BSD500 contains 500 images with labels. We first cluster each image, and label each cluster with the best map of cluster orders against the ground truth. Numerically, we use mean of intersection-over-union (mIOU), also known as the mean Jaccard Distance, for all clusters in each image to measure the performance. The Jaccard Distance between two groups $A$ and $B$ is defined as
\[
J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{76}
\]
A larger mIOU indicates stronger performance. To better illustrate the results, we considered two setups of data sets, i.e., one containing fewer classes (coarse) and one containing all images (all). We compare our methods together with $k$-means, GSP-based spectral clustering, invariant information clustering (IIC) [44], graph-based segmentation (GS) [45], back propagation (BP) [46] and differentiable feature clustering (DFC) [47]. The best performance is marked in bold. From the results of Table II, we can see that larger number of clusters in the first two columns generate worse performance. There are two natural reasons. First, the mapping of the best order of cluster labels is more difficult for more classes. Second, the graph-based spectral clustering is sensitive to the number of $K$ leading spectra and the structure of graphs. Regardless, MLG-based methods still demonstrate better performance. To estimate parameters $a_i$ of $f(\cdot)$, Optimization can be formulated to minimize, e.g., the mean square error (MSE) from ground truth label $y_L$
\[
\min_a ||M(f(s))_L - y_L||_2^2 \tag{78}
\]
where $M(\cdot)$ is a mapping of filtered signals to their discrete representation. For example, in a $\{\pm 1\}$ binary classification, one can assign a label to a filtered signal against a threshold (e.g., 0). Some other objective functions include labeling uncertainty, Laplacian regularization, and total variation. Using estimated parameters, we can filter the signal one more time to determine labels for some unlabeled data by following the same process.

Similarly, in an MLG, we can also apply polynomial filters for label estimation. Given an arbitrary data set $X = [x_1, \ldots, x_N] \in \mathbb{R}^{K \times N}$ with $N$ signal points and $K$ features for each node, we can construct an MLG by defining $M = K$ layers based on features and $N$ entities based on signal points. The inter- and intra-layer connections are calculated by the Gaussian distance with different parameters. Let its adjacency tensor $\mathbf{A} \in \mathbb{R}^{M \times N \times M \times N}$. A signal is defined by
\[
\mathbf{s} = \begin{bmatrix} s_{L_1} & \cdots & s_{L_K} \\ 0_{UL_1} & \cdots & 0_{UL_K} \end{bmatrix} \in \mathbb{R}^{M \times N} \tag{79}
\]
which is an extended version of graph signal. Note that we do not necessarily need to order signals by placing zeros in the rear. We only write the signal as (79) for notational convenience. We now apply polynomial filters on signals, i.e.,
\[
s_1 = h_1(s) = \sum_i a_i A^{[i]} \odot s \tag{80}
\]
and
\[
s_2 = h_2(s) = A^{[i]} \odot s. \tag{81}
\]

For a filtered signal $s_X \in \mathbb{R}^{M \times N}$ ($X = 1, 2$), we define a function to map 2-D signals into 1-D by calculated the columnwise mean of $s_X$, i.e.,
\[
\bar{s}_X = \text{mean}_c(\mathbf{s}_X) \in \mathbb{R}^{1 \times N}. \tag{82}
\]

Next, we can define a function $M(\cdot)$ on $\bar{s}_X$ and consider certain objective functions in filter design. To validate the efficacy of polynomial filtering in the MLG framework, we test $h_1(\cdot)$ and $h_2(\cdot)$ for the binary classification problem on the Cleveland Heart Disease data set. In this data set, there are 297 data points with 13 feature dimensions. We directly build an MLG with $N = 297$ nodes in each of the $M = 13$ layers.

### Table II

|                | BSD300(N=100, all) | BSD300(N=300, all) | BSD300(N=100, Coarse) | BSD300(N=300, Coarse) | BSD300(N=900, Coarse) | BSD500 | BSD500 | BSD500 |
|----------------|-------------------|--------------------|-----------------------|-----------------------|-----------------------|--------|--------|--------|
| GSP            | 0.1237            | 0.1149             | 0.3225                | 0.3087                | 0.3067                | 0.3554 |        |        |
| K-MEANS        | 0.1293            | 0.1252             | 0.3044                | 0.3105                | 0.3124                | 0.3154 |        |        |
| MLG-SVD        | 0.1326            | 0.1366             | 0.3344                | 0.3394                | 0.3355                | 0.3743 |        |        |
| MLG-CP         | 0.1321            | 0.1293             | 0.3195                | 0.3236                | 0.3243                | 0.3641 |        |        |
| IIC            |                   |                    |                       |                       | 0.2071                |        |        |        |
| GS             |                   |                    |                       |                       | 0.2538                |        |        |        |
| BP             |                   |                    |                       |                       | 0.2339                |        |        |        |
| DFC            |                   |                    |                       |                       | 0.3719                |        |        |        |

C. Semi-Supervised Classification

Semi-supervised classification is an important practical application for IoT intelligence. In this application, we apply MLG polynomial filters for semi-supervised classification. Traditional GSP defines adaptive filter as
\[
f(s) = \sum_i a_i W^i s \tag{77}
\]
where $W$ is an adjacency matrix based on pairwise distance or a representing matrix constructed from the adjacency matrix.
More specifically, we directly use the labels as \( s \). For \( h_1(\cdot) \) (MLN-AF), we set \( a_i \neq 0 \) for at least one \( i > 0 \). Using MSE as objective function, we apply a greedy algorithm to estimate parameters \( \{a_i\} \). We limit the highest polynomial order to 10. For \( h_2(\cdot) \) (MLN-APF), we estimate a classification threshold via the mean of \( \bar{s}_X \) by setting the polynomial order \( i = 10 \).

We compare our methods with the GSP-based method in similar setups as in aforementioned examples. The only difference is that we use \( \bar{s}_X \) in M-GSP and use \( s' = f([s_{\mathbf{L}}^T, 0_{10}^T]^T) \) in GSP for mapping and classification. We also present the results of label propagation and SVM for comparison. We randomly split the test and training data for 100 rounds. From the results shown in Fig. 10, GSP-based and M-GSP-based methods exhibit better performance than traditional learning algorithms, particularly when the fraction of test samples is large. In general, M-GSP-based methods demonstrate superior performance among all methods owing to its strength to extract “multilayer” features, which could potentially benefit semi-supervised classification tasks in IoT systems.

**D. Dynamic Point Cloud Analysis**

3-D perception plays an important role in the high-growth fields of IoT devices and CPSs, and continues to drive many progresses made in advanced point cloud processing [48]. Here, we propose a short time M-GST method to analyze dynamic point cloud. Given a dynamic point cloud with \( M \) frames and at most \( N \) points in each frame, we model it as an MLG with \( M \) layers and \( N \) nodes in each layer. More specifically, we test the singular spectral analysis over the motion sequences of subject 86 in the CMU database [49]. To implement the M-GST, we first divide the motion sequence into several shorter sequences with \( N_f \) frames. Next for each shorter sequence, we model interlayer connections by connecting points with the same label among successive frames. For points in the same frame, we connect two points based on the Gaussian-kernel within a Euclidean threshold \( \tau_e \) [6]. Let \( \mathbf{x}_i \) be the 3-D coordinates of the \( i \)th point. We assign an edge weight between two points \( \mathbf{x}_i \) and \( \mathbf{x}_j \) as a nonzero \( A_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma^2) \) only if \( \|\mathbf{x}_i - \mathbf{x}_j\|_2 \leq \tau_e \). Next, we estimate the spatial and temporal basis vectors of each shorter term sequences by HOSVD in (48). Finally, we use the 3-D coordinates of all points in each shorter term sequences as signals and calculate their M-GST. To illustrate the results of M-GST, we examine the spectrogram similar to that of short-time Fourier transform (STFT) [50]. In Fig. 11, we transform the signal defined by the coordinates in \( Z \) dimension via M-GST and illustrate the transformation results for the divided frame sequence. From Fig. 11, one can easily identify different motions based on the MLG singular analysis.

To explore motions in dynamic point clouds, we can also apply the entitywise MLG highpass filters described in Section VI-B to capture critical details of human bodies. More specifically, we select the first 140 frames in “walking” and define the norm of three coordinates as signals. We select five body joints (entities) in each temporal frame (layers) shown as Fig. 12. From the results shown, entity 1 and entity 2 exhibit periodic patterns which are linked to the leg motion. Entity 3 (head) shows little movement relative to the main body. Entity 4 and entity 5 (hands) display more irregular patterns since they do not directly identify “walking.” To summarize, the MLG highpass filter can efficiently capture some key information of body movement and identify the meaning of nodes (entities). These and related information can assist further analysis of dynamic point clouds, including compression and classification.

Our future works shall target more practical applications of point cloud on IoT devices, including point cloud compression, low-complex point cloud segmentation, and robust denoising.

**E. Other Potential Applications in IoT Systems**

Along with the widespread deployment of IoT technologies, system structures become increasingly complex. Traditional graph-based tools are less adept at modeling “multilayer” graph interactions. The more general model of M-GSP...
provides additional opportunities for IoT applications. Here, we suggest several potential scenarios in IoT systems for M-GSP.

1) **Complex System Analysis:** In many IoT systems, such as vehicular network and smart grid, each station or vehicle may include multiple functions and thus different topological connections for each function, which fit naturally to MLG. With the MLG system model, M-GSP can be designed for various tasks, such as intrusion detection, resource management, and state prediction.

2) **Multilevel Data Processing:** In the era of big data, data sets are increasingly more complex and may display multilevel heterogeneous structures. One typical example is a multiband radiomap [51], where each band describes the spatial radio power distributions, for which the spatial-spectral relationships among each location can be modeled by MLG. Another example models time-varying signals generated by sensors where spatial–temporal data structures can also be characterized by MLG. Other typical examples in IoT include dynamic point cloud for autonomous driving, videos for object tracking and hyperspectral images for satellite sensing. For such multilevel IoT-generated data, M-GSP tools, such as adaptive filters and MLG learning machines, can be developed for signal prediction and node classification.

Overall, the power of MLG in extracting underlying “multilayer/multilevel” structures in the IoT systems makes M-GSP a potentially important tool in handling high-dimensional signal processing and learning tasks. Extending the M-GSP introduced in this work, we have successfully developed additional M-GSP applications in image processing, smart health, and hyperspectral image analysis. Interested readers may refer to [52] and [53] for more details.

**IX. CONCLUSION AND FUTURE DIRECTIONS**

In this work, we present a novel tensor-based framework of M-GSP that naturally generalizes the traditional GSP to MLGs. We first present the basic foundation and definitions of M-GSP, including MLG signals, signal shifting, spectral space, singular space, and filter design. We also provide interpretable discussion and physical insights through numerical results and examples to illustrate the strengths, general insights, and benefits of novel M-GSP framework. We further demonstrate exciting potentials of M-GSP in data processing applications through experimental results in several practical scenarios.

As an emerging field in system analysis and data processing, there are both challenges and opportunities in M-GSP and its applications. From the perspective of IoT intelligence and data analysis, we discuss the challenges and suggest potential future directions in M-GSP.

1) Graph models play important roles in graph-based analysis. However, good graph construction remains an open problem. In traditional GSP analysis, graphs are constructed from data similarity based on the Gaussian models [3], which requires tuning Gaussian parameters. For a more efficient graph construction, one potential direction is to optimize MLG structure to emphasize graph properties, such as graph smoothness and sparsity. In addition, one can also directly optimize MLG spectrum instead of adjacency tensor based on M-GSP properties, such as MLG total variation and MLG stationary process.

2) As discussed in Section VII-E, efficient complexity reduction benefits M-GSP analysis in large system and data sets. In general, an efficient M-GSP shall consider low-complexity spectrum calculation and fast MLG transform. Thus, one emerging direction is to develop fast M-GFT/M-GST based on graph structures and tensor properties, similar to fast GFT in classic GSP [54]. Other directions also include efficient tensor decomposition for M-GSP.

We plan to explore these directions in our future works.

**APPENDIX**

Unlike for undirected graphs, representing tensors of directed graphs is asymmetric, thereby making each layer or entity characterized by a pair of spectral vectors. To find the spectrum space of a directed MLG, we also present two ways to compute.

1) **Flattening Analysis:** Similar to the representing tensor of undirected graphs, we flatten the representing tensor as a second-order supra-matrix, and define spectral space as the reshaped eigenvectors of the supra-matrix. The flattened matrix $A_{FX}$ (or $A_{FN}$, $A_{FL}$) can be decomposed as

$$A_{FX} \approx \Sigma \Sigma^{-1}$$

where $\Sigma \in \mathbb{R}^{MN \times MN}$ is the matrix of eigenvectors and $\Sigma = \text{diag}(\lambda_i)$ is a diagonal matrix of eigenvalues. Then, we can reshape the eigenvectors, i.e., each column of $\Sigma$ as $V_k \in \mathbb{R}^{M \times N}$, and reshape each row of $\Sigma^{-1}$ as $U_k \in \mathbb{R}^{M \times N}$. Consequently, the original tensor can be decomposed into

$$A \approx \sum_{k=1}^{MN} \lambda_k V_k \circ U_k.$$  \hspace{1cm} (84)

2) **Tensor Factorization:** We can also compute the spectrum from the tensor factorization based on CP-decomposition

$$A \approx \sum_{k=1}^{R} \lambda_k a_k \circ b_k \circ c_k \circ d_k$$

$$= \sum_{k=1}^{R} \lambda_k V_k \circ U_k$$

where $R$ is the rank of tensor, $a_k, c_k \in \mathbb{R}^M$ characterize layers, $b_k, d_k \in \mathbb{R}^N$ characterize entities, and
\( \mathbf{V}_k = \mathbf{a}_k \odot \mathbf{b}_k \), \( \mathbf{U}_k = \mathbf{c}_k \odot \mathbf{d}_k \in \mathbb{R}^{M \times N} \) characterize the joint features. Since there are MN nodes, it is clear that \( R \leq MN \). Note that, for a single layer, (85) reduces to

\[
\mathbf{A} \approx \sum_{k=1}^{N} \lambda_k \mathbf{v}_k \odot \mathbf{u}_k. \tag{87}
\]

Moreover, if \( \mathbf{V} = (\mathbf{v}_k) \) and \( \mathbf{U} = (\mathbf{u}_k) = \mathbf{V}^{-1} \) are orthogonal bases, (87) is in a consistent form of the eigendecomposition in a single-layer normal graph. In addition, (28) is also a special case of (85) if the MLG is undirected.

Since tensor decomposition is less stable when exploring the factorization of a specific order or when extracting the separate features in the asymmetric tensors, we will defer more general analysis of directed networks to future works.

REFERENCES

[1] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs,” IEEE Trans. Signal Process., vol. 61, no. 7, pp. 1644–1656, Apr. 2013.
[2] D. J. 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 Process. Mag., vol. 30, no. 3, pp. 83–98, May 2013.
[3] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” Proc. IEEE, vol. 106, no. 5, pp. 808–828, May 2018.
[4] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs: Frequency analysis,” IEEE Trans. Signal Process., vol. 62, no. 10, pp. 3042–3054, Jun. 2014.
[5] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, “Signal denoising on graphs via graph filtering,” in Proc. IEEE GlobalSIP, Atlanta, GA, USA, Dec. 2014, pp. 872–876.
[6] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovačević, “Fast resampling of three-dimensional point clouds via graphs,” IEEE Trans. Signal Process., vol. 66, no. 3, pp. 666–681, Feb. 2018.
[7] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, “Adaptive graph filtering: Multiresolution classification on graphs,” in Proc. IEEE Global Conf. Signal Inf. Process., Austin, TX, USA, Dec. 2013, pp. 427–430.
[8] S. Zhang, Z. Ding, and S. Cui, “Introducing hypergraph signal processing: Theoretical foundation and practical applications,” IEEE Internet Things J., vol. 7, no. 1, pp. 639–660, Jan. 2020.
[9] S. Barbarossa and S. Sardellitti, “Topological signal processing over simplicial complexes,” IEEE Trans. Signal Process., vol. 68, pp. 2992–3007, Mar. 2020.
[10] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter,”Multilayer networks,” J. Complex Netw., vol. 2, no. 3, pp. 203–271, Jul. 2014.
[11] M. De Domenico et al., “Mathematical formulation of multilayer networks,” Proc. 56th Allerton Conf. Commun., Control, Comput., Monticello, VA, USA, Sep. 2018, pp. 203–271.
[12] M. De Domenico, C. Granell, M. A. Porter, and A. Arenas, “The physics of spreading processes in multilayer networks,” Nat. Phys., vol. 12, no. 10, pp. 901–906, Aug. 2016.
[13] A. Bogojeska, S. Filiposka, I. Mishkovski, and L. Kocarev, “On opinion formation and synchronization in multiplex networks,” in Proc. 21st TELFOR, Belgrade, Serbia, Nov. 2013, pp. 172–175.
[14] A. C. Kinsley, G. Rossi, M. J. Silk, and K. Vanderwael, “Multilayer and multi-network structural robustness: An introduction to their use in veterinary epidemiology,” Front. Veterinary Sci., vol. 7, p. 596, Sep. 2020.
[15] S. Barbarossa and S. Sardellitti, “Topological signal processing over simplicial complexes,” IEEE Trans. Signal Process., vol. 68, pp. 2992–3007, Mar. 2020.
Songyang Zhang received the Ph.D. degree in electrical and computer engineering from the University of California at Davis, Davis, CA, USA, in 2021. He was a Postdoctoral Research Associate with the Department of Electrical and Computer Engineering, University of California at Davis from 2021 to 2023. Since 2023, he has been an Assistant Professor with the Department of Electrical and Computer Engineering, University of Louisiana at Lafayette, Lafayette, LA, USA. His current research interests include signal processing and learning over high-dimensional graphs, federated learning, biomedical signal processing, and wireless network optimization.

Zhi Ding (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Cornell University, Ithaca, NY, USA, in 1990. He is with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA, USA. His research team of enthusiastic researchers works very closely with industry to solve practical problems and contributes to technological advances. His team has collaborated with researchers around the world and welcomes self-motivated young talents as new members. His major research interests and expertise cover the areas of wireless networking, communications, signal processing, multimedia, and learning.

Dr. Ding received the IEEE Communication Society’s WTC Award in 2012 and the IEEE Communication Society’s Education Award in 2020. He has served as the Chief Information Officer and Chief Marketing Officer for the IEEE Communications Society. He was an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1994 to 1997, and from 2001 to 2004, and an Associate Editor of IEEE SIGNAL PROCESSING LETTERS from 2002 to 2005. He was the General Chair of the 2016 IEEE International Conference on Acoustics, Speech, and Signal Processing and the Technical Program Chair of the 2006 IEEE Globecon. He was also an IEEE Distinguished Lecturer (Circuits and Systems Society, 2004–2006, Communications Society, 2008–2009). He is a member of the Technical Committee on Statistical Signal and Array Processing and the Technical Committee on Signal Processing for Communications (1994–2003). He served on as IEEE Transactions on Wireless Communications Steering Committee Member (2007–2009) and its Chair (2009–2010).

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.