Jay S. Stanley III, Eric C. Chi, and Gal Mishne

Multiway Graph Signal Processing on Tensors

Integrative analysis of irregular geometries

Digital Object Identifier 10.1109/MSP.2020.3013555
Date of current version: 28 October 2020

Graph signal processing (GSP) is an important methodology for studying data residing on irregular structures. Because acquired data are increasingly taking the form of multiway tensors, new signal processing tools are needed to maximally utilize the multiway structure within the data. In this article, we review modern signal processing frameworks that generalize GSP to multiway data, starting from graph signals coupled to familiar regular axes, such as time in sensor networks, and then extending to general graphs across all tensor modes. This widely applicable paradigm motivates reformulating and improving classical problems and approaches to creatively address the challenges in tensor-based data. We synthesize common themes arising from current efforts to combine GSP with tensor analysis and highlight future directions in extending GSP to the multiway paradigm.

Introduction

During the past decade, GSP [1] has laid the foundation for generalizing classical Fourier theory as defined on a regular grid, such as time, to handle signals on irregular structures, such as networks. GSP, however, is currently limited to single-way analysis: graph signals are processed independently of one another, thus ignoring the geometry between multiple graph signals. Through the coming decade, generalizing GSP to handle multiway data, which are represented by multidimensional arrays and tensors, with graphs underlying each axis of the data will be essential for modern signal processing. This survey discusses the burgeoning family of multiway GSP (MWGSP) methods for analyzing data tensors as a dependent collection of axes.

To introduce the concept of *way*, consider a network of N sensors, each measuring a signal sampled at T time points. On the one hand, classic signal processing treats these signals as a collection of N independent, 1D time series, ignoring the relation structure of the graph. On the other hand, the standard GSP perspective treats the data as a collection of T independent, 1D graph signals that describe the state of all sensors for a given time point $t_j \in T$. Both are single-way perspectives that ignore

the underlying geometry of the other way (also referred to as a *mode*). The recent time–vertex (T–V) framework [2], [3] unifies these perspectives to form a dual-way framework that processes graph signals that are time-varying (note that the graph itself is static, while the signals are time-varying), thus bridging the gap between classical signal processing and GSP. While one of the axes of a T–V signal is a regular grid (time), in general, a regular geometry may not underlie any of the ways of the data, e.g., genes and cells in sequencing data and users and items in recommendation systems [4]–[6]. Thus, the T–V framework is a subset of a more general MWGSP framework that considers the coupling of multiple geometries, whether they are predefined temporal or spatial axes or irregular graph-based axes. MWGSP is, by definition, more versatile, and it is our main focus.

New signal processing tools are needed to maximally utilize the multiway structure within the data.

Classical signal processing and GSP typically address 1D and 2D signals [1]–[3], [7] and do not address data sets of higher dimensions. However, such data sets, given as multiway tensors, are becoming increasingly common in many domains. Mathematically, tensors generalize matrices to higher dimensions [8], and, in this article, the term *tensors* includes matrices (as they are order-two tensors). Examples of tensors include video, hyperspectral imaging, magnetic resonance imaging (MRI) scans, multisubject functional MRI data, chemometrics, epigenetics, trial-based neural data, and higher-order sparse tensor data, such as databases of crime incident reports, taxi rides, and ad-click information [9]–[14]. While tensors are the primary structure for representing D-dimensional signals, research on tensors and signal processing on tensors has primarily focused on factorization methods [8], [15], devoting less attention to leveraging the underlying geometry on the tensor modes. Recent MWGSP approaches incorporate graph smoothness in multiway tensor analysis for both robust tensor factorization [12]–[14] and the direct data analysis of tensors [9], [10].

In this overview of multiway data analysis, we present a broad viewpoint to simultaneously consider the general graphs underlying all modes of a tensor. Thus, we interpret multiway analyses in light of graph-based signal processing to consider tensors as multiway graph signals defined on multiway graphs. GSP is a powerful framework in the multiway setting, leading to intuitive and uniform interpretations of operations on irregular geometry. Thus, MWGSP is a nontrivial departure from classical signal processing, producing an opportunity to exploit joint structures and correlations across modes to more accurately model and process signals in real-world applications of current societal importance: the climate, the spread of epidemics and traffic, and complex systems in biology.

Both the GSP and tensor analysis communities have been developing methods for multiway data analysis and taken different but complementary strategies to solving common problems. We lay the mathematical and theoretical foundations, drawing on work from both communities to develop a framework for the higher-order signal processing of tensor data, and we explore the challenges and algorithms that result when one imposes relational structure along all axes of data tensors. At the heart of this framework is the graph Laplacian, which provides a basis for the harmonic analysis of data in MWGSP and an important regularizer in the modeling and recovery of multiway graph signals. We illustrate the breadth of MWGSP by reinterpreting classic techniques, such as the 2D discrete Fourier transform (DFT), as a special case of MWGSP and introduce a general multiway graph Fourier transform (MWGFT). Further, we review novel multiway regularizations that are not immediately obvious by viewing the data purely as a tensor. Thus, we synthesize into a coherent family a spectrum of recent and novel MWGSP methods across varied applications in inpainting, denoising, data completion, factor analysis, dictionary learning, and graph learning [4], [10], [11], [16]–[21].

Single-way GSP

GSP generalizes classical signal processing from regular Euclidean geometries, such as time and space, to irregular and non-Euclidean geometries represented discretely by a graph. In this section, we review basic concepts. A complete survey of GSP is provided in [1].

Graphs

This tutorial considers undirected, connected, and weighted graphs $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$ consisting of a finite vertex set $\mathcal{V}$, an edge set $\mathcal{E}$, and a weighted adjacency matrix $\mathbf{W}$. If two vertices v_i, v_j are connected, then $(v_i, v_j) \in \mathcal{E}$ and $\mathbf{W}_{i,j} = \mathbf{W}_{j,i} > 0$; otherwise, $\mathbf{W}_{i,j} = \mathbf{W}_{j,i} = 0$. We employ a superscript parenthetical index to reference graphs and their accompanying characteristics from a set of graphs $\mathcal{G}^{(i)}$; i.e., $G = \{\mathcal{G}^{(i)} = (\mathcal{V}^{(i)}, \mathcal{E}^{(i)}, \mathbf{W}^{(i)})\}_{i=1}^{D}$. Contextually, we will refer to the cardinality of the vertex set of a graph $\mathcal{G}^{(i)}$ as $|\mathcal{V}^{(i)}| = n_i$. When parenthetical indexing is not used, we refer to a general graph $\mathcal{G}$ on n nodes.

Graph signals

A signal $f : \mathcal{V} \to \mathbb{R}^n$ on the vertices of a graph on n nodes may be represented as a vector $\mathbf{f} \in \mathbb{R}^n$, where $\mathbf{f}_i = f(i)$ is the signal value at vertex $v_i \in \mathcal{V}$. The graph Fourier transform (GFT) decomposes a graph signal in terms of the eigenvectors of a graph shift operator. Many choices have been proposed for graph shifts, including the adjacency matrix $\mathbf{W}$ and various forms of the graph Laplacian $\boldsymbol{\mathcal{L}}$, which is a second-order difference operator across the edge set of the graph. In this article, we use the popular combinatorial graph Laplacian defined as $\boldsymbol{\mathcal{L}} \coloneqq \mathbf{D} - \mathbf{W}$, where the degree matrix $\mathbf{D}$ is diagonal, with elements $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. This matrix is real and symmetric. Its eigendecomposition is $\boldsymbol{\mathcal{L}} = \boldsymbol{\Psi}\boldsymbol{\Lambda}_{\mathcal{G}}\boldsymbol{\Psi}^*$, where the columns of $\boldsymbol{\Psi}$ are a complete set of orthonormal eigenvectors $\{\boldsymbol{\psi}_\ell\}_{\ell=0}^{n-1}$, $\boldsymbol{\Psi}^*$ is the conjugate transpose of $\boldsymbol{\Psi}$, and the diagonal of $\boldsymbol{\Lambda}_{\mathcal{G}}$ constitutes the real eigenvalues $\{\lambda_\ell\}_{\ell=0}^{n-1}$.

Graph Fourier analysis

The GFT and its inverse are

$$\hat{f}(\lambda_\ell) = \sum_{k=1}^{N} f(k)\psi_\ell^*(k) \quad \text{and} \quad f(k) = \sum_{\ell=0}^{N-1} \hat{f}(\lambda_\ell)\psi_\ell(k), \tag{1}$$

or, in matrix form, $\text{GFT}\{\mathbf{f}\} = \mathbf{\Psi}^*\mathbf{f}$. The GFT generalizes the classical Fourier transform since the former is the spectral expansion of a vector in the discrete graph Laplacian eigensystem, while the latter is the spectral expansion of a function in the eigensystem of the continuous Laplacian operator. Indeed, the GFT is synonymous with the DFT when the graph Laplacian is built on a cyclic path or a ring graph. It is typical to reinforce the classical Fourier analogy by referring to the eigenvectors of $\mathcal{L}$ as *graph harmonics* and the eigenvalues as *graph frequencies* and indexing the harmonics in ascending order of the eigenvalues such that the lowest indexed harmonics are the smoothest elements of the graph eigenbasis.

Despite these analogies, it is nontrivial to directly extend classical tools to signals on graphs. For example, there is no straightforward analog of convolution in the time domain to convolution in the vertex domain. Instead, filtering signals in the GFT domain is defined analogously to filtering in the frequency domain, with a filtering function $\hat{h}(\cdot)$ applied to the eigenvalues λ_ℓ that take the place of the frequencies

$$\tilde{f}(k) = \sum_{\ell=0}^{N-1} \hat{h}(\lambda_\ell)\hat{f}(\lambda_\ell)\psi_\ell(k), \tag{2}$$

where $\tilde{f}$ is the result of filtering f with the graph spectral filter $h(\mathcal{L})$. This spectral analogy is a common approach for generalizing classical notions that lack clear vertex interpretations.

Extending GSP to multiway spaces

Classical D-dimensional Fourier analysis provides a template for constructing unified geometries from various data sources. The D-dimensional Fourier transform sequentially applies a 1D Fourier transform to each axis of the data. For example, a 2D DFT applied to an $n_1 \times n_2$ real image $\mathbf{X}$ is

$$\begin{aligned}\text{2D-DFT}\{\mathbf{X}\} &= \text{DFT}_c(\text{DFT}_r(\mathbf{X})) = \text{DFT}_r(\text{DFT}_c(\mathbf{X})) \\ &= \mathbf{U}_{n_1}\mathbf{X}\mathbf{U}_{n_2},\end{aligned} \tag{3}$$

where DFT_r (DFT_c) applies the DFT to the rows (columns) of $\mathbf{X}$ and $\mathbf{U}_n$ denotes a normalized n-point DFT matrix: $\mathbf{U}_n(t,k) = (1/\sqrt{n})\exp\{-2\pi\mathbf{j}t(k-1)/n\}$ for $t, k = 1,\ldots,n$. This 2D transform decomposes the input into a set of plane waves. The 2D GFT is algebraically analogous to the 2D DFT. For two graphs $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$ on n_1 and n_2 vertices, the 2D DFT of $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$ is

$$\text{2D-GFT}(\mathbf{X}) = \text{GFT}_{n_1}(\text{GFT}_{n_2}(\mathbf{X})) = \text{GFT}_{n_2}(\text{GFT}_{n_1}(\mathbf{X})), \tag{4}$$

and it was presented in [7] as a method for efficiently processing big data. Note that when $\mathcal{G}^{(1)} = \mathcal{P}^{n_1}$ and $\mathcal{G}^{(2)} = \mathcal{P}^{n_2}$, i.e., they are cyclic path graphs on n_1 and n_2 vertices, this transform is equivalent to a 2D DFT [7].

In this section, we present the MWGSP framework for general D-dimensional signal processing on coupled and irregular domains, which enables holistic data analysis by considering relational structures on potentially all modes of a multiway signal. MWGSP encompasses standard GSP while extending fundamental GSP tools, such as graph filters to D dimensions. Furthermore, because graphs can be used to model discrete structures from classical signal processing, MWGSP forms an intuitive superset of discrete signal processing in domains such as images and video.

Tensors

Tensors are a data structure representing D-dimensional signals as well as a mathematical tool for analyzing multilinear spaces. We use both perspectives to formulate MWGSP. In this article, we adopt the tensor terminology and notation used by [8].

Tensors as a D-dimensional array

The number of ways or modes of a tensor is its order. Vectors are tensors of order one and denoted by boldface lowercase letters, e.g., $\mathbf{a}$. Matrices are tensors of order two and represented by boldface capital letters, e.g., $\mathbf{A}$. Tensors of higher orders, namely, order three and greater (see "Order-Three Tensors"), are indicated by boldface Euler script letters, e.g., $\mathcal{A}$. If $\mathcal{A}$ is a D-way data array of size $n_1 \times \cdots \times n_D$, we say $\mathcal{A}$ is a tensor of order D.

There are multiple operations to reshape tensors and that are used for convenient calculations. Vectorization maps the elements of a matrix into a vector in column-major order. That is, for $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$,

$$\text{vec}(\mathbf{X}) = [\mathbf{X}_{1,1}, \ldots, \mathbf{X}_{n_1,1}, \mathbf{X}_{12}, \ldots, \mathbf{X}_{n_1,2}, \ldots, \mathbf{X}_{1,n_2}, \ldots, \mathbf{X}_{n_1,n_2}]^{\text{T}}.$$

A tensor mode-ℓ vectorization operator, $\text{vec}_\ell(\mathcal{X})$, is similarly defined by stacking the elements of $\mathcal{X}$ in mode-d major order. Let $\text{ten}(\mathbf{x}, \ell, \{n_1, \ldots, n_D\}) = \mathcal{X}$ be the ℓth tensorization of $\mathbf{x}$, which is the inverse of the ℓ-major vectorization of $\mathcal{X}$. Denote by $n\backslash\ell = \prod_{i=1}^{\ell-1} n_i \prod_{j=\ell+1}^{D} n_j$ the product of all factor sizes except for the ℓth factor. Then, let $\text{mat}(\mathcal{X}, \ell) = \mathbf{X}^{(\ell)} \in \mathbb{R}^{n_\ell \times n\backslash\ell}$ be the mode-ℓ matricization of $\mathcal{X}$ formed by setting the ℓth mode of $\mathcal{X}$ to the rows of $\mathbf{X}^{(\ell)}$, vectorizing the remaining modes to form the columns of $\mathbf{X}^{(\ell)}$, as in Figure S1.

Tensor products

Up to this point, we have explicitly avoided constructing D-dimensional transforms. In the 2D case, applying a 2D transform is calculated via linear operators, as in (3); generalizing to higher-order tensors requires multilinear operators. Therefore, we introduce the tensor product and its discrete form, which is known as the *Kronecker product*. These products are powerful tools for succinctly describing D-dimensional transforms.

The great utility of the tensor product is that it simultaneously transforms spaces alongside their linear operators. This is the so-called universal property of the tensor

Order-Three Tensors

For simplicity, we briefly review some tensor terminology for the three-way tensor $\mathcal{X} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$. The size of each mode is denoted by n_k, with n_1 being the number of columns, n_2 the number of rows, and n_3 the number of tubes [8]. Video and the time-series recording of matrix-valued signals are common applications for tensors of this form (Figure S1). In videos, the first and second modes of the tensor encode pixel values for each frame, while the third mode indexes the frames in time. We can slice a video tensor to produce different views of the data, as presented in Figure S1.

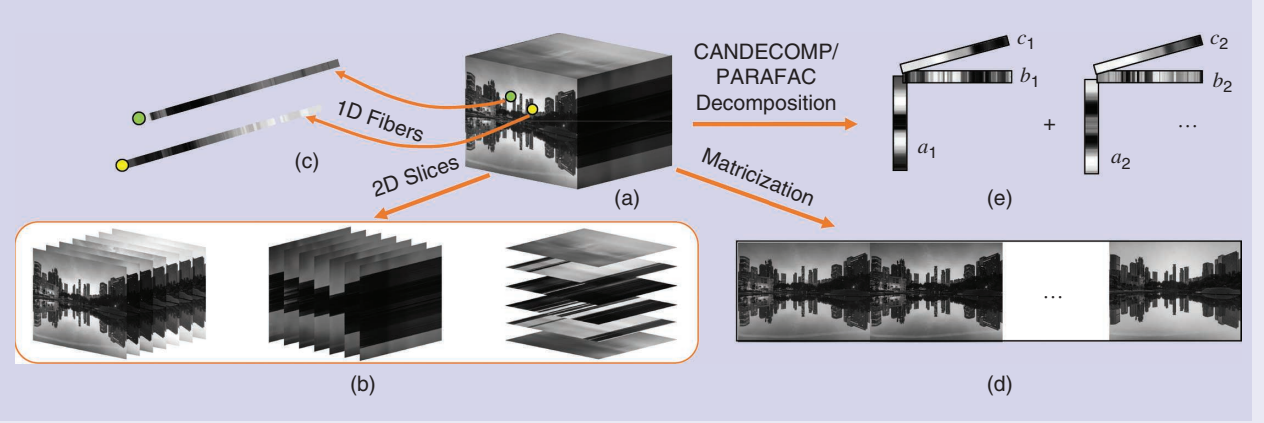

FIGURE S1. Tensor terminology. (a) A time-lapse video is an order-three tensor. (b) Tensor slices (left to right): A frontal slice is the matrix $\mathcal{X}_{::k}$ that is formed by selecting the kth frame of the video. The lateral slice, $\mathcal{X}_{:j:}$, is a matrix (viewable as an image) that shows the time evolution of the jth column of pixels in the input. The horizontal slice, $\mathcal{X}_{i::}$, similarly contains the time evolution of one row of pixels. (c) The 2D indexing of third-order tensors yields a 1D fiber. For example, the tubular fiber $\mathcal{X}_{ij:}$ is an n_3-dimensional time series of the i,jth pixel across all frames; the two tubular fibers correspond to the highlighted pixels in the tensor. (d) Mode 1 matricization concatenates all frontal slices side by side. (e) Canonical decomposition/parallel factor analysis decomposition (CANDECOMP/PARAFAC) is a sum of the rank-one tensors. (Video source: Stock footage provided by peacezxp, downloaded from https://www.videvo.net/video/timelapse-of-klcc-kuala-lumpur/2654/.)

product. In brief, it states that the tensor product, denoted by $\otimes$, of two vector spaces V and W is the unique result of a bilinear map $\varphi : V \times W \to V \otimes W$. The power in φ is that it uniquely factors any bilinear map on $V \times W$ into a linear map on $V \otimes W$. The universal property implies that the tensor product is symmetric: $V \otimes W$ is a canonical isomorphism of $W \otimes V$. Although the tensor product is defined in terms of two vector spaces, it can be repeatedly applied to combine many domains, so we generically refer to it as a *product of many spaces*.

In this article, we are concerned with the tensor product on Hilbert spaces $\mathcal{H}^{(k)}$, $k = 1, \ldots, D$. These metric spaces include both continuous and discrete Euclidean domains from classical signal processing as well as the non-Euclidean vertex domain. Since tensor products on Hilbert spaces produce Hilbert spaces, we can combine time, space, the vertex, and other signal processing domains via the tensor product and remain in a Hilbert space. Under some constraints, an orthonormal basis for the product of D Hilbert spaces is directly admitted by the tensor product of the factor spaces. These properties of the tensor product are the mathematical foundations for the remainder of this tutorial, in which we construct a multiway signal processing framework based on unifying multiple input spaces and their Fourier operators into a single linear representation.

Kronecker products

The Kronecker product produces the matrix of a tensor product with respect to a standard basis and generalizes the outer product of vectors $\mathbf{x}\mathbf{y}^*$ for $\mathbf{x} \in \mathbb{C}^m$ and $\mathbf{y} \in \mathbb{C}^n$. For an analogy, it is common to use the same notation to denote the Kronecker product and the tensor product. The Kronecker product is associative. Consequently, the matrix $\mathbf{M}$ that is the Kronecker product of a sequence of D matrices $\mathbf{M}^{(k)} \in \mathbb{C}^{n_k \times n_k}$ for $k = 1, \ldots, D$ is

$$\begin{aligned}\mathbf{M} = \bigotimes_{k=1}^{D} \mathbf{M}^{(k)} &= \mathbf{M}^{(1)} \otimes \left(\bigotimes_{k=2}^{D} \mathbf{M}^{(k)} \right) = \left(\bigotimes_{k=1}^{D-1} \mathbf{M}^{(k)} \right) \otimes \mathbf{M}^{(D)} \\ &= \mathbf{M}^{(1)} \otimes \cdots \otimes \mathbf{M}^{(D)}. \end{aligned} \tag{5}$$

It is important to note that the Kronecker product is, in general, noncommutative. For brevity, we will apply a decremental Kronecker product using the notation

$$\downarrow \bigotimes_{k=1}^{D} \mathbf{M}^{(k)} = \bigotimes_{k=0}^{D-1} \mathbf{M}^{(D-k)} = \mathbf{M}^{(D)} \otimes \cdots \otimes \mathbf{M}^{(1)}.$$

The Kronecker product has many convenient algebraic properties for computing multidimensional transforms. Vectorization enables one to express bilinear matrix multiplication as a linear transformation

$$\text{vec}(\mathbf{CXB}) = (\mathbf{B}^\top \otimes \mathbf{C})\,\text{vec}(\mathbf{X}), \tag{6}$$

assuming that the dimensions of $\mathbf{C}$, $\mathbf{X}$, and $\mathbf{B}$ are compatible such that $\mathbf{CXB}$ is a valid operation. This identity is a discrete realization of the universal property of tensors and shows that the Kronecker product corresponds to a bilinear operator. We will use this identity to 1) construct multidimensional discrete Fourier bases and 2) decompose multiway algorithms for computational efficiency.

Multiway transforms and filters

We now apply (6) to explicitly construct a 2D GFT. If $\mathbf{\Psi}^{(1)}$ and $\mathbf{\Psi}^{(2)}$ are Fourier bases for graph signals on any two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, a 2D GFT basis is $\mathbf{\Psi}^{(2)} \otimes \mathbf{\Psi}^{(1)}$. This is a single orthonormal basis of dimension $|\mathcal{V}^{(1)}||\mathcal{V}^{(2)}| \times |\mathcal{V}^{(1)}||\mathcal{V}^{(2)}|$, which can be used to describe a 2D graph signal $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$ in the geometry of a single multiway graph by the GFT

$$\hat{\mathbf{x}} = (\mathbf{\Psi}^{(2)} \otimes \mathbf{\Psi}^{(1)})^* \text{vec}(\mathbf{X}).$$

Unlike the DFT, where it is clear that increasing the dimension yields grids, cubes, and hypercubes, interpreting the geometry of $\mathbf{\Psi}^{(2)} \otimes \mathbf{\Psi}^{(1)}$ is less intuitive. For this, we must turn to a graph product.

Product graphs

MWGSP relies on a graph underlying each mode of the given tensor data. The question is: What joint geometry arises from these graphs, and what multilinear operators exist on this joint graph structure? Our approach is to construct a multiway graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$ across the entirety of a data $\mathcal{X}$ as the product graph of a set of factor graphs $G = \{\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(D)}\}$. For example, if $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ contains the results of an n_3 sample longitudinal survey of n_2 genes on a cohort of n_1 patients, then the intramodal relationships of $\mathcal{X}$ are modeled by separate graphs: $\mathcal{G}^{(1)}$, in which each patient is a vertex; $\mathcal{G}^{(2)}$, in which each gene is a vertex; and $\mathcal{G}^{(3)}$, which represents time as a path graph on n_3 vertices. We will use this example throughout this section, although our derivation generalizes to tensors of arbitrary order.

While one could treat matrix-valued slices of $\mathcal{X}$ as signals on each individual graph, we use the graph product to model $\mathcal{X}$ as a single graph signal on $\mathcal{G}$. We begin by constructing $\mathcal{V}$, the vertices of $\mathcal{G}$, which, for all graph products, is performed by assigning a single vertex to every element in the Cartesian product of the factor vertex sets; i.e., $\mathcal{V} = \mathcal{V}^{(1)} \times \ldots \times \mathcal{V}^{(D)}$. Thus, the cardinality of the vertex set of $\mathcal{G}$ is $n = \prod_{k=1}^{D} n_k$. For example, our longitudinal survey will be modeled by the product graph $\mathcal{G}$ on $n = n_1 n_2 n_3$ vertices. As a Cartesian product, the elements $v \in \mathcal{V}$ can be expressed as the tuple $v = (\textit{patient}, \textit{gene}, \textit{time})$. The experimental observation tensor can be modeled as a graph signal $\mathbf{x} = \text{vec}(\mathcal{X})$ in $\mathbb{R}^n$. [We can do this because the vectorization $\text{vec}(\mathcal{X})$ is isomorphic to $\mathcal{X}$, which can be shown using (6).]

Our next step is to learn the topology of $\mathcal{G}$ by mapping the edge sets (weights) of the factor graphs into a single set of product edges (weights) $\mathcal{E}$. There are a variety of graph products, each of which differs from the others only in the construction of this map. We focus on the Cartesian graph product, as it is the most widely employed in multiway algorithms. However, other products, such as the tensor and strong graph products, each induce novel edge topologies that warrant further exploration of MWGSP [7].

Cartesian graph products

We denote the Cartesian product of D graphs as

$$\mathcal{G} = \square_{\ell=1}^{D} \mathcal{G}^{(\ell)} = \mathcal{G}^{(1)} \square \cdots \square \mathcal{G}^{(D)}. \tag{7}$$

The Cartesian graph product is intuitively an exclusive-or product since, for any two vertices

$$\{v = (v^{(1)}, \ldots, v^{(D)}), u = (u^{(1)}, \ldots, u^{(D)})\} \subset \mathcal{V}, \tag{8}$$

the edge (v, u) exists if and only if there exists a single i such that $(v^{(i)}, u^{(i)}) \in \mathcal{E}^{(i)}$ and $v^{(\ell)} = u^{(\ell)}$ for all $\ell \neq i$. In other words, the vertices of $\mathcal{G}$ are connected if and only if, exclusively, one pair of factor vertices is adjacent, and the remaining factor vertices are the same. Figure 1(a) illustrates the generation of an $n_1 \times n_2$ 2D grid graph via the product of two path graphs on n_1 and n_2 vertices.

The Cartesian graph product can induce topological properties, such as regularity, onto a graph. Since the path graph basis is well characterized as a discrete Fourier basis, it is a convenient tool for including Euclidean domains in multiway analysis. For example, we can model time series and longitudinal graph signals as a single vector by using a path graph product. In the case of our gene expression data $\mathcal{X}$, the product of the gene and the patient mode graphs with a path on n_3 vertices; i.e., $\mathcal{G}^{(1)} \square \mathcal{G}^{(2)} \square \mathcal{P}^{n_3}$, models the data by treating the temporal mode as a sequence. One can intuit this operation as copying $\mathcal{G}^{(1)} \square \mathcal{G}^{(2)}$ n_3, times and connecting the edges between each copy.

Product graph matrices

The Kronecker product links graph shift operators on Cartesian product graphs to the corresponding operators on the factors. The Kronecker sum of D matrices $\mathbf{A}^{(k)} \in \mathbb{C}^{n_k \times n_k}$ for $k = 1, \ldots, D$ is

$$\mathbf{A} = \bigoplus_{k=1}^{D} \mathbf{A}^{(k)} = \sum_{k=1}^{D} \mathbf{I}_{n>k} \otimes \mathbf{I}_{n>k},$$
$$\text{where } n_{>k} = \prod_{\ell=k+1}^{D} n_\ell, \quad \text{and } n_{<k} = \prod_{\ell=1}^{k-1} n_\ell.$$

The joint adjacency matrix $\mathbf{A}$ and the graph Laplacian $\mathcal{L}$ are constructed by the Kronecker sum of their corresponding factor graph matrices. The eigensystem of a Kronecker

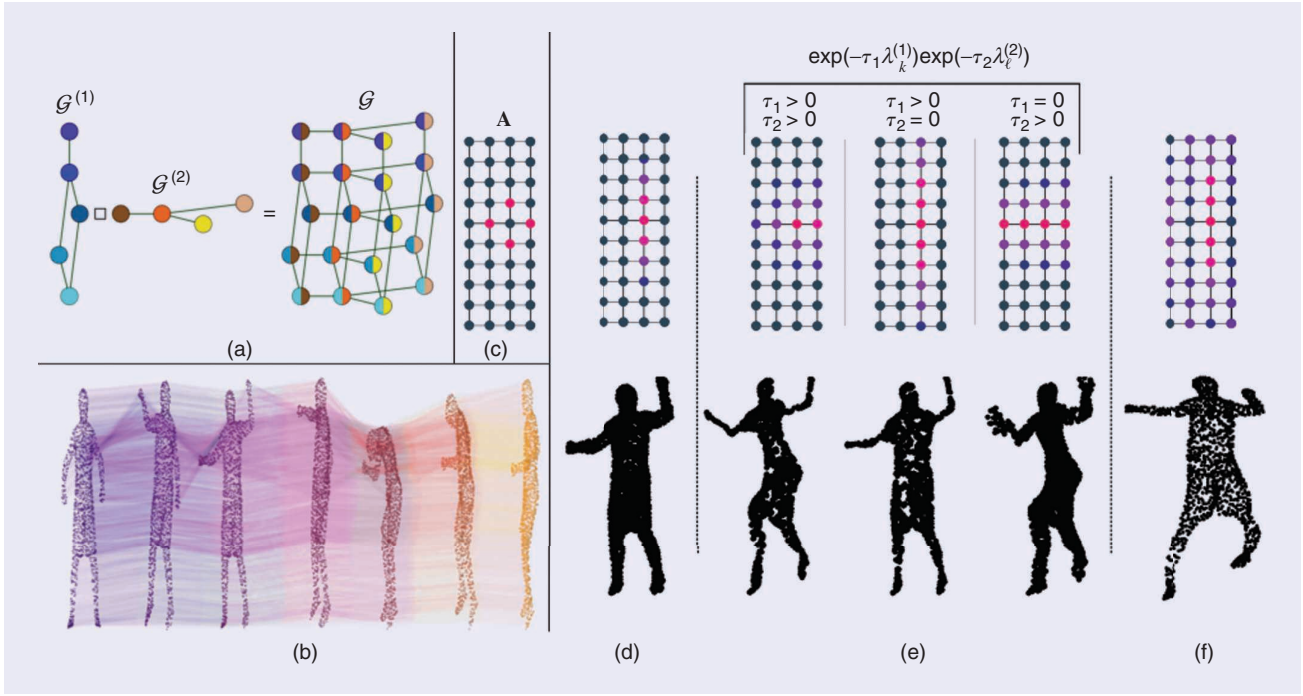

FIGURE 1. Multiway graphs, signals, and spectral filters. (a) The Cartesian graph product generates a copy of $\mathcal{G}^{(1)}$ at each vertex of $\mathcal{G}^{(2)}$. (b) A multiway graph formed from a dynamic mesh [2]. This T–V graph (purple $= t_0$; yellow $= t_7$) connects each point in the mesh to its counterpart in adjacent frames. The temporal evolution of the 3D coordinates is a graph signal on this graph. (c) A single column of the joint adjacency matrix of a 2D grid shifts signals to their neighbors. (d)–(f) Multiway filtering. A multiway signal on a 2D grid (top). This signal can be decomposed into an impulse and a smooth signal; thus, it is band-limited along one way of the grid. A frame of the dancer mesh (bottom). (e) A separable diffusion filter is the product of domain-specific heat kernels. Separable filters can filter along both axes in unison (left) or each axis independently (middle/right). (f) A nonseparable filter. For the dynamic mesh, filtering along only one mode reveals either the skeleton structure $(\tau_2 = 0)$ or the averaged (blurred) dynamics of the figure $(\tau_1 = 0)$, while joint separable and nonseparable filtering reveals joint dependencies.

sum is generated by the pairwise sum of the eigenvalues of its factors and the tensor product of the factor eigenbases [22, Th. 4.4.5]. Thus, the Fourier basis $\boldsymbol{\Psi}$ for the product graph $\mathcal{G}$ is immediate from the factors. For $k = 1, \ldots, D$, let $(\lambda_{\ell_k}, \psi_{\ell_k})$ be the ℓ_kth eigenpair of $\boldsymbol{\mathcal{L}}^{(k)}$ for $0 \le \ell_k \le n_k - 1$. Then, let $I_\ell = (\ell_1, \ldots, \ell_D) \in [n_1] \times \ldots \times [n_D]$ be a multi-index to the ℓth eigenpair of $\boldsymbol{\mathcal{L}}$. The product graph Fourier basis is then

$$(\lambda_{I_\ell}, \psi_{I_\ell}) = \left(\sum_{k=1}^{D} \lambda_{\ell_k}^{(k)}, \downarrow \bigotimes_{k=1}^{D} \psi_{\ell_k}^{(k)} \right). \tag{9}$$

Thus, the MWGFT of a multiway graph signal $\mathcal{X}$ is

$$\hat{\mathbf{x}} = \boldsymbol{\Psi}^* \operatorname{vec}(\mathcal{X}) = \left(\downarrow \bigotimes_{k=1}^{D} \boldsymbol{\Psi}^{(k)} \right)^* \operatorname{vec}(\mathcal{X}). \tag{10}$$

This formulation includes applying a single-way transform along one mode of the tensor; for example, $\mathrm{DFT}_{n_1}\{\mathcal{X}\} = \left(\downarrow \bigotimes_{k=2}^{D} \mathbf{I}_{n_k} \otimes \mathbf{U}_{n_1} \right)^{\mathrm{T}} \operatorname{vec}(\mathcal{X})$ applies the DFT along the first mode of the tensor (see "Multiway Signal Compression").

Efficient MWGSP by graph factorization

On the surface, the computational cost of an MWGFT (and MWGSP, in general) seems high, as multiway product graphs are often much larger than their individual factors; the cardinality of the product vertex set is the product of the number of vertices in each factor. However, the product graph structure actually yields efficient algorithms. With small adjustments to fundamental operations, such as matrix multiplication, in the best case, one can effectively reduce the computational burden of an order-D tensor with $n = \prod_{\ell=1}^{D} n_\ell$ total elements to a sequence of problems on $n^{(1/D)}$ elements. The computational strategy is to apply (6) and its order-D generalization to avoid storing and computing large product graph operators. We introduce the order-D form of (6) via an algorithm. Given a sequence of operators $\mathbf{M}^{(\ell)} \in \mathbb{R}^{n_\ell}$, $\ell = 1, \ldots, D$, an efficient algorithm for computing $y = \bigotimes_{\ell=1}^{D} \mathbf{M}^{(\ell)} \operatorname{vec}(\mathcal{X})$ proceeds by applying each $\mathbf{M}^{(\ell)}$ to the corresponding mode-wise matricization of $\mathcal{X}$. Algorithm 1 presents pseudocode for computing this product.

As a sequential product of an $n_\ell \times n_\ell$ matrix with an $n_\ell \times n \backslash \ell$ matrix, this method can dramatically improve the cost of algorithms that depend on matrix multiplication. Further, the number of operations depends only on computations across smaller factor matrices, enabling one to perform computations on the product graph without computing and storing expensive operators. For example, consider the computational cost of applying an MWGFT for a product graph $\mathcal{G}$ on $n = \Pi_{\ell=1}^{D} n_\ell$ nodes. In the worst case, Algorithm 1 is as fast as directly computing (10). However, in the best-case scenario, $n_\ell = \sqrt[D]{n}$ for all $\ell = 1, \ldots, D$, and computing D graph Fourier bases of

A key motivation for MWGSP is the capability of encoding multiway data in a compact way. Transforms with good energy compactness summarize the data well and can be used to construct efficient regularizers for regression problems. Figure S2 demonstrates energy compression in four data sets. The dancer mesh [2] shown in Figure 1(b) couples $n_1 = 1{,}502$ points to temporal evolution across $n_2 = 570$ time steps. The Molene weather data set [23] ($n_1 = 32$ weather stations measuring temperatures across $n_2 = 24$ hours across $n_3 = 30$ days) couples a spatially determined graph to two temporal scales (hours and days). The time-lapse video [2] couples a 2D spatial grid (492 × 853 pixels) to a temporal axis (602 time steps), while the hyperspectral data set [24] couples a 2D spatial grid (145 × 145 pixels) to 200 spectral bandwidths (treated as a graph). All graphs were constructed using k nearest neighbors with weighted edges that were set using a Gaussian kernel on the matricized modes of the tensor.

To measure the energy compactness, we compute the relevant transforms among the GFT, DFT (temporal axis), joint time–vertex Fourier transform (JFT), 2D DFT (spatial grid), 3D DFT (spatial grid + temporal axis), and MWGFT (all tensor modes) for each data set. We replace the spectrum coefficients with magnitudes smaller than the pth percentile with zeros and perform the corresponding inverse transform on the resulting coefficients. The normalized compression error is computed from the signal reconstructed after thresholding the values of the transforms below the pth percentile, which is denoted by $\mathcal{X}_p$ and given by $\|\operatorname{vec}(\mathcal{X}_p - \mathcal{X})\|_2 / \|\operatorname{vec}(\mathcal{X})\|_2$. MWGFT achieves the best compactness in all data sets, providing the insight that there are advantages to treating classical domains (time and space) as themselves lying on graphs.

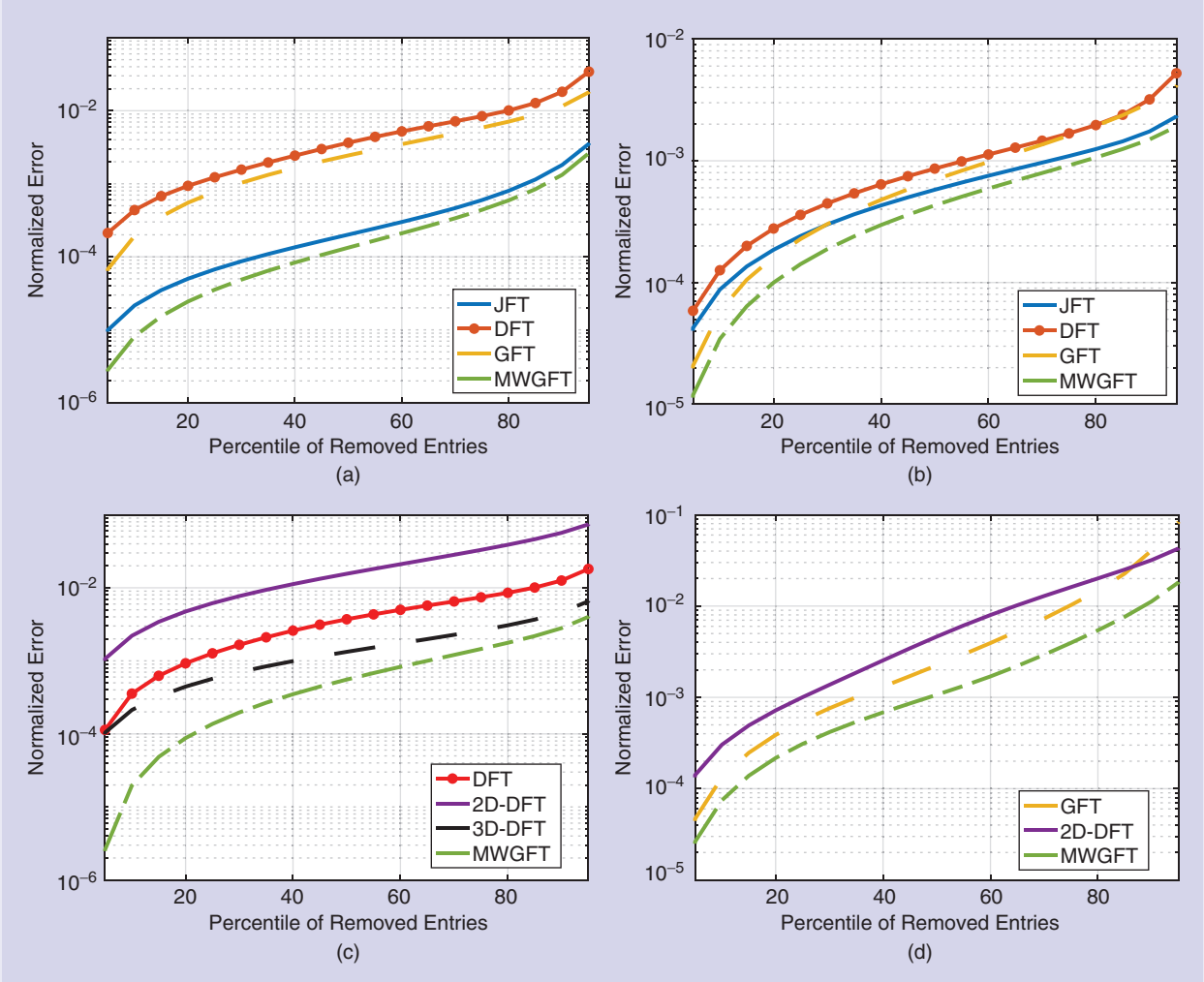

FIGURE S2. The compactness of single- and multiway transforms for different data sets: (a) the dancer mesh (Figure 1), (b) the Molene weather data set, (c) a time-lapse video (Figure S1), and (d) an Airborne Visible/Infrared Imaging Spectrometer Indiana pines hyperspectral image.

size $\sqrt[D]{n} \times \sqrt[D]{n}$ requires $O(n^{3/D})$ operations. To compute an MWGFT using the factor bases, we use $\Psi^{(\ell)}$ as the sequence of operators in Algorithm 1, which costs $O(Dn^{1/D+1})$ operations. This improves the standard GFT, which costs $O(n^3)$ operations to obtain an eigenbasis and $O(n^2)$ operations to apply. For example, when $D = 3$ and $n_1 = n_2 = n_3 = \sqrt[3]{n}$, we obtain an asymptotically linear factorization of a graph Fourier basis for $\mathcal{G}$, and the corresponding MWGFT can be applied in $O(3n^{1/3+1})$ operations.

Algorithm 1. *D*-tensor multilinear transformations.

1: Initialize $\mathcal{Y} = \mathcal{X}$
2: **for** $\ell = 1, \ldots, D$ **do**
3: Matricize: $\mathbf{Y}^{(\ell)} = \text{mat}(\mathcal{Y}, \ell)$
4: Factor update: $\mathbf{Y}^{(\ell)} = \mathbf{M}^{(\ell)^\top}\mathbf{Y}^{(\ell)}$
5: Reform tensor: $\mathcal{Y} = \text{ten}(\mathbf{Y}^{(\ell)}, \ell, \{n_1, \ldots, n_D\})$
6: **end for**
7: Vectorization: $\mathbf{y} = \text{vec}(\mathcal{Y})$.

Edge density

The graph edge density impacts the scalability of signal processing algorithms for multiway data. Matrix equations can be efficiently solved by iteratively computing sparse matrix–vector products. The computational complexity of such algorithms, which include fundamental techniques, such as Krylov subspace methods and polynomial approximation, typically depend linearly on the number of edges in the graphs, e.g., [2], [5], and [25]. This dependency suggests using the sparsest possible graph that still captures the main similarity structure along each mode. Indeed, a common strategy is to construct sparse graph Laplacian matrices [25] or edge-incidence matrices [5] using k-nearest-neighbor graphs that produce edge sets whose cardinality is linear in the number of nodes. Yet, given a sparse factor graph, there is no guarantee that the product will be sparse. Thus, major efficiency gains for multiway algorithms can be made by replacing iterative matrix–vector multiplications (both sparse and dense) with a sequence of factor graph sparse matrix–vector multiplications through Algorithm 1.

Three immediate applications for such a factorization are multiway filter approximations (see, e.g., [2]), compressive spectral clustering [26], and fast GFT [27]. We detail the former while briefly describing future directions for the latter. For filtering, one could spectrally define and exactly compute a multiway product graph filter (see "Multiway Filters") by using the MWGSP techniques described in the previous section. Yet, Chebyshev approximations [2] are an efficient, robust, and accurate technique for approximate filtering. These approaches

Multiway Filters

It is natural to define spectral filters for multiway graph signals on the product $\mathcal{G}$ as a function across the product graph eigenvalues $h: \Lambda \mapsto \mathbb{R}$ as if they are traditional spectrally defined GSP filters. Since these functions operate on the product eigenvalues, they directly consider the edge topology induced by a particular choice of product. Yet, it is feasible to develop filters for multiway graph signals on $\mathcal{G}$ that are defined by multivariate functions $h: \Lambda^{(1)} \times \cdots \times \Lambda^{(D)} \mapsto \mathbb{R}$. These multivariate filters are split into two classes: separable and nonseparable.

Separable filters have multivariate response functions that can be written as the product of separate univariate functions. In the $D = 2$ case, a separable filter for the product graph could be written as $(\mathbf{H}^{(2)} \otimes \mathbf{H}^{(1)})\mathbf{x}$, in which $\mathbf{H}^{(1)} = \mathbf{\Psi}^{(1)} h^{(1)}(\mathbf{\Lambda}^{(1)})\mathbf{\Psi}^{(1)^*}$ and $\mathbf{H}^{(2)} = \mathbf{\Psi}^{(2)} h^{(2)}(\mathbf{\Lambda}^{(2)})\mathbf{\Psi}^{(2)^*}$. Since this Kronecker product is permutation-equivalent, we can treat its operation as an order-independent unimodal filtering of $\mathbf{x}$ (6). If $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ are both filters defined in a Laplacian eigenbasis of their respective factor graph, then the tensor product $(\mathbf{H}^{(2)} \otimes \mathbf{H}^{(1)})$ is also diagonalized by the product eigenbasis. Thus, this filter is merely a reweighting of the product graph eigenbasis. In Figure 1(e), we demonstrate the application of a product of mode-wise heat filters to a graph signal on a grid [Figure 1(d), top] and to a T–V signal that is a dynamic mesh [Figure 1(d), bottom]. While there is a choice of τ_1 and τ_2 such that certain regions of this filter can be computed from a heat kernel on the Cartesian product graph spectrum, such an approach abandons the flexibility of bilinear filtering. By separability, each mode can be analyzed independently of the others by setting the appropriate τ_k to zero. This enables independently analyzing a joint signal along each mode, for example, by filtering out high-frequency structures along one domain while preserving the frequency content of the other mode. A D-way separable filter applied to $\mathbf{x} = \text{vec}(\mathcal{X})$ is given by

$$\tilde{\mathbf{x}} = \mathbf{\Psi} h\Big(\bigtimes_{k=1}^{D} \Lambda^{(k)}\Big)\mathbf{\Psi}^* \mathbf{x},$$

where $h(\times_{k=1}^{D}\Lambda^{(k)})$ is a diagonal matrix whose elements are given by $\Pi_{k=1}^{D} h^{(k)}(\lambda_{\ell_k})$, i.e., the product of separate spectral functions $h^{(k)}$ for each factor graph $\mathcal{G}^{(k)}$.

Nonseparable filters cannot be designed from separate univariate filters on each mode. This class of filters encompasses a broad group of functions that includes many filters defined in terms of the product graph eigenvalues as well as multivariate functions [Figure 1(f)]. Indeed, the authors of [2] find that one cannot, in general, describe partial differential equations that define diffusion, wave, and disease propagation with separable filters, as the relationship between frequencies is not independent.

approximate spectrally defined filters by applying a recurrently defined weighted matrix–vector multiplication. Efficient multiway Chebyshev approximation leverages the Kronecker sum definition for product graph Laplacians $\mathcal{L}$. That is, by noting that $\mathcal{L}\mathbf{x} = \Sigma_{k=1}^{D}(\mathbf{I}_{n_{>k}} \otimes \mathcal{L}^{(k)} \otimes \mathbf{I}_{n_{<k}})\mathbf{x}$ is equivalent to computing

$$(\mathbf{I}_{n_{>1}} \otimes \mathcal{L}^{(1)})\mathbf{x} + (\mathbf{I}_{n_{>2}} \otimes \mathcal{L}^{(2)} \otimes \mathbf{I}_{n_1})\mathbf{x} + \ldots + (\mathcal{L}^{(D)}\mathbf{I}_{n_{<D}})\mathbf{x},$$

it is clear that Chebyshev approximations of functions on $\mathcal{L}$ (such as spectral graph wavelets) can be written as a sum of sparse matrix vector multiplications; all operations are now dominated by the densest factor graph.

The efficiency of this approach cannot be understated, as it facilitates many algorithms, including the compressive spectral algorithm [26]. Indeed, it is increasingly common to estimate geometric and spectral qualities of the graph Laplacian by applying ideal filter approximations for eigencounting and coherence estimation. Finally, factor graph sparsity and Algorithm 1 could be combined with recently proposed approaches for approximate orthogonal decompositions [27] to construct a fast-product GFT. This algorithm would admit striking similarities to the classical fast Fourier transform (FFT).

MWGSP frameworks

Here, we highlight two recent multiway frameworks: the T–V framework [2] and generalized GSP [28].

T–V framework

The joint T–V framework [2], [3], [23] arose to address the limitations of GSP in analyzing dynamic data on graphs. This required generalizing harmonic analysis to a coupled time-graph setting by connecting a regular axis (time) to an arbitrary graph. The central application of these techniques is to analyze graph signals that are time-varying, for example, a time series that resides on a sensor graph. Each time point of this series is itself a graph signal, while each vertex on the graph maps to a time series of T samples. This enables learning covariate structures from T–V signals, which are bivariate functions on the vertex and the time domain. Such sequences of graph signals are commonly collected longitudinally through sensor networks, video, health data, and social networks. The JFT [2] for a T–V signal $\mathbf{X} \in \mathbb{R}^{|V| \times T}$ is defined as

$$\text{JFT}\{\mathbf{X}\} = \mathbf{\Psi}^{*}\mathbf{X}\mathbf{U}_T \quad \text{or} \quad \text{JFT}\{\text{vec}(\mathbf{X})\} = (\bar{\mathbf{U}}_T \otimes \mathbf{\Psi})^{*}\text{vec}(\mathbf{X}),$$

such that the multiway Fourier transform of a T–V signal is a tensor product of the DFT basis with a GFT basis (see Figure S2). Consequently, the JFT admits a fast transform in which one first performs an FFT along the time mode of the data before taking the GFT of the result, thus requiring only one Laplacian diagonalizaiton.

Including the DFT basis in this framework immediately admits novel joint T–V structures that are based on classical tools, such as variational norms that combine classical variation with graph variation [2]. For efficient filter analysis, they also propose an FFT and a Chebyshev-based algorithm for computing fast T–V filters, which applies to both separable and nonseparable filters; see an example of T–V filtering in Figure 1. Finally, overcomplete dictionary representations are constructed as a tensor-like composition of graph spectral dictionaries with classical a short-time Fourier transform and wavelet frames. These joint dictionaries can be constructed to form frames, enabling the analysis and manipulation of data in terms of time-frequency–vertex-frequency localized atoms. T–V spectral filtering was also introduced in [3], in addition to a T–V Kalman filter, with both batch and online function estimators. Further works have integrated ideas from classical signal processing, such as stationarity to graph and T–V signals [23], [29], [30]. Thus, recent developments in the T–V framework can serve as a road map for the future development of general MWGSP methods.

Generalized GSP

Another recent development is that of the generalized GSP [28] framework, which extends the notions of MWGSP to arbitrary, nongraphical geometries. Generalized GSP facilitates the multivariate signal processing of interesting signals in which at least one domain lacks a discrete geometry. This framework recognizes that the key intuition of GSP is the utility of irregular, non-Euclidean geometries for analyzing signals. However, where GSP techniques axiomatize a finite relational structure encoded by a graph shift operator, generalized GSP extends classical Fourier analogies to arbitrary Hilbert spaces (i.e., complete inner-product spaces) $\mathcal{H} \in H$ equipped with a compact, self-adjoint operator A. This broad class of geometries contains GSP as the standard space of square summable graph signals; i.e., $L^2(V) = \{f : V \mapsto \mathbb{C}, \|f\|_2 < \infty\}$ is itself a Hilbert space.

The geometries and corresponding signals that can be induced by generalized GSP offer an intriguing juxtaposition of continuous and discrete topologies. As an example, consider the tensor product of a graph $\mathcal{G}$ with the space of square integrable functions on an interval, e.g., $\mathcal{G} \otimes L^2([-1,1])$. Graph signals in this space map each vertex to an L^2 function. Conversely, L^2 functions can be mapped to specific vertices. To generate a Fourier basis for the product space, one simply takes the tensor product of the factor space eigenbases. This is a promising future direction for MWGSP, as it implies that one can, for instance, combine graph Fourier bases with generalized Fourier bases for innovative signal representations.

The authors of [11] proposed an early example of generalized GSP, though under a different name. This work modeled videos and collections of related matrices as matrix-valued graph signals using matrix convolutional networks. The authors aimed to solve the challenging missing-data problem of node undersampling: some matrix slices from the networks are completely unobserved. When matrices have a low-rank GFT, the network's graph structure enables the recovery of missing slices. In light of the development of generalized GSP, it is clear that [11] proposed an algorithm for the denoising of multiway signals on $\mathcal{G} \otimes \mathbb{R}^{n_1 \times n_2}$.

Signal processing on multiway graphs

In the previous section, we focused on signal processing through the lens of harmonic analysis, using the graph Laplacian to analyze data in the spectral domain. In this section, we focus on signal modeling and recovery in the multiway setting through the lens of optimization, where the graph Laplacian serves the role of imposing signal smoothness. Including graph structures along the modes of multiway matrices and higher-order tensors has led to more robust and efficient approaches for denoising, matrix completion, and inpainting; collaborative filtering; recommendation systems; biclustering; factorization; and dictionary learning [4], [10], [11], [16], [18], [21]. We begin with dual-graph modeling in the matrix setting and then extend to the higher-order tensor setting. In the tensor setting, we review the use of multiway graph regularization in tensor factorization methods and in a complementary fashion using tensor factorization in signal modeling and recovering to make graph regularization computationally tractable.

Signal processing on dual graphs

The quadratic form of the graph Laplacian of a graph $\mathcal{G}$

$$\mathbf{f}^\top \mathcal{L}\mathbf{f} = \sum_{(i,j)\in\varepsilon} \mathbf{W}_{i,j}(\mathbf{f}_i - \mathbf{f}_j)^2, \tag{11}$$

quantifies the smoothness of a signal $\mathbf{f}$ with respect to the graph, where the smoother a signal is, the smaller the value. Consequently, the typical model in the multiway signal recovery setting is to add dual row–column graph regularizers of the form $\gamma_r \mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_r \mathbf{X}) + \gamma_c \mathrm{Tr}(\mathbf{X}\mathcal{L}_c \mathbf{X}^\top)$ to classical problem formulations; such a regularization incentivizes the recovered signal to be smooth with respect to the underlying data graphs (11). The matrices $\mathcal{L}_r$ and $\mathcal{L}_c$ denote the graph Laplacians on the rows and columns of $\mathbf{X}$, respectively, and the nonnegative tuning parameters γ_r and γ_c trade off data fitting for smoothness with respect to the row and column geometries encoded in $\mathcal{L}_r$ and $\mathcal{L}_c$, respectively. The choice of underlying graphs consequently greatly influences the quality of signal reconstruction. For details on how to construct a graph, see "Graph Construction."

Graph Construction

A question that arises in graph-based methods is how to construct the graphs themselves. In some applications, e.g., social and citation networks, a graph is known a priori. In transportation and communication networks, vertices represent physical locations (traffic intersections) and sensors (routers in a Wi-Fi network), and edges encode connected locations. In other settings, there is no a priori graph, and the topology must be learned from the data. We describe common strategies and challenges.

Data-driven graphs

One of the most popular ways to construct a graph is from the data itself, for example, using a k-nearest-neighbor graph with Gaussian kernel weights. For example, in our simulations, if rows i and j are $k = 7$ nearest neighbors, their weight is $\mathbf{W}^{(1)}_{i,j} = \exp\{-\|\mathbf{X}_{i\cdot} - \mathbf{X}_{j\cdot}\|_2^2/\sigma\}$ with kernel bandwidth σ. Otherwise, their weight is zero. One difficulty that arises is that in the presence of noise, outliers, and missing entries, constructing a graph from the data yields a corrupted graph. Figure 2(b) compares a "noisy" graph constructed from the missing data to an "oracle" graph constructed from the original complete data. The noisy graph along the images $(\mathbf{A}^{(1)})$ connects images of different people together, while the noisy feature graph $(\mathbf{A}^{(2)})$ loses the local pixel geometry. The results in Figure 2(a) demonstrate that, for a higher percentage of the missing values, the noisy graph degrades the performance compared to the oracle graph.

Graphs from side information

Supplementary information can be leveraged to define the similarity structure among rows and columns for the purpose of graph construction. In some cases, there may be a natural geometry that easily translates into similarity graphs for rows and columns. For example, in [23], the authors constructed a graph among weather stations by using the stations' physical coordinates. In other cases, various supplemental data sets may be leveraged to provide similarity structure among rows and columns. As an example regarding music recommendation systems, in [36], the authors used a publicly available playlist categorization as well as summary statistics extracted from an audio signal to construct a graph for estimating a latent association matrix between playlists and songs.

Graph learning

In [16] and [19], the graphs on the feature space are learned alongside the signal by minimizing over $\mathcal{L}$, in addition to the signal recovery in the optimization problem. For a detailed review, see [37].

Dynamically varying graphs

Graphs may not be static, presenting a current challenge in GSP. This is especially acute in T–V frameworks, which admit time as one of components in the analysis. The difficulties include determining how to identify when a graph needs to be updated, i.e., when the underlying topology has changed. The task of accounting for dynamically varying graphs also poses computational questions, namely, finding computationally efficient ways to update graphs within the processing framework that will minimally spawn artifacts at transitions.

Table 1 presents formulations of these different algorithms; multiple extensions and other methods exist in the literature. For the T–V framework [2], the graph on the columns is a temporal graph modeled explicitly with a ring graph Laplacian $\mathcal{L}_T$. The mapping $\mathcal{P}_\Omega$ is a projection operator on the set of observed entries Θ in missing-data scenarios. The methods may differ in their fidelity term, minimizing the Frobenius norm for denoising and the one-norm to impart robustness to outliers [25], and several methods assume a low-rank structure, either with a nuclear norm penalty [31] or with an explicit low-rank factorization of the data matrix $\mathbf{Y}$ as $\mathbf{DX}$, sometimes with additional constraints on the factor matrices (nonnegativity [33] and sparsity [16]). A few methods aim to solve a matrix completion problem (see Figure 2). Finally, while most instances of graph regularization rely on the quadratic penalty term $\mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_r \mathbf{X}) = \Sigma_{(i,j)\in\mathcal{E}_r} w_{i,j} \| \mathbf{X}_{i\cdot} - \mathbf{X}_{j\cdot} \|_2^2$, the biclustering formulation in [5] and [32] employs a penalty that is either linear in the l_2-norm or concave and continuously differentiable, relying on the mapping $\Omega(\| \mathbf{X}_{i\cdot} - \mathbf{X}_{j\cdot} \|_2)$. The motivation there is that convex penalties, when Ω is either linear or quadratic, do not introduce enough smoothing for small differences and too much smoothing for large differences, resulting in poorer clustering results.

Typically, an alternating optimization algorithm is used to solve the various problems in Table 1. The T–V regularization problem is the only one with a closed-form solution given by a joint nonseparable low-pass filter (generalizing Tikhonov regularization to the T–V case). The graph DNMF [33] relies on an alternating optimization scheme for the nonnegative factor matrices. Other solutions are computed with proximal methods, such as the alternating-direction method of multipliers to handle multiple regularization terms via variable splitting. Dual-graph regularized approaches have been shown to consistently outperform their nonregularized and single-graph

Table 1. Multiway graph regularization formulations.

	Fidelity Term	Graph Regularizers	Additional Constraints and Regularizers
MCG [31]	$\| \mathcal{P}_\Theta(\mathbf{Y}-\mathbf{X}) \|_F^2$	$\gamma_r \mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_r \mathbf{X}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_c \mathbf{X}^\top)$	$\gamma_n \| \mathbf{X} \|_*$
CFGI [4]	$\| \mathcal{P}_\Theta(\mathbf{Y}-\mathbf{DX}) \|_F^2$	$\gamma(\mathrm{Tr}(\mathbf{D}^\top \mathcal{L}_r \mathbf{D}) + \mathrm{Tr}(\mathbf{X} \mathcal{L}_c \mathbf{X}^\top))$	$\alpha \| \mathbf{D} \|_F^2 + \beta \| \mathbf{X} \|_F^2$
DGRDL [16]	$\| \mathbf{Y}-\mathbf{DX} \|_F^2$	$\gamma_r \mathrm{Tr}(\mathbf{D}^\top \mathcal{L}_r \mathbf{D}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_c \mathbf{X}^\top)$	$\| \mathbf{x}_i \|_0$
T–V regularization [2]	$\| \mathbf{Y}-\mathbf{X} \|_F^2$	$\gamma_r \mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_G \mathbf{X}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_T \mathbf{X}^\top)$	
T–V inpainting [2]	$\| \mathcal{P}_\Theta(\mathbf{Y}-\mathbf{X}) \|_F^2$	$\gamma_r \mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_G \mathbf{X}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_T \mathbf{X}^\top)$	
Cvx biclustering [5]	$\| \mathbf{Y}-\mathbf{X} \|_F^2$	$\gamma_r \sum_{(i,j)\in\mathcal{E}_r} w_{i,j} \| \mathbf{X}_{i\cdot} - \mathbf{X}_{j\cdot} \|_2 + \gamma_c \sum_{(i,j)\in\mathcal{E}_c} \tilde{w}_{i,j} \| \mathbf{X}_{\cdot i} - \mathbf{X}_{\cdot j} \|_2$	
Comani missing [32]	$\| \mathcal{P}_\Theta(\mathbf{Y}-\mathbf{X}) \|_F^2$	$\gamma_r \sum_{(i,j)\in\mathcal{E}_r} \Omega(\| \mathbf{X}_{i\cdot} - \mathbf{X}_{j\cdot} \|_2) + \gamma_c \sum_{(i,j)\in\mathcal{E}_c} \Omega(\| \mathbf{X}_{\cdot i} - \mathbf{X}_{\cdot j} \|_2) \|_2$	
FRPCAG [25]	$\| \mathbf{Y}-\mathbf{X} \|_1$	$\gamma_r \mathrm{Tr}(\mathbf{X}^\top \mathcal{L}_r \mathbf{X}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_c \mathbf{X}^\top)$	
DNMF [33]	$\| \mathbf{Y}-\mathbf{DX} \|_F^2$	$\gamma_r \mathrm{Tr}(\mathbf{D}^\top \mathcal{L}_r \mathbf{D}) + \gamma_c \mathrm{Tr}(\mathbf{X} \mathcal{L}_c \mathbf{X}^\top)$	$\mathbf{D} \geq 0, \mathbf{X} \geq 0$

CFGI: collaborative filtering with graph information; Cvx: convex; DGDRL: dual graph regularized dictionary learning; DNMF: dual regularization nonnegative matrix factorization; FRPCAG: fast robust PCA on graphs; MCG: matrix completion on graphs.

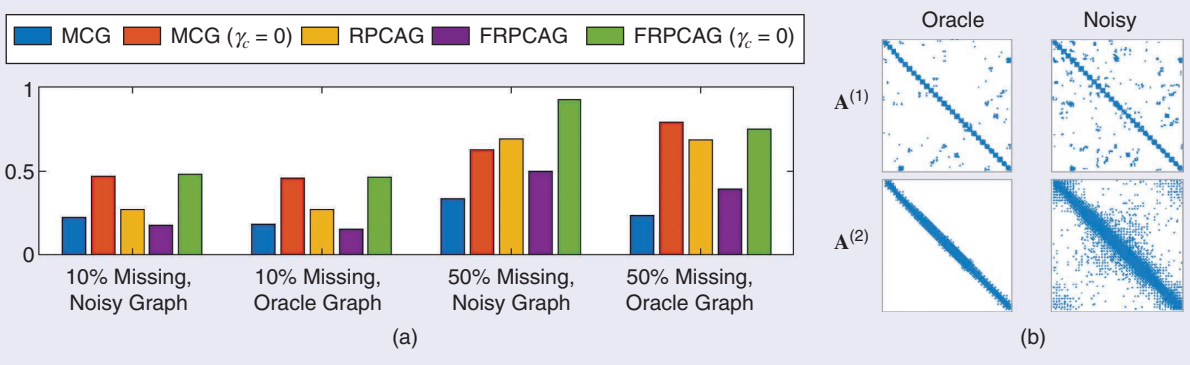

FIGURE 2. Matrix completion on the Olivetti Research Laboratory data set. (a) The relative error for 10 and 50% missing values, using noisy and oracle graphs. (b) The adjacency matrix of row $\mathbf{A}_r$ and column $\mathbf{A}_c$ graphs for complete data ("oracle") and 50% missing data ("noisy").

regularized counterparts across a wide range of applications and domains.

In Figure 2(a), we compare several approaches for matrix completion [25], [31], [34] with single- and multiway graph regularization on the ORL data set with 10 or 50% of the entries missing at random. The ORL [35] data set consists of 300 images of faces (30 people, with 10 images per person), which are flattened into 2,576 features. We used a row graph that connects similar images together and a column graph that ignores the natural 2D grid geometry and, instead, considers a wider geometry in the image plane. To set γ_r, γ_c, we ran each method for a range of values and selected the result with the best performance. For a comparison with single-way graph regularization, we also set $\gamma_c = 0$ in MCG [31] and FRPCAG [34] to ignore the graph on the feature (column) space. In general, γ_r and γ_c induce row and column smoothness at different levels, and their choice should be driven by the tradeoff in the smoothness of the data along the two modes and the aspect ratio of the matrix, or the decision should be informed by cross validation. We report the relative reconstruction error on the missing values, averaged across 10 realizations. The multiway graph regularized approaches outperformed their corresponding single-way versions ($\gamma_c = 0$) in all cases. Both FRPCAG and MCG always outperformed RPCAG, a single-way graph regularized method.

This method can dramatically improve the cost of algorithms that depend on matrix multiplication.

Tensor processing on graphs

A challenge of many well-studied problems in signal processing and machine learning is that algorithm complexity typically grows exponentially when one considers tensors with three or more modes. Early multiway data analysis approaches flattened data tensors to matrices and then applied classical two-way analysis techniques. Flattening, however, obscures higher-order patterns and interactions between the different modes of the data. Thus, multilinear tensor decompositions have been the main workhorse in tensor signal processing and data analysis, generalizing the notion of matrix factorizations to higher-order tensors, and they have become common in applications such as hyperspectral imaging and biomedical imaging.

While there is no single generalization of a spectral decomposition for tensors, the two most common tensor decompositions are the CANDECOMP/PARAFAC (CP) (see Figure S1) and the Tucker decomposition [8]. Just as the singular-value decomposition can be used to construct a lower-dimensional approximation to a data matrix, finding a coupled pair of lower-dimensional subspaces for the rows and columns, these two decompositions can be used to construct lower-dimensional approximations to a D-way tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_D}$. Under mild conditions, the CP decomposition, which approximates $\mathcal{X}$ by a sum of rank-one tensors, is unique up to the scaling and the permutations of the columns of its factor matrices [8], but the CP factor matrices typically cannot be guaranteed to have orthogonal columns. The Tucker decomposition permits orthonormal factor matrices but, in general, fails to have unique representations [8].

Much of the multiway literature has focused on improving and developing new tensor factorizations. Graph-based regularizations along modes of the tensor are proving to be versatile for developing robust tensor and low-rank decompositions [12]–[14] as well as new approaches to problems in higher-order data processing, such as tensor completion, data imputation, recommendation systems, feature selection, anomaly detection, and coclustering, which is a generalization of biclustering to tensors [10], [11], [17]–[20]. The generalization of these problems to tensors incurs a higher computational cost than the equivalent matrix problems. Thus, multiway graph-regularized formulations typically combine a low-rank tensor factorization with graph-based regularization along the rows of the factor matrices; for example, [18] and [20] rely on a CP decomposition, while [13] employs a Tucker decomposition. In [38], a Tucker decomposition is used within MWGSP to construct wavelets on multislice graphs through a two-stage approach.

An example of combining tensor decompositions with graph regularization is the following "low-rank + sparse" model for anomaly detection in Internet traffic data [20]:

$$\min_{\mathcal{X}, \mathcal{E}\{\mathbf{A}^{(i)}\}_i} \|(\mathcal{Y} - \mathcal{E}) - \mathcal{X}\|_{\mathrm{F}}^2 + \sum_{i=1}^{d} \gamma_i \mathrm{Tr}((\mathbf{A}^{(i)})^{\top} \mathcal{L}_i \mathbf{A}^{(i)})$$
$$\text{s.t.} \quad \mathcal{X} = \sum_{i=1}^{R} \mathbf{a}_i^{(1)} \circ \mathbf{a}_i^{(2)} \circ \mathbf{a}_i^{(3)}, \quad \|\mathcal{E}\|_0 \leq \epsilon, \tag{12}$$

where $\mathcal{Y}$ is a data tensor and $\mathcal{E}$ is the tensor of sparse outliers. The equality constraint on $\mathcal{X}$ requires that $\mathcal{X}$ has a rank-R CP decomposition, where $\mathbf{a}_i^{(d)}$ is the ith column of the dth factor matrix $\mathbf{A}^{(d)} \in \mathbb{R}^{n_d \times R}$, and $\circ$ denotes an outer product. Note that the graph regularization terms in (12) are applied to the factor matrices $\mathbf{A}^{(i)} \in \mathbb{R}^{n_i \times R}$, reducing the computational complexity of the estimation algorithm. Decomposing a data tensor into the sum of low-rank and sparse components is also used in [12], [13], and [19]. In [14], the computational complexity is further reduced by precalculating $\mathbf{P}_R^{(i)}$ mode-specific graph Laplacian eigenvectors of rank R from the matricization of the tensor along each mode and using these in solving tensor-robust PCA. The solution relies on projecting the tensor onto a tensor product of the graph basis $\{\mathbf{P}_R^{(i)}\}$, resulting in a formulation similar to the Tucker decomposition.

Coclustering assumes that the observed tensor is the sum of a "checkerbox" tensor (under suitable permutations along the modes) and additive noise. For example, Chi et al. [10] propose estimating a "checkerbox" tensor with the minimizer to a convex criterion. In the case of three-way tensors, the criterion is

$$\frac{1}{2}\|\mathcal{Y} - \mathcal{X}\|_{\mathrm{F}}^2 + \gamma \Bigg[\sum_{(i,j) \in \mathcal{E}^{(1)}} w_{ij}^{(1)} \|\mathcal{X}_{i::} - \mathcal{X}_{j::}\|_{\mathrm{F}} + \sum_{(i,j) \in \mathcal{E}^{(2)}} w_{ij}^{(2)} \|\mathcal{X}_{:i:} - \mathcal{X}_{:j:}\|_{\mathrm{F}} + \sum_{(i,j) \in \mathcal{E}^{(3)}} w_{ij}^{(3)} \|\mathcal{X}_{::i} - \mathcal{X}_{::j}\|_{\mathrm{F}} \Bigg],$$

where $\mathcal{E}^{(d)}$ is a set of edges for the mode-d graph, γ is a nonnegative tuning parameter, and $w_{ij}^{(d)}$ is a weight encoding the similarity between the ith and jth mode-d slices. Minimizing the criterion in (13) can be interpreted as simultaneously denoising all modes of the tensor via vector-valued-graph total variation.

Manifold learning on multiway data

Tensor factorization can fail to recover meaningful latent variables when nonlinear relationships exist among slices along each of the modes. Manifold learning overcomes such limitations by estimating nonlinear mappings from high-dimensional data to low-dimensional representations (embeddings). While GSP uses the eigenvectors of the graph Laplacian as a basis to linearly expand graph signals (1), manifold learning uses the eigenvectors ψ_ℓ themselves as a nonlinear d-dimensional map Ψ for the datapoints $\{\mathbf{x}_i\}_i$ as $\Psi: \mathbf{x}_i \to (\psi_1(i), \ldots, \psi_d(i))$.

A naïve strategy to apply manifold learning to the multiway data is to take the D different matricizations of a D-way tensor and construct a graph Laplacian using a generic metric on each of the D modes independently, thereby ignoring the higher-order coupled structure in the tensor. Recent work [6], [9], [32], however, incorporates a higher-order tensor structure in manifold learning by thoughtfully designing the similarity measures used to construct the mode-k graph weights $\mathbf{W}^{(k)}$. The comanifold learning framework can be viewed as blending GSP and manifold learning and has most recently extended to tensors and the missing-data setting [9], [32].

From an MWGSP perspective, the key contribution of this line of work is a new metric that is defined between tensor slices as the difference between a graph-based multiscale decomposition of each slice along its remaining modes; for example, the distance between two horizontal slices in a three-way tensor is

$$\begin{aligned} d(\mathcal{X}_{i\cdot\cdot}, \mathcal{X}_{j\cdot\cdot}) &= \|(\mathbf{M}^{(3)} \otimes \mathbf{M}^{(2)}) \operatorname{vec}(\mathcal{X}_{i::} - \mathcal{X}_{j::})\|_1 \\ &= \|\operatorname{vec}(\mathbf{M}^{(2)}(\mathcal{X}_{i::} - \mathcal{X}_{j::})(\mathbf{M}^{(3)})^\top)\|_1, \end{aligned} \tag{13}$$

where $\mathbf{M}^{(k)}$ is a multiscale transform in the kth mode. This metric was shown to be a tree-based Earth mover's distance in the 2D setting [39]. The resulting similarity depends on a multiway, multiscale difference between slices and has been successfully used, in practice, to construct weighted graphs in multiway data. The multiscale decompositions are constructed either from data-adaptive tree transforms [6] or through a series of multiway, graph-based coclustering solutions [32].

Future outlook

Although multiway signal processing frameworks continue to mature, several challenges remain ahead. While novel techniques are continually introduced into single-way GSP, one approach to developing multiway techniques is to identify, extend, and adapt techniques that are particularly useful for multiway signals. For instance, multiway analysis on directed graphs will greatly broaden the versatility of MWGSP. From a computational perspective, it is clear that the efficiency gains offered by the march of single-way GSP toward fast transforms [27] are compounded in the multiway setting.

From a theoretical perspective, open questions include the following:

- What additional advantages can be gained by treating classical domains as lying on graphs?
- How do we learn mode-specific and coupled graphs from data, in general, and in dynamical settings?
- Are such tensor data sets typically low or high rank?
- How do we process data whose generative model is nonlinear across the different modes?

From a practical perspective, the ongoing growth in computational power and parallel computing has enabled large-scale analyses. The MWGSP framework can leverage these recent advances in computational building blocks. Nonetheless, there are existing computational challenges, such as applications requiring online real-time processing. Thus, future directions include developing online and distributed versions of MWGSP, especially in the presence of large-scale data, where streaming solutions are necessary (the data does not fit in memory). In addition, there is need for new optimization techniques to efficiently solve problems that combine tensors with graph-based penalties. Deep learning is also emerging as a framework to learn, rather than design, wavelet-type filter banks in signal processing, and these approaches can be extended to the graph and multiway settings to learn joint multiscale decompositions. Finally, as the GSP community continues to address real-world data domains, such as climate, traffic, and biomedical research, interdisciplinary collaboration is essential to define relevant problems and demonstrate the significant utility of these approaches within a domain.

Deep learning is also emerging as a framework to learn, rather than design, wavelet-type filter banks in signal processing.

Acknowledgments

This research was supported by the National Institutes of Health, under grants R01RGM131642, P50CA121974, R01HG008383, R01GM135928, and R01EB026936, and the National Science Foundation, under grant DMS-1752692.

Authors

Jay S. Stanley III (jay.stanley@yale.edu) received his B.A. degree in biology from Hendrix College, Conway, Arkansas, in 2016 and his Ph.D. degree in computational biology and bioinformatics from Yale University, New Haven, Connecticut, in 2020. He is a postdoctoral associate in the Department of Mathematics, Yale University. His research interests include graph signal processing and applied harmonic analysis.

Eric C. Chi (ecchi@ncsu.edu) received his B.A. degree in physics from Rice University, Houston, Texas; his M.S. degree in electrical engineering from the University of

California, Berkeley; and his Ph.D. degree in statistics from Rice University. He is an assistant professor of statistics at North Carolina State University, Raleigh. His research interests include statistical learning and numerical optimization and their application to analyzing large and complicated modern data in biological science and engineering applications.

Gal Mishne (gmishne@ucsd.edu) received her B.Sc. degrees (summa cum laude) in electrical engineering and physics and her Ph.D. degree in electrical engineering from Technion–Israel Institute of Technology, Haifa, in 2009 and 2017, respectively. She is an assistant professor at the Halıcıoğlu Data Science Institute, University of California, San Diego. From 2017 to 2019, she was a Gibbs Assistant Professor in the Department of Mathematics, Yale University, New Haven, Connecticut. Her research interests include high-dimensional data analysis, image processing, applied harmonic analysis, and computational neuroscience. She is a Member of IEEE.

References

[1] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, 2018. doi: 10.1109/JPROC.2018.2820126.

[2] F. Grassi, A. Loukas, N. Perraudin, and B. Ricaud, "A time-vertex signal processing framework: Scalable processing and meaningful representations for time-series on graphs," *IEEE Trans. Signal Process.*, vol. 66, no. 3, pp. 817–829, 2018. doi: 10.1109/TSP.2017.2775589.

[3] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 856–869, 2017. doi: 10.1109/JSTSP.2017.2726976.

[4] N. Rao, H.-F. Yu, P. K. Ravikumar, and I. S. Dhillon, "Collaborative filtering with graph information: Consistency and scalable methods," in *Proc. 28th Int. Conf. Neural Information Processing Systems*, 2015, pp. 2107–2115. doi: 10.5555/2969442.2969475.

[5] E. C. Chi, G. I. Allen, and R. G. Baraniuk, "Convex biclustering," *Biometrics*, vol. 73, no. 1, pp. 10–19, 2017. doi: 10.1111/biom.12540.

[6] G. Mishne, R. Talmon, I. Cohen, R. R. Coifman, and Y. Kluger, "Data-driven tree transforms and metrics," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 3, pp. 451–466, 2018. doi: 10.1109/TSIPN.2017.2743561.

[7] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, 2014. doi: 10.1109/MSP.2014.2329213.

[8] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009. doi: 10.1137/07070111X.

[9] G. Mishne, R. Talmon, R. Meir, J. Schiller, M. Lavzin, U. Dubin, and R. R. Coifman, "Hierarchical coupled-geometry analysis for neuronal structure and activity pattern discovery," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 7, pp. 1238–1253, 2016. doi: 10.1109/JSTSP.2016.2602061.

[10] E. C. Chi, B. R. Gaines, W. W. Sun, H. Zhou, and J. Yang, Provable convex co-clustering of tensors. 2018. [Online]. Available: arXiv:1803.06518

[11] Q. Sun, M. Yan, D. Donoho, and S. Boyd, "Convolutional imputation of matrix networks," in *Proc. 35th Int. Conf. Machine Learning*, 2018, pp. 4818–4827.

[12] Y. Nie, L. Chen, H. Zhu, S. Du, T. Yue, and X. Cao, "Graph-regularized tensor robust principal component analysis for hyperspectral image denoising," *Appl. Opt.*, vol. 56, no. 22, pp. 6094–6102, 2017. doi: 10.1364/AO.56.006094.

[13] K. Zhang, M. Wang, S. Yang, and L. Jiao, "Spatial–spectral-graph-regularized low-rank tensor decomposition for multispectral and hyperspectral image fusion," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 4, pp. 1030–1040, 2018. doi: 10.1109/JSTARS.2017.2785411.

[14] N. Shahid, F. Grassi, and P. Vandergheynst, "Tensor robust PCA on graphs," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5406–5410. doi: 10.1109/ICASSP.2019.8682990.

[15] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 145–163, Mar. 2015. doi: 10.1109/MSP.2013.2297439.

[16] Y. Yankelevsky and M. Elad, "Dual graph regularized dictionary learning," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 611–624, 2016. doi: 10.1109/TSIPN.2016.2605763.

[17] C. Li, Q. Zhao, J. Li, A. Cichocki, and L. Guo, "Multi-tensor completion with common structures," in *Proc. 29th AAAI Conf. Artificial Intelligence*, 2015, pp. 2743–2749. doi: 10.5555/2886521.2886703.

[18] V. N. Ioannidis, A. S. Zamzam, G. B. Giannakis, and N. D. Sidiropoulos, "Coupled graph and tensor factorization for recommender systems and community detection," *IEEE Trans. Knowl. Data Eng.*, to be published. doi: 10.1109/TKDE.2019.2941716.

[19] Y. Su, X. Bai, W. Li, P. Jing, J. Zhang, and J. Liu, "Graph regularized low-rank tensor representation for feature selection," *J. Vis. Commun. Image Represent.*, vol. 56, pp. 234–244, Oct. 2018. doi: 10.1016/j.jvcir.2018.09.020.

[20] K. Xie, X. Li, X. Wang, G. Xie, J. Wen, and D. Zhang, "Graph based tensor recovery for accurate internet anomaly detection," in *Proc. IEEE INFOCOM 2018: IEEE Conf. Computer Communications*, pp. 1502–1510. doi: 10.1109/INFOCOM.2018.8486332.

[21] N. Rabin and D. Fishelov, "Two directional Laplacian pyramids with application to data imputation," *Adv. Comput. Math.*, vol. 45, pp. 2123–2146, Apr. 2019. doi: 10.1007/s10444-019-09697-7.

[22] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1994.

[23] A. Loukas and N. Perraudin, "Stationary time-vertex signal processing," *EURASIP J. Adv. Signal Process.*, vol. 2019, p. 36, Aug. 2019. doi: 10.1186/s13634-019-0631-7.

[24] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe, "220 band AVIRIS hyperspectral image data set: June 12, 1992 Indian Pine Test Site 3," Purdue Univ. Res. Repository, West Lafayette, IN, Sept. 2015. [Online]. Available: https://purr.purdue.edu/publications/1947/1

[25] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst, "Fast robust PCA on graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 4, pp. 740–756, 2016. doi: 10.1109/JSTSP.2016.2555239.

[26] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst, "Compressive spectral clustering," in *Proc. 33rd Int. Conf. Machine Learning*, 2016, pp. 1002–1011. doi: 10.5555/3045390.3045497.

[27] T. Frerix and J. Bruna, "Approximating orthogonal matrices with effective Givens factorization," in *Proc. 36th Int. Conf. Machine Learning*, 2019, pp. 1993–2001.

[28] F. Ji and W. P. Tay, "A Hilbert space theory of generalized graph signal processing," *IEEE Trans. Signal Process.*, vol. 67, no. 24, pp. 6188–6203, 2019. doi: 10.1109/TSP.2019.2952055.

[29] B. Girault, "Stationary graph signals using an isometric graph translation," in *Proc. 23rd European Signal Processing Conf. (EUSIPCO)*, 2015, pp. 1516–1520. doi: 10.1109/EUSIPCO.2015.7362637.

[30] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Trans. Signal Process.*, vol. 65, no. 22, pp. 5911–5926, 2017. doi: 10.1109/TSP.2017.2739099.

[31] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, Matrix completion on graphs. 2014. [Online]. Available: arXiv:1408.1717

[32] G. Mishne, E. C. Chi, and R. R. Coifman, "Co-manifold learning with missing data," in *Proc. 36th Int. Conf. Machine Learning (PMLR),* vol. 97, 2019, pp. 4605–4614.

[33] F. Shang, L. Jiao, and F. Wang, "Graph dual regularization non-negative matrix factorization for co-clustering," *Pattern Recognit.*, vol. 45, no. 6, pp. 2237–2250, 2012. doi: 10.1016/j.patcog.2011.12.015.

[34] N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Robust principal component analysis on graphs," in *Proc. IEEE Int. Conf. Computer Vision*, 2015, pp. 2812–2820. doi: 10.1109/ICCV.2015.322.

[35] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proc. IEEE Workshop Applications Computer Vision*, 1994, pp. 138–142. doi: 10.1109/ACV.1994.341300.

[36] K. Benzi, V. Kalofolias, X. Bresson, and P. Vandergheynst, "Song recommendation with non-negative matrix factorization and graph total variation," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016, pp. 2439–2443. doi: 10.1109/ICASSP.2016.7472115.

[37] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 16–43, 2019. doi: 10.1109/MSP.2018.2890143.

[38] N. Leonardi and D. Van De Ville, "Tight wavelet frames on multislice graphs," *IEEE Trans. Sig. Process.*, vol. 61, no. 13, pp. 3357–3367, July 2013. doi: 10.1109/TSP.2013.2259825.

[39] R. R. Coifman and W. E. Leeb, "Earth mover's distance and equivalent metrics for spaces with hierarchical partition trees," Yale Univ., New Haven, CT, Tech. Rep. YALEU/DCS/TR1482, 2013.

SP