

Discrete Signal Processing on Graphs

Aliaksei Sandryhaila, *Member, IEEE* and José M. F. Moura, *Fellow, IEEE*

Abstract—In social settings, individuals interact through webs of relationships. Each individual is a node in a complex network (or graph) of interdependencies and generates data, lots of data. We label the data by its source, or formally stated, we *index* the data by the nodes of the graph. The resulting signals (data indexed by the nodes) are far removed from time or image signals indexed by well ordered time samples or pixels. DSP, discrete signal processing, provides a comprehensive, elegant, and efficient methodology to describe, represent, transform, analyze, process, or synthesize these well ordered time or image signals. This paper extends to *signals on graphs* DSP and its basic tenets, including filters, convolution, z -transform, impulse response, spectral representation, Fourier transform, frequency response, and illustrates *DSP on graphs* by classifying blogs, linear predicting and compressing data from irregularly located weather stations, or predicting behavior of customers of a mobile service provider.

Keywords: Network science, signal processing, graphical models, Markov random fields, graph Fourier transform.

I. INTRODUCTION

There is an explosion of interest in processing and analyzing large datasets collected in very different settings, including social and economic networks, information networks, internet and the world wide web, immunization and epidemiology networks, molecular and gene regulatory networks, citation and coauthorship studies, friendship networks, as well as physical infrastructure networks like sensor networks, power grids, transportation networks, and other networked critical infrastructures. We briefly overview some of the existing work.

Many authors focus on the underlying *relational structure* of the data by: 1) inferring the structure from community relations and friendships, or from perceived alliances between agents as abstracted through game theoretic models [1], [2]; 2) quantifying the connectedness of the world; and 3) determining the relevance of particular agents, or studying the strength of their interactions. Other authors are interested in the *network function* by quantifying the impact of the network structure on the diffusion of disease, spread of news and information, voting trends, imitation and social influence, crowd behavior, failure propagation, global behaviors developing from seemingly random local interactions [2], [3], [4]. Much of these works either develop or assume network models that capture the interdependencies among the data and then analyze the structural properties of these networks. Models often considered may be deterministic like complete or regular

graphs, or random like the Erdős-Rényi and Poisson graphs, the configuration and expected degree models, small world or scale free networks [2], [4], to mention a few. These models are used to quantify network characteristics, such as connect-edness, existence and size of the giant component, distribution of component sizes, degree and clique distributions, and node or edge specific parameters including clustering coefficients, path length, diameter, betweenness and closeness centralities.

Another body of literature is concerned with inference and learning from such large datasets. Much work falls under the generic label of graphical models [5], [6], [7], [8], [9], [10]. In graphical models, data is viewed as a family of random variables indexed by the nodes of a graph, where the graph captures probabilistic dependencies among data elements. The random variables are described by a family of joint probability distributions. For example, directed (acyclic) graphs [11], [12] represent Bayesian networks where each random variable is independent of others given the variables defined on its parent nodes. Undirected graphical models, also referred to as Markov random fields [13], [14], describe data where the variables defined on two sets of nodes separated by a boundary set of nodes are statistically independent given the variables on the boundary set. A key tool in graphical models is the Hammersley-Clifford theorem [13], [15], [16], and the Markov-Gibbs equivalence that, under appropriate positivity conditions, factors the joint distribution of the graphical model as a product of potentials defined on the cliques of the graph. Graphical models exploit this factorization and the structure of the indexing graph to develop efficient algorithms for inference by controlling their computational cost. Inference in graphical models is generally defined as finding from the joint distributions lower order marginal distributions, likelihoods, modes, and other moments of individual variables or their subsets. Common inference algorithms include belief propagation and its generalizations, as well as other message passing algorithms. A recent block-graph algorithm for fast approximate inference, in which the nodes are non-overlapping clusters of nodes from the original graph, is in [17]. Graphical models are employed in many areas; for sample applications, see [18] and references therein.

Extensive work is dedicated to discovering efficient data representations for large high-dimensional data [19], [20], [21], [22]. Many of these works use spectral graph theory and the graph Laplacian [23] to derive low-dimensional representations by projecting the data on a low-dimensional subspace generated by a small subset of the Laplacian eigenbasis. The graph Laplacian approximates the Laplace-Beltrami operator on a compact manifold [24], [21], in the sense that if the dataset is large and samples uniformly randomly a low-dimensional manifold then the (empirical) graph Laplacian acting on a smooth function on this manifold is a good discrete

Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

This work was supported in part by AFOSR grant FA95501210087. A. Sandryhaila and J. M. F. Moura are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3890. Ph: (412)268-6341; fax: (412)268-3890. Email: asandryh@andrew.cmu.edu, moura@ece.cmu.edu.

approximation that converges pointwise and uniformly to the elliptic Laplace-Beltrami operator applied to this function as the number of points goes to infinity [25], [26], [27]. One can go beyond the choice of the graph Laplacian by choosing discrete approximations to other continuous operators and obtaining possibly more desirable spectral bases for the characterization of the geometry of the manifold underlying the data. For example, if the data represents a non-uniform sampling of a continuous manifold, a conjugate to an elliptic Schrödinger-type operator can be used [28], [29], [30].

More in line with our paper, several works have proposed multiple transforms for data indexed by graphs. Examples include regression algorithms [31], wavelet decompositions [32], [33], [34], [30], [35], filter banks on graphs [36], [37], denoising [38], and compression [39]. Some of these transforms focus on distributed processing of data from sensor fields while addressing sampling irregularities due to random sensor placement. Others consider localized processing of signals on graphs in multiresolution fashion by representing data using wavelet-like bases with varying “smoothness” or defining transforms based on node neighborhoods. In the latter case, the graph Laplacian and its eigenbasis are sometimes used to define a spectrum and a Fourier transform of a signal on a graph. This definition of a Fourier transform was also proposed for use in uncertainty analysis on graphs [40], [41]. This graph Fourier transform is derived from the graph Laplacian and restricted to undirected graphs with real, non-negative edge weights, not extending to data indexed by directed graphs or graphs with negative or complex weights.

The algebraic signal processing (ASP) theory [42], [43], [44], [45] is a formal, algebraic approach to analyze data indexed by special types of line graphs and lattices. The theory uses an algebraic representation of signals and filters as polynomials to derive fundamental signal processing concepts. This framework has been used for discovery of fast computational algorithms for discrete signal transforms [42], [46], [47]. It was extended to multidimensional signals and nearest neighbor graphs [48], [49] and applied in signal compression [50], [51]. The framework proposed in this paper generalizes and extends the ASP to signals on arbitrary graphs.

Contribution

Our goal is to develop a linear discrete signal processing (DSP) framework and corresponding tools for datasets arising from social, biological, and physical networks. DSP has been very successful in processing time signals (such as speech, communications, radar, or econometric time series), space-dependent signals (images and other multidimensional signals like seismic and hyperspectral data), and time-space signals (video). We refer to data *indexed* by nodes of a graph as a *graph signal* or simply signal and to our approach as *DSP on graphs* (DSP_G)¹. We introduce the basics of linear²

DSP_G , including the notion of a shift on a graph, filter structure, filtering and convolution, signal and filter spaces and their algebraic structure, the graph Fourier transform, frequency, spectrum, spectral decomposition, and impulse and frequency responses. With respect to other works, ours is a deterministic framework to signal processing on graphs rather than a statistical approach like graphical models. Our work is an extension and generalization of the traditional DSP, and generalizes the ASP theory [42], [43], [44], [45] and its extensions and applications [49], [50], [51]. We emphasize the contrast between the DSP_G and the approach to the graph Fourier transform that takes the graph Laplacian as a point of departure [32], [38], [36], [35], [39], [41]. In the latter case, the Fourier transform on graphs is given by the eigenbasis of the graph Laplacian. However, this definition is not applicable to directed graphs, which often arise in real-world problems, as demonstrated by examples in Section VI, and graphs with negative weights. In general, the graph Laplacian is a second-order operator for signals on a graph, whereas an adjacency matrix is a first-order operator. Deriving a graph Fourier transform from the graph Laplacian is analogous in traditional DSP to restricting signals to be even (like correlation sequences) and Fourier transforms to represent power spectral densities of signals. Instead, we demonstrate that the graph Fourier transform is properly defined through the Jordan normal form and generalized eigenbasis of the adjacency matrix³. Finally, we illustrate the DSP_G with applications like classification, compression, and linear prediction for datasets that include blogs, customers of a mobile operator, or collected by a network of irregularly placed weather stations.

II. SIGNALS ON GRAPHS

Consider a dataset with N elements, for which some *relational* information about its data elements is known. Examples include preferences of individuals in a social network and their friendship connections, the number of papers published by authors and their coauthorship relations, or topics of online documents in the World Wide Web and their hyperlink references. This information can be represented by a graph $G = (\mathcal{V}, \mathbf{A})$, where $\mathcal{V} = \{v_0, \dots, v_{N-1}\}$ is the set of nodes and \mathbf{A} is the weighted⁴ adjacency matrix of the graph. Each dataset element corresponds to node v_n , and each weight $\mathbf{A}_{n,m}$ of a *directed* edge from v_m to v_n reflects the degree of relation of the n th element to the m th one. Since data elements can be related to each other differently, in general, G is a *directed, weighted* graph. Its edge weights $\mathbf{A}_{n,m}$ are not restricted to being nonnegative reals; they can take arbitrary real or complex values (for example, if data elements are negatively correlated). The set of indices of nodes connected to v_n is called the *neighborhood* of v_n and denoted by $\mathcal{N}_n = \{m \mid \mathbf{A}_{n,m} \neq 0\}$.

¹The term “signal processing for graphs” has been used in [52], [53] in reference to graph structure analysis and subgraph detection. It should not be confused with our proposed DSP framework, which aims at the analysis and processing of *data* indexed by the nodes of a graph.

²We are concerned with linear operations; in the sequel, we refer only to DSP_G but have in mind that we are restricted to linear DSP_G .

³Parts of this material also appeared in [54], [55]. In this paper, we present a complete theory with all derivations and proofs.

⁴Some literature defines the adjacency matrix \mathbf{A} of a graph $G = (\mathcal{V}, \mathbf{A})$ so that $\mathbf{A}_{n,m}$ only takes values of 0 or 1, depending on whether there is an edge from v_m to v_n , and specifies edge weights as a function on pairs of nodes. In this paper, we incorporate edge weights into \mathbf{A} .

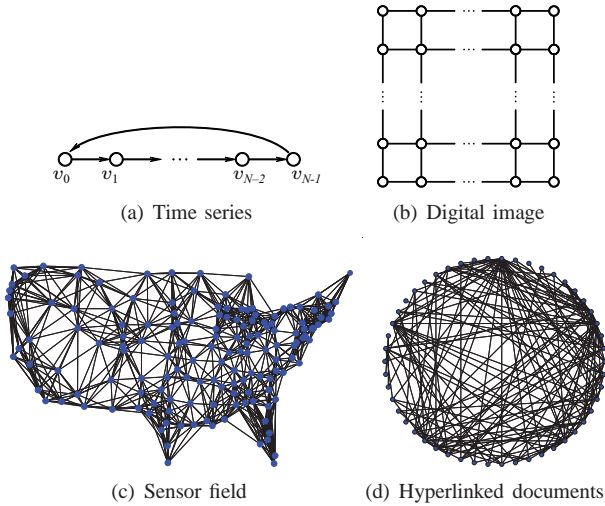


Fig. 1. Graph representations for different datasets (graph signals.)

Assuming, without a loss of generality, that dataset elements are complex scalars, we define a *graph signal* as a map from the set \mathcal{V} of nodes into the set of complex numbers \mathbb{C} :

$$\begin{aligned} \mathbf{s} &: \mathcal{V} \rightarrow \mathbb{C}, \\ v_n &\mapsto s_n. \end{aligned} \quad (1)$$

Notice that each signal is isomorphic to a complex-valued vector with N elements. Hence, for simplicity of discussion, we write graph signals as vectors $\mathbf{s} = (s_0 \ s_1 \ \dots \ s_{N-1})^T$, but remember that each element s_n is *indexed* by node v_n of a given representation graph $G = (\mathcal{V}, \mathbf{A})$, as defined by (1). The space \mathcal{S} of graphs signals (1) then is identical to \mathbb{C}^N .

We illustrate representation graphs with examples shown in Fig. 1. The directed cyclic graph in Fig. 1(a) represents a finite, periodic discrete time series [44]. All edges are directed and have the same weight 1, reflecting the causality of a time series; and the edge from v_{N-1} to v_0 reflects its periodicity. The two-dimensional rectangular lattice in Fig. 1(b) represents a general digital image. Each node corresponds to a pixel, and each pixel value (intensity) is related to the values of the four adjacent pixels. This relation is symmetric, hence all edges are undirected and have the same weight, with possible exceptions of boundary nodes that may have directed edges and/or different edge weights, depending on boundary conditions [45]. Other lattice models can be used for images as well [48]. The graph in Fig. 1(c) represents temperature measurements from 150 weather stations (sensors) across the United States. We represent the relations of temperature measurements by geodesic distances between sensors, so each node is connected to its closest neighbors. The graph in Fig. 1(d) represents a set of 50 political blogs in the World Wide Web connected by hyperlink references. By their nature, the edges are directed and have the same weights. We discuss the two latter examples in Section VI, where we also consider a network of customers of a mobile service provider. Clearly, representation graphs depend on prior knowledge and assumptions about datasets. For example, the graph in Fig. 1(d) is obtained by following the hyperlinks networking the blogs, while the graph in Fig. 1(c) is constructed from known locations of sensors under assumption that temperature measurements at nearby sensors

have highly correlated temperatures.

III. FILTERS ON GRAPHS

In classical DSP, signals are processed by *filters*—systems that take a signal as input and produce another signal as output. We now develop the equivalent concept of *graph filters* for graph signals in DSP_G. We consider only linear, shift-invariant filters, which are a generalization of linear time-invariant filters used in DSP for time series. This section uses Jordan normal form and characteristic and minimal polynomials of matrices; these concepts are reviewed in Appendix A. The use of Jordan decomposition is required since for many real-world datasets the adjacency matrix \mathbf{A} is not diagonalizable. One example is the blog dataset, considered in Section VI.

Graph Shift

In classical DSP, the basic building block of filters is a special filter $x = z^{-1}$ called the *time shift* or *delay* [56]. This is the simplest non-trivial filter that delays the input signal \mathbf{s} by one sample, so that the n th sample of the output is $\tilde{s}_n = s_{n-1 \bmod N}$. Using the graph representation of finite, periodic time series in Fig. 1(a), for which the adjacency matrix is the $N \times N$ circulant matrix $\mathbf{A} = \mathbf{C}_N$, with weights [43], [44]

$$\mathbf{A}_{n,m} = \begin{cases} 1, & \text{if } n - m = 1 \bmod N \\ 0, & \text{otherwise} \end{cases}, \quad (2)$$

we can write the time shift operation as

$$\tilde{\mathbf{s}} = \mathbf{C}_N \mathbf{s} = \mathbf{A} \mathbf{s}. \quad (3)$$

In DSP_G, we extend the notion of the shift (3) to general graph signals \mathbf{s} where the relational dependencies among the data are represented by an arbitrary graph $G = (\mathcal{V}, \mathbf{A})$. We call the operation (3) the *graph shift*. It is realized by replacing the sample s_n at node v_n with the weighted linear combination of the signal samples at its neighbors:

$$\tilde{s}_n = \sum_{m=0}^{N-1} \mathbf{A}_{n,m} s_m = \sum_{m \in \mathcal{N}_n} \mathbf{A}_{n,m} s_m. \quad (4)$$

Note that, in classical DSP, shifting a finite signal requires one to consider boundary conditions. In DSP_G, this problem is implicitly resolved, since the graph $G = (\mathcal{V}, \mathbf{A})$ explicitly captured the boundary conditions.

Graph Filters

Similarly to traditional DSP, we can represent filtering on a graph using matrix-vector multiplication. Any system $\mathbf{H} \in \mathbb{C}^{N \times N}$, or *graph filter*, that for input $\mathbf{s} \in \mathcal{S}$ produces output $\mathbf{H}\mathbf{s}$ represents a *linear* system, since the filter's output for a linear combination of input signals equals the linear combination of outputs to each signal:

$$\mathbf{H}(\alpha \mathbf{s}_1 + \beta \mathbf{s}_2) = \alpha \mathbf{H}\mathbf{s}_1 + \beta \mathbf{H}\mathbf{s}_2.$$

Furthermore, we focus on *shift-invariant* graph filters, for which applying the graph shift to the output is equivalent to applying the graph shift to the input prior to filtering:

$$\mathbf{A}(\mathbf{H}\mathbf{s}) = \mathbf{H}(\mathbf{A}\mathbf{s}). \quad (5)$$

The next theorem establishes that all linear, shift-invariant graph filters are given by *polynomials* in the shift \mathbf{A} .

Theorem 1: Let \mathbf{A} be the graph adjacency matrix and assume that its characteristic and minimal polynomials are equal: $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$. Then, a graph filter \mathbf{H} is linear and shift invariant if and only if (iff) \mathbf{H} is a *polynomial* in the graph shift \mathbf{A} , i.e., iff there exists a polynomial

$$h(x) = h_0 + h_1x + \dots + h_Lx^L \quad (6)$$

with possibly complex coefficients $h_\ell \in \mathbb{C}$, such that:

$$\mathbf{H} = h(\mathbf{A}) = h_0\mathbf{I} + h_1\mathbf{A} + \dots + h_L\mathbf{A}^L. \quad (7)$$

Proof: Since the shift-invariance condition (5) holds for all graph signals $\mathbf{s} \in \mathcal{S} = \mathbb{C}^N$, the matrices \mathbf{A} and \mathbf{H} commute: $\mathbf{A}\mathbf{H} = \mathbf{H}\mathbf{A}$. As $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$, all eigenvalues of \mathbf{A} have exactly one eigenvector associated with them, [57], [58]. Then, the graph matrix \mathbf{H} commutes with the shift \mathbf{A} iff it is a polynomial in \mathbf{A} (see Proposition 12.4.1 in [58]). ■

Analogous to the classical DSP, we call the coefficients h_ℓ of the polynomial $h(x)$ in (6) the graph filter *taps*.

Properties of Graph Filters

Theorem 1 requires the equality of the characteristic and minimal polynomials $p_{\mathbf{A}}(x)$ and $m_{\mathbf{A}}(x)$. This condition does not always hold, but can be successfully addressed through the concept of *equivalent* graph filters, as defined next.

Definition 1: Given any shift matrices \mathbf{A} and $\tilde{\mathbf{A}}$, filters $h(\mathbf{A})$ and $g(\tilde{\mathbf{A}})$ are called *equivalent* if for all input signals $\mathbf{s} \in \mathcal{S}$ they produce equal outputs: $h(\mathbf{A})\mathbf{s} = g(\tilde{\mathbf{A}})\mathbf{s}$.

Note that, when no restrictions are placed on the signals, so that $\mathcal{S} = \mathbb{C}^N$, Definition 1 is equivalent to requiring $h(\mathbf{A}) = g(\tilde{\mathbf{A}})$ as matrices. However, if additional restrictions exist, filters may not necessarily be equal as matrices and still produce the same output for the considered set of signals.

It follows that, given an arbitrary $G = (\mathcal{V}, \mathbf{A})$ with $p_{\mathbf{A}}(x) \neq m_{\mathbf{A}}(x)$, we can consider another graph $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{A}})$ with the same set of nodes \mathcal{V} but potentially different edges and edge weights, for which $p_{\tilde{\mathbf{A}}}(x) = m_{\tilde{\mathbf{A}}}(x)$ holds true. Then graph filters on G can be expressed as equivalent filters on \tilde{G} , as described by the following theorem (proven in Appendix B).

Theorem 2: For any matrix \mathbf{A} there exists a matrix $\tilde{\mathbf{A}}$ and polynomial $r(x)$, such that $\mathbf{A} = r(\tilde{\mathbf{A}})$ and $p_{\tilde{\mathbf{A}}}(x) = m_{\tilde{\mathbf{A}}}(x)$.

As a consequence of Theorem 2, any filter on the graph $G = (\mathcal{V}, \mathbf{A})$ is equivalent to a filter on the graph $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{A}})$, since $h(\mathbf{A}) = h(r(\tilde{\mathbf{A}})) = (h \circ r)(\tilde{\mathbf{A}})$, where $h \circ r$ is the composition of polynomials h and r and thus is a polynomial. Thus, the condition $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$ in Theorem 1 can be assumed to hold for any graph $G = (\mathcal{V}, \mathbf{A})$. Otherwise, by Theorem 2, we can replace the graph by another $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{A}})$ for which the condition holds and assign $\tilde{\mathbf{A}}$ to \mathbf{A} .

The next result demonstrates that we can limit the number of taps in any graph filter.

Theorem 3: Any graph filter (7) has a unique equivalent filter on the same graph with at most $\deg m_{\mathbf{A}}(x) = N_{\mathbf{A}}$ taps.

Proof: Consider the polynomials $h(x)$ in (6). By polynomial division, there exist unique polynomials $q(x)$ and $r(x)$:

$$h(x) = q(x)m_{\mathbf{A}}(x) + r(x), \quad (8)$$

where $\deg r(x) < N_{\mathbf{A}}$. Hence, we can express (7) as

$$h(\mathbf{A}) = q(\mathbf{A})m_{\mathbf{A}}(\mathbf{A}) + r(\mathbf{A}) = q(\mathbf{A})\mathbf{0}_N + r(\mathbf{A}) = r(\mathbf{A}).$$

Thus, $h(\mathbf{A}) = r(\mathbf{A})$ and $\deg r(x) < \deg m_{\mathbf{A}}(x)$. ■

As follows from Theorem 3, all linear, shift-invariant filters (7) on a graph $G = (\mathcal{V}, \mathbf{A})$ form a vector space

$$\mathcal{F} = \left\{ \mathbf{H} : \mathbf{H} = \sum_{\ell=0}^{N_{\mathbf{A}}-1} h_\ell \mathbf{A}^\ell \mid h_\ell \in \mathbb{C} \right\}. \quad (9)$$

Moreover, addition and multiplication of filters in \mathcal{F} produce new filters that are equivalent to filters in \mathcal{F} . Thus, \mathcal{F} is closed under these operations, and has the structure of an algebra [43]. We discuss it in detail in Section IV.

Another consequence of Theorem 3 is that the inverse of a filter on a graph, if it exists, is also a filter on the same graph, i.e., it is a polynomial in (9).

Theorem 4: A graph filter $\mathbf{H} = h(\mathbf{A}) \in \mathcal{F}$ is invertible iff polynomial $h(x)$ satisfies $h(\lambda_m) \neq 0$ for all distinct eigenvalues $\lambda_0, \dots, \lambda_{M-1}$, of \mathbf{A} . Then, there is a unique polynomial $g(x)$ of degree $\deg g(x) < N_{\mathbf{A}}$ that satisfies

$$h(\mathbf{A})^{-1} = g(\mathbf{A}) \in \mathcal{F}. \quad (10)$$

Appendix C contains the proof and the procedure for the construction of $g(x)$. Theorem 4 implies that instead of inverting the $N \times N$ matrix $h(\mathbf{A})$ directly we only need to construct a polynomial $g(x)$ specified by at most $N_{\mathbf{A}}$ taps.

Finally, it follows from Theorem 3 and (9) that any graph filter $h(\mathbf{A}) \in \mathcal{F}$ is completely specified by its taps $h_0, \dots, h_{N_{\mathbf{A}}-1}$. As we prove next, in DSP_G , as in traditional DSP, filter taps uniquely determine the *impulse response* of the filter, i.e., its output $\mathbf{u} = (g_0, \dots, g_{N-1})^T$ for unit impulse input $\boldsymbol{\delta} = (1, 0, \dots, 0)^T$, and vice versa.

Theorem 5: The filter taps $h_0, \dots, h_{N_{\mathbf{A}}-1}$ of the filter $h(\mathbf{A})$ uniquely determine its impulse response \mathbf{u} . Conversely, the impulse response \mathbf{u} uniquely determines the filter taps, provided $\text{rank } \hat{\mathbf{A}} = N_{\mathbf{A}}$, where $\hat{\mathbf{A}} = (\mathbf{A}^0 \boldsymbol{\delta}, \dots, \mathbf{A}^{N_{\mathbf{A}}-1} \boldsymbol{\delta})$.

Proof: The first part follows from the definition of filtering: $\mathbf{u} = h(\mathbf{A})\boldsymbol{\delta} = \hat{\mathbf{A}}\mathbf{h}$ yields the first column of $h(\mathbf{A})$, which is uniquely determined by the taps $\mathbf{h} = (h_0, \dots, h_{N_{\mathbf{A}}-1})^T$. Since we assume $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$, then $N = N_{\mathbf{A}}$, and the second part holds if $\hat{\mathbf{A}}$ is invertible, i.e., $\text{rank } \hat{\mathbf{A}} = N_{\mathbf{A}}$. ■

Notice that a relabeling of the nodes v_0, \dots, v_{N-1} does not change the impulse response. If \mathbf{P} is the corresponding permutation matrix, then the unit impulse is $\mathbf{P}\boldsymbol{\delta}$, the adjacency matrix is $\mathbf{P}\mathbf{A}\mathbf{P}^T$, and the filter becomes $h(\mathbf{P}\mathbf{A}\mathbf{P}^T) = \mathbf{P}h(\mathbf{A})\mathbf{P}^T$. Hence, the impulse response is simply reordered according to same permutation: $\mathbf{P}h(\mathbf{A})\mathbf{P}^T\mathbf{P}\boldsymbol{\delta} = \mathbf{P}\mathbf{u}$.

IV. ALGEBRAIC MODEL

So far, we presented signals and filters on graphs as vectors and matrices. An alternative representation exists for filters and signals as polynomials. We call this representation the graph z -transform, since, as we show, it generalizes the traditional z -transform for discrete time signals that maps signals and filters to polynomials or series in z^{-1} . The graph z -transform is defined separately for graph filters and signals.

Consider a graph $G = (\mathcal{V}, \mathbf{A})$, for which the characteristic and minimal polynomials of the adjacency matrix coincide: $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$. The mapping $\mathbf{A} \mapsto x$ of the adjacency matrix \mathbf{A} to the indeterminate x maps the graph filters $\mathbf{H} = h(\mathbf{A})$ in \mathcal{F} to polynomials $h(x)$. By Theorem 3, the filter space \mathcal{F} in (9) becomes a *polynomial algebra* [43]

$$\mathcal{A} = \mathbb{C}[x]/m_{\mathbf{A}}(x). \quad (11)$$

This is a space of polynomials of degree less than $\deg m_{\mathbf{A}}(x)$ with complex coefficients that is closed under addition and multiplication of polynomials modulo $m_{\mathbf{A}}(x)$. The mapping $\mathcal{F} \rightarrow \mathcal{A}$, $h(\mathbf{A}) \mapsto h(x)$, is an isomorphism of \mathbb{C} -algebras [43], which we denote as $\mathcal{F} \cong \mathcal{A}$. We call it the *graph z -transform of filters* on graph $G = (\mathcal{V}, \mathbf{A})$.

The signal space \mathcal{S} is a vector space that is also closed under filtering, i.e., under multiplication by graph filters from \mathcal{F} : for any signal $\mathbf{s} \in \mathcal{S}$ and filter $h(\mathbf{A})$, the output is a signal in the same space: $h(\mathbf{A})\mathbf{s} \in \mathcal{S}$. Thus, \mathcal{S} is an \mathcal{F} -module [43]. As we show next, the *graph z -transform of signals* is defined as an isomorphism (13) from \mathcal{S} to an \mathcal{A} -module.

Theorem 6: Under the above conditions, the signal space \mathcal{S} is isomorphic to an \mathcal{A} -module

$$\mathcal{M} = \mathbb{C}[x]/p_{\mathbf{A}}(x) = \left\{ s(x) = \sum_{n=0}^{N-1} s_n b_n(x) \right\} \quad (12)$$

under the mapping

$$\mathbf{s} = (s_0, \dots, s_{N-1})^T \mapsto s(x) = \sum_{n=0}^{N-1} s_n b_n(x). \quad (13)$$

The polynomials $b_0(x), \dots, b_{N-1}(x)$ are linearly independent polynomials of degree at most $N-1$. If we write

$$\mathbf{b}(x) = (b_0(x), \dots, b_{N-1}(x))^T, \quad (14)$$

then the polynomials satisfy

$$\mathbf{b}^{(r)}(\lambda_m) = \left(b_0^{(r)}(\lambda_m) \quad \dots \quad b_{N-1}^{(r)}(\lambda_m) \right)^T = r! \tilde{\mathbf{v}}_{m,0,r} \quad (15)$$

for $0 \leq r < R_{m,0}$ and $0 \leq m < M$, where λ_m and $\tilde{\mathbf{v}}_{m,0,r}$ are generalized eigenvectors of \mathbf{A}^T , and $b_n^{(r)}(\lambda_m)$ denotes the r th derivative of $b_n(x)$ evaluated at $x = \lambda_m$. Filtering in \mathcal{M} is performed as multiplication modulo $p_{\mathbf{A}}(x)$: if $\tilde{\mathbf{s}} = h(\mathbf{A})\mathbf{s}$, then

$$\tilde{\mathbf{s}} \mapsto \tilde{s}(x) = \sum_{n=0}^{N-1} \tilde{s}_n b_n(x) = h(x)s(x) \quad \text{mod } p_{\mathbf{A}}(x). \quad (16)$$

Proof: Due to the linearity and shift-invariance of graph filters, we only need to prove (16) for $h(\mathbf{A}) = \mathbf{A}$. Let us write $s(x) = \mathbf{b}(x)^T \mathbf{s}$ and $\tilde{s}(x) = \mathbf{b}(x)^T \tilde{\mathbf{s}}$, where $\mathbf{b}(x)$ is given by (14). Since (16) must hold for all $\mathbf{s} \in \mathcal{S}$, for $h(\mathbf{A}) = \mathbf{A}$ it is equivalent to

$$\begin{aligned} \mathbf{b}(x)^T \tilde{\mathbf{s}} &= \mathbf{b}(x)^T (\mathbf{A}\mathbf{s}) = \mathbf{b}(x)^T (x\mathbf{s}) \quad \text{mod } p_{\mathbf{A}}(x) \\ \Leftrightarrow (\mathbf{A}^T - x\mathbf{I}) \mathbf{b}(x) &= \mathbf{c} p_{\mathbf{A}}(x), \end{aligned} \quad (17)$$

where $\mathbf{c} \in \mathbb{C}^N$ is a vector of constants, since $\deg p_{\mathbf{A}}(x) = N$ and $\deg(xb_n(x)) \leq N$ for $0 \leq n < N$.

It follows from the factorization (43) of $p_{\mathbf{A}}(x)$ that, for each eigenvalue λ_m and $0 \leq k < A_m$, the characteristic polynomial satisfies $p_{\mathbf{A}}^{(k)}(\lambda_m) = 0$. By taking derivatives of both sides of (17) and evaluating at $x = \lambda_m$, $0 \leq m < M$, we construct $A_0 + \dots + A_{M-1} = N$ linear equations

$$\begin{aligned} (\mathbf{A}^T - \lambda_m \mathbf{I}) \mathbf{b}(\lambda_m) &= 0 \\ (\mathbf{A}^T - \lambda_m \mathbf{I}) \mathbf{b}^{(r)}(\lambda_m) &= r \mathbf{b}(\lambda_m), \quad 1 \leq r < A_m \end{aligned}$$

Comparing these equations with (35), we obtain (15). Since N polynomials $b_n(x) = b_{n,0} + \dots + b_{n,N-1}x^{N-1}$ are characterized by N^2 coefficients $b_{n,k}$, $0 \leq n, k < N$, (15) is a system of N linear equations with N^2 unknowns that can always be solved using inverse polynomial interpolation [58]. ■

Theorem 6 extends to the general case $p_{\mathbf{A}}(x) \neq m_{\mathbf{A}}(x)$. By Theorem 2, there exists a graph $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{A}})$ with $p_{\tilde{\mathbf{A}}}(x) = m_{\tilde{\mathbf{A}}}(x)$, such that $\mathbf{A} = r(\tilde{\mathbf{A}})$. By mapping $\tilde{\mathbf{A}}$ to x , the filter space (9) has the structure of the polynomial algebra $\mathcal{A} = \mathbb{C}[x]/m_{\tilde{\mathbf{A}}}(r(x)) = \mathbb{C}[x]/(m_{\tilde{\mathbf{A}}} \circ r)(x)$ and the signal space has the structure of the \mathcal{A} -module $\mathcal{M} = \mathbb{C}[x]/p_{\tilde{\mathbf{A}}}(x)$. Multiplication of filters and signals is performed modulo $p_{\tilde{\mathbf{A}}}(x)$. The basis of \mathcal{M} satisfies (15), where λ_m and $\mathbf{v}_{m,d,r}$ are eigenvalues and generalized eigenvectors of $\tilde{\mathbf{A}}$.

V. FOURIER TRANSFORM ON GRAPHS

After establishing the structure of filter and signal spaces in DSP_G , we define other fundamental DSP concepts, including spectral decomposition, signal spectrum, Fourier transform, and frequency response. They are related to the Jordan normal form of the adjacency matrix \mathbf{A} , reviewed in Appendix A.

Spectral Decomposition

In DSP, spectral decomposition refers to the identification of subspaces $\mathcal{S}_0, \dots, \mathcal{S}_{K-1}$ of the signal space \mathcal{S} that are *invariant* to filtering, so that, for any signal $\mathbf{s}_k \in \mathcal{S}_k$ and filter $h(\mathbf{A}) \in \mathcal{F}$, the output $\tilde{\mathbf{s}}_k = h(\mathbf{A})\mathbf{s}_k$ lies in the same subspace \mathcal{S}_k . A signal $\mathbf{s} \in \mathcal{S}$ can then be represented as

$$\mathbf{s} = \mathbf{s}_0 + \mathbf{s}_1 + \dots + \mathbf{s}_{K-1}, \quad (18)$$

with each component $\mathbf{s}_k \in \mathcal{S}_k$. Decomposition (18) is uniquely determined for every signal $\mathbf{s} \in \mathcal{S}$ if and only if: 1) invariant subspaces \mathcal{S}_k have zero intersection, i.e., $\mathcal{S}_k \cap \mathcal{S}_m = \{0\}$ for $k \neq m$; 2) $\dim \mathcal{S}_0 + \dots + \dim \mathcal{S}_{K-1} = \dim \mathcal{S} = N$; and 3) each \mathcal{S}_k is *irreducible*, i.e., it cannot be decomposed into smaller invariant subspaces. In this case, \mathcal{S} is written as a direct sum of vector subspaces

$$\mathcal{S} = \mathcal{S}_0 \oplus \mathcal{S}_1 \oplus \dots \oplus \mathcal{S}_{K-1}. \quad (19)$$

As mentioned, since the graph may have arbitrary structure, the adjacency matrix \mathbf{A} may not be diagonalizable; in fact, \mathbf{A} for the blog dataset (see Section VI) is not diagonalizable. Hence, we consider the Jordan decomposition (39) $\mathbf{A} = \mathbf{V} \mathbf{J} \mathbf{V}^{-1}$, which is reviewed in Appendix A. Here, \mathbf{J} is the Jordan normal form (40), and \mathbf{V} is the matrix of generalized eigenvectors (38). Let $\mathcal{S}_{m,d} = \text{span}\{\mathbf{v}_{m,d,0}, \dots, \mathbf{v}_{m,d,R_{m,d}-1}\}$ be a vector subspace of \mathcal{S}

spanned by the d th Jordan chain of λ_m . Any signal $\mathbf{s}_{m,d} \in \mathcal{S}_{m,d}$ has a unique expansion

$$\begin{aligned} \mathbf{s}_{m,d} &= \hat{\mathbf{s}}_{m,d,0} \mathbf{V}_{m,d,0} + \dots + \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \mathbf{V}_{m,d,R_{m,d}-1} \\ &= \mathbf{V}_{m,d} \left(\hat{\mathbf{s}}_{m,d,0} \quad \dots \quad \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \right)^T, \end{aligned}$$

where $\mathbf{V}_{m,d}$ is the block of generalized eigenvectors (37). As follows from the Jordan decomposition (39), shifting the signal $\mathbf{s}_{m,d}$ produces the output $\hat{\mathbf{s}}_{m,d} \in \mathcal{S}_{m,d}$ from the same subspace, since

$$\begin{aligned} \hat{\mathbf{s}}_{m,d} &= \mathbf{A} \mathbf{s}_{m,d} = \mathbf{A} \mathbf{V}_{m,d} \left(\hat{\mathbf{s}}_{m,d,0} \quad \dots \quad \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \right)^T \\ &= \mathbf{V}_{m,d} \mathbf{J}_{R_{m,d}}(\lambda_m) \left(\hat{\mathbf{s}}_{m,d,0} \quad \dots \quad \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \right)^T \\ &= \mathbf{V}_{m,d} \begin{pmatrix} \lambda_m \hat{\mathbf{s}}_{m,d,0} + \hat{\mathbf{s}}_{m,d,1} \\ \vdots \\ \lambda_m \hat{\mathbf{s}}_{m,d,R_{m,d}-2} + \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \\ \lambda_m \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \end{pmatrix}. \end{aligned} \quad (20)$$

Hence, each subspace $\mathcal{S}_{m,d} \leq \mathcal{S}$ is invariant to shifting.

Using (39) and Theorem 1, we write the graph filter (7) as

$$\begin{aligned} h(\mathbf{A}) &= \sum_{\ell=0}^L h_\ell (\mathbf{V} \mathbf{J} \mathbf{V}^{-1})^\ell = \sum_{\ell=0}^L h_\ell \mathbf{V} \mathbf{J}^\ell \mathbf{V}^{-1} \\ &= \mathbf{V} \left(\sum_{\ell=0}^L h_\ell \mathbf{J}^\ell \right) \mathbf{V}^{-1} = \mathbf{V} h(\mathbf{J}) \mathbf{V}^{-1}. \end{aligned} \quad (21)$$

Similarly to (20), we observe that filtering a signal $\mathbf{s}_{m,d} \in \mathcal{S}_{m,d}$ produces an output $\hat{\mathbf{s}}_{m,d} \in \mathcal{S}_{m,d}$ from the same subspace:

$$\begin{aligned} \hat{\mathbf{s}}_{m,d} &= h(\mathbf{A}) \mathbf{s}_{m,d} = h(\mathbf{A}) \mathbf{V}_{m,d} \begin{pmatrix} \hat{\mathbf{s}}_{m,d,0} \\ \vdots \\ \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \end{pmatrix} \\ &= \mathbf{V}_{m,d} \left[h(\mathbf{J}_{R_{m,d}}(\lambda_m)) \begin{pmatrix} \hat{\mathbf{s}}_{m,d,0} \\ \vdots \\ \hat{\mathbf{s}}_{m,d,R_{m,d}-1} \end{pmatrix} \right] \end{aligned} \quad (22)$$

Since all N generalized eigenvectors of \mathbf{A} are linearly independent, all subspaces $\mathcal{S}_{m,d}$ have zero intersections, and their dimensions add to N . Thus, the *spectral decomposition* (19) of the signal space \mathcal{S} is

$$\mathcal{S} = \bigoplus_{m=0}^{M-1} \bigoplus_{d=0}^{D_m-1} \mathcal{S}_{m,d}. \quad (23)$$

Graph Fourier Transform

The spectral decomposition (23) expands each signal $\mathbf{s} \in \mathcal{S}$ on the basis of the invariant subspaces of \mathcal{S} . Since we chose the generalized eigenvectors as bases of the subspaces $\mathcal{S}_{m,d}$, the expansion coefficients are given by

$$\mathbf{s} = \mathbf{V} \hat{\mathbf{s}}, \quad (24)$$

where \mathbf{V} is the generalized eigenvector matrix (38). The vector of expansion coefficients is given by

$$\hat{\mathbf{s}} = \mathbf{V}^{-1} \mathbf{s}. \quad (25)$$

The union of the bases of all spectral components $\mathcal{S}_{m,d}$, i.e., the basis of generalized eigenvectors, is called the *graph Fourier basis*. We call the expansion (25) of a signal \mathbf{s} into the

graph Fourier basis the *graph Fourier transform* and denote the graph Fourier transform matrix as

$$\mathbf{F} = \mathbf{V}^{-1}. \quad (26)$$

Following the conventions of classical DSP, we call the coefficients $\hat{\mathbf{s}}$ in (25) the *spectrum* of a signal \mathbf{s} . The *inverse graph Fourier transform* is given by (24); it reconstructs the signal from its spectrum.

Frequency Response of Graph Filters

The *frequency response* of a filter characterizes its effect on the frequency content of the input signal. Let us rewrite the filtering of \mathbf{s} by $h(\mathbf{A})$ using (21) and (24) as

$$\begin{aligned} \tilde{\mathbf{s}} &= h(\mathbf{A}) \mathbf{s} = \mathbf{F}^{-1} h(\mathbf{J}) \mathbf{F} \mathbf{s} = \mathbf{F}^{-1} h(\mathbf{J}) \hat{\mathbf{s}} \\ \Rightarrow \mathbf{F} \tilde{\mathbf{s}} &= h(\mathbf{J}) \hat{\mathbf{s}}. \end{aligned} \quad (27)$$

Hence, the spectrum of the output signal is the spectrum of the input signal modified by the block-diagonal matrix

$$h(\mathbf{J}) = \begin{pmatrix} h(\mathbf{J}_{r_{0,0}}(\lambda_0)) & & \\ & \ddots & \\ & & h(\mathbf{J}_{r_{M-1,D_{M-1}}}(\lambda_{M-1})) \end{pmatrix}, \quad (28)$$

so that the part of the spectrum corresponding to the invariant subspace $\mathcal{S}_{m,d}$ is multiplied by $h(\mathbf{J}_m)$. Hence, $h(\mathbf{J})$ in (28) represents the frequency response of the filter $h(\mathbf{A})$.

Notice that (27) also generalizes the *convolution theorem* from classical DSP [56] to arbitrary graphs.

Theorem 7: Filtering a signal is equivalent, in the frequency domain, to multiplying its spectrum by the frequency response of the filter.

Discussion

The connection (25) between the graph Fourier transform and the Jordan decomposition (39) highlights some desirable properties of representation graphs. For graphs with *diagonalizable* adjacency matrices \mathbf{A} , which have N linearly independent eigenvectors, the frequency response (28) of filters $h(\mathbf{A})$ reduces to a diagonal matrix with the main diagonal containing values $h(\lambda_m)$, where λ_m are the eigenvalues of \mathbf{A} . Moreover, for these graphs, Theorem 6 provides the closed-form expression (15) for the inverse graph Fourier transform $\mathbf{F}^{-1} = \mathbf{V}$. Graphs with symmetric (or Hermitian) matrices, such as undirected graphs, are always diagonalizable and, moreover, have orthogonal graph Fourier transforms: $\mathbf{F}^{-1} = \mathbf{F}^H$. This property has significant practical importance, since it yields a closed-form expression (15) for \mathbf{F} and \mathbf{F}^{-1} . Moreover, orthogonal transforms are well-suited for efficient signal representation, as we demonstrate in Section VI.

DSP_G is consistent with the classical DSP theory. As mentioned in Section II, finite discrete periodic time series are represented by the directed graph in Fig. 1(a). The corresponding adjacency matrix is the $N \times N$ circulant matrix (2). Its eigendecomposition (and hence, Jordan decomposition) is

$$\mathbf{C}_N = \frac{1}{N} \mathbf{DFT}_N^{-1} \begin{pmatrix} e^{-j\frac{2\pi \cdot 0}{N}} & & \\ & \ddots & \\ & & e^{-j\frac{2\pi \cdot (N-1)}{N}} \end{pmatrix} \mathbf{DFT}_N,$$

where \mathbf{DFT}_N is the discrete Fourier transform matrix. Thus, as expected, the graph Fourier transform is $\mathbf{F} = \mathbf{DFT}_N$. Furthermore, for a general filter $h(\mathbf{C}_N) = \sum_{\ell=0}^{N-1} h_\ell \mathbf{C}_N^\ell$, coefficients of the output $\hat{\mathbf{s}} = h(\mathbf{C}_N)\mathbf{s}$ are calculated as

$$\begin{aligned} \hat{s}_n &= h_n s_0 + \dots + h_0 s_n + h_{N-1} s_{n+1} + \dots + h_{n+1} s_{N-1} \\ &= \sum_{k=0}^{N-1} s_k h_{(n-k) \bmod N}. \end{aligned}$$

This is the standard circular convolution. Theorem 5 holds as well, with impulse response identical to filter taps: $\mathbf{u} = \mathbf{h}$.

Similarly, it has been shown in [45], [43] that unweighted line graphs similar to Fig. 1(a), but with undirected edges and different, non-periodic boundary conditions, give rise to all 16 types of discrete cosine and sine transforms as their graph Fourier transform matrices. Combined with [59], it can be shown that graph Fourier transforms for images on the lattice in Fig. 1(b) are different types of two-dimensional discrete cosine and sine transforms, depending on boundary conditions. This result serves as additional motivation for the use of these transforms in image representation and coding [60].

In discrete-time DSP, the concepts of filtering, spectrum, and Fourier transform have natural, physical interpretations. In DSP_G , when instantiated for various datasets, the interpretation of these concepts may be drastically different and not immediately obvious. For example, if a representation graph reflects the proximity of sensors in some metric (such as time, space, or geodesic distance), and the dataset contains sensor measurements, then filtering corresponds to linear recombination of related measurements and can be viewed as a graph form of regression analysis with constant coefficients. The graph Fourier transform then decomposes signals over equilibrium points of this regression. On the other hand, if a graph represents a social network of individuals and their communication patterns, and the signal is a social characteristic, such as an opinion or a preference, then filtering can be viewed as diffusion of information along established communication channels, and the graph Fourier transform characterizes signals in terms of stable, unchangeable opinions or preferences.

VI. APPLICATIONS

We consider several applications of DSP_G to data processing. These examples illustrate the effectiveness of the framework in standard DSP tasks, such as predictive filtering and efficient data representation, as well as demonstrate that the framework can tackle problems less common in DSP, such as data classification and customer behavior prediction.

Linear Prediction

Linear prediction (LP) is an efficient technique for representation, transmission, and generation of time series [61]. It is used in many applications, including power spectral estimation and direction of arrival analysis. Two main steps of LP are the construction of a prediction filter and the generation of an (approximated) signal, implemented, respectively, with forward and backward filters, shown in Fig. 2. The forward (prediction) filter converts the signal into a *residual*, which is

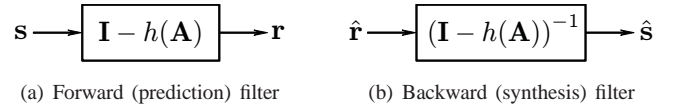


Fig. 2. Components of linear prediction.

then closely approximated, for example, by a white noise–flat power spectrum signal or efficient quantization with few bits. The backward (synthesis) filter constructs an approximation of the original signal from the approximated residual.

Using the DSP_G , we can extend LP to graph signals. We illustrate it with the dataset [62] of daily temperature measurements from sensors located near 150 major US cities. Data from each sensor is a separate time series, but encoding it requires buffering measurements from multiple days before they can be encoded for storage or transmission. Instead, we build a LP filter on a graph to encode daily snapshots of all 150 sensor measurements.

We construct a representation graph $G = (\mathcal{V}, \mathbf{A})$ for the sensor measurements using geographical distances between sensors. Each sensor corresponds to a node v_n , $0 \leq n < 150$, and is connected to K nearest sensors with undirected edges weighted by the normalized inverse exponents of the squared distances: if d_{nm} denotes the distance between the n th and m th sensors⁵ and $m \in \mathcal{N}_n$, then

$$A_{n,m} = \frac{e^{-d_{nm}^2}}{\sqrt{\sum_{k \in \mathcal{N}_n} e^{-d_{nk}^2} \sum_{\ell \in \mathcal{N}_m} e^{-d_{m\ell}^2}}}. \quad (29)$$

For each snapshot \mathbf{s} of $N = 150$ measurements, we construct a prediction filter $h(\mathbf{A})$ with L taps by minimizing the energy of the residual $\mathbf{r} = \mathbf{s} - h(\mathbf{A})\mathbf{s} = (\mathbf{I}_N - h(\mathbf{A}))\mathbf{s}$. We set $h_0 = 0$ to avoid the trivial solution $h(\mathbf{A}) = \mathbf{I}$, and obtain

$$(h_1 \quad \dots \quad h_{L-1})^T = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{s}.$$

Here, $\mathbf{B} = (\mathbf{A}\mathbf{s} \quad \dots \quad \mathbf{A}^{L-1}\mathbf{s})$ is a $N \times (L-1)$ matrix. The residual energy $\|\mathbf{r}\|_2^2$ is relatively small compared to the energy of the signal \mathbf{s} , since shifted signals are close approximations of \mathbf{s} , as illustrated in Fig. 3. This phenomenon provides the intuition for the graph shift: if the graph represents a similarity relation, as in this example, then the shift replaces each signal sample with a sum of related samples with more similar samples weighted heavier than less similar ones.

The residual \mathbf{r} is then quantized using B bits, and the quantized residual $\hat{\mathbf{r}}$ is processed with the inverse filter to synthesize an approximated signal $\hat{\mathbf{s}} = (\mathbf{I}_N - h(\mathbf{A}))^{-1} \hat{\mathbf{r}}$.

We considered graphs with $1 \leq K \leq 15$ nearest neighbors, and for each K constructed optimal prediction filters with $2 \leq L \leq 10$ taps. As shown in Fig. 4, the lowest and highest errors $\|\mathbf{s} - \hat{\mathbf{s}}\|_2 / \|\mathbf{s}\|_2$ were obtained for $K = 11$ and $L = 3$, and for $K = 8$ and $L = 9$. During the experiments, we observed that graphs with few neighbors (approximately, $3 \leq K \leq 7$) lead to lower errors when prediction filters have impulse responses of

⁵The construction of representation graphs for datasets is an important research problem and deserves a separate discussion that is beyond the scope of this paper. The procedure we use here is a popular choice for construction of similarity graphs based on distances between nodes [21], [30], [35].

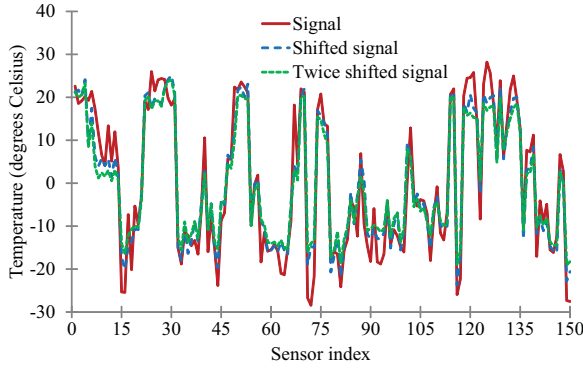


Fig. 3. A signal representing a snapshot of temperature measurements from $N = 150$ sensors. Shifting the signal produces signals similar to the original.

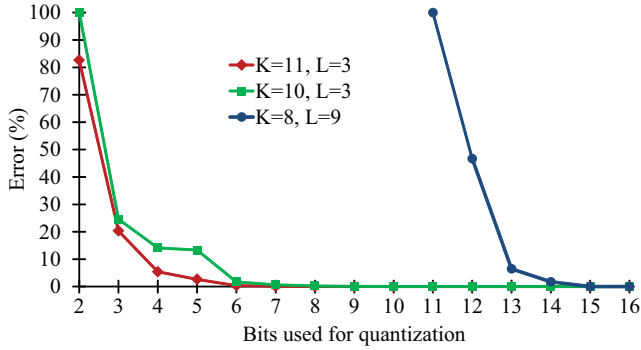


Fig. 4. Average approximation errors $\|s - \hat{s}\|_2 / \|s\|_2$ for LP coding of 365 signals s representing daily temperature snapshots. Graphs with $1 \leq K \leq 15$ nearest neighbors for each sensor were analyzed, and filters with $2 \leq L \leq 10$ taps were constructed. The residual was quantized using $1 \leq B \leq 16$ bits. The lowest, second lowest, and highest errors were obtained, respectively for $K = 11$ and $L = 3$, $K = 10$ and $L = 3$, and $K = 8$ and $L = 9$.

medium length ($4 \leq L \leq 6$), while graphs with $7 \leq K \leq 11$ neighbors yield lower errors for $3 \leq L \leq 5$. Using larger values of K and L leads to large errors. This tendency may be due to overfitting filters to signals, and demonstrates that there exists a trade-off between graph and filter parameters.

Signal Compression

Efficient signal representation is required in multiple DSP areas, such as storage, compression, and transmission. Some widely-used techniques are based on expanding signals into orthonormal bases with the expectation that most information is captured with few basis functions. The expansion coefficients are calculated using an orthogonal transform. If the transform represents a Fourier transform in some model, it means that signals are sparse in the frequency domain in this model, i.e., they contain only few frequencies. Some widely-used image compression standards, such as JPEG and JPEG 2000, use orthogonal expansions implemented, respectively, by discrete cosine and wavelet transforms [60].

As discussed in the previous example, given a signal s on a graph $G = (\mathcal{V}, \mathbf{A})$, where \mathbf{A} reflects similarities between data elements, the shifted signal $\mathbf{A}s$ can be a close approximation of s , up to a scalar factor: $\mathbf{A}s \approx \rho s$. This is illustrated in

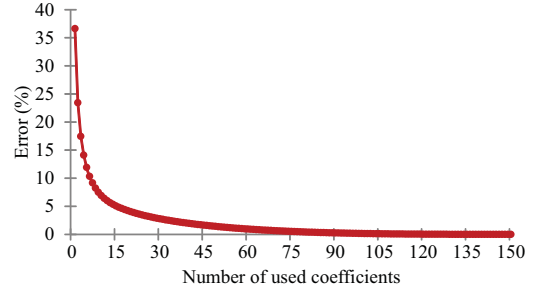


Fig. 5. Average reconstruction error $\|s - \hat{s}\|_2 / \|s\|_2$ for the compression of 365 daily temperature snapshots based on the graph Fourier transform using $1 \leq C \leq N$ coefficients.

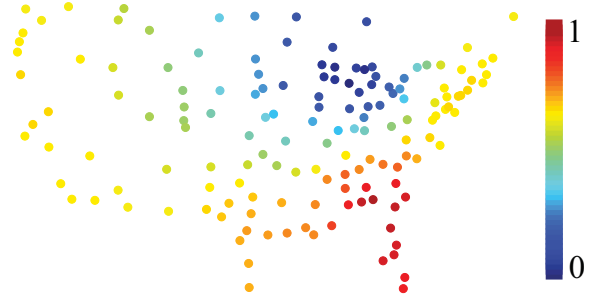


Fig. 6. The Fourier basis vector that captures most energy of temperature measurements reflects the relative distribution of temperature across the mainland United States. The coefficients are normalized to the interval $[0, 1]$.

Fig. 3, where $\rho \approx 1$. Hence, s can be effectively expressed as a linear combination of a few [generalized] eigenvectors of \mathbf{A} .

Consider the above dataset of temperature measurements. The matrix \mathbf{A} in (29) is symmetric by construction, hence its eigenvectors form an orthonormal basis, and the graph Fourier transform matrix \mathbf{F} is orthogonal. In this case, we can compress each daily update s of $N = 150$ measurements by keeping only the C spectrum coefficients (25) \hat{s}_n with largest magnitudes. Assuming that $|\hat{s}_0| \geq |\hat{s}_1| \geq \dots \geq |\hat{s}_{N-1}|$, the signal reconstructed after compression is

$$\tilde{s} = \mathbf{F}^{-1} (\hat{s}_0, \dots, \hat{s}_{C-1}, 0, \dots, 0)^T. \quad (30)$$

Fig. 5 shows the average reconstruction errors obtained by retaining $1 \leq C \leq N$ spectrum coefficients.

This example also provides interesting insights into the temperature distribution pattern in the United States. Consider the Fourier basis vector that most frequently (for 217 days out of 365) captures most energy of the snapshot s , i.e., yields the spectrum coefficient \hat{s}_0 in (30). Fig. 6 shows the vector coefficients plotted on the representation graph according to the sensors' geographical coordinates, so the graph naturally takes the shape of the mainland US. It can be observed that this basis vector reflects the relative temperature distribution across the US: the south-eastern region is the hottest one, and the Great Lakes region is the coldest one [63].

Data Classification

Classification and labeling are important problems in data analysis. These problems have traditionally been studied in

machine learning [64], [65]. Here, we propose a novel data classification algorithm by demonstrating that a classifier system can be interpreted as a filter on a graph. Thus, the construction of an optimal classifier can be viewed and studied as the design of an adaptive graph filter. Our algorithm scales linearly with the data size N , which makes it an attractive alternative to existing classification methods based on neural networks and support vector machines.

Our approach is based on the label propagation [66], [67], which is a simple, yet efficient technique that advances known labels from labeled graph nodes along edges to unlabeled nodes. Usually this propagation is modeled as a stationary discrete-time Markov process [68], and the graph adjacency matrix is constructed as a probability transition matrix, i.e., $\mathbf{A}_{n,m} \geq 0$ for all n, m , and $\mathbf{A}\mathbf{1}_N = \mathbf{1}_N$, where $\mathbf{1}_N$ is a column vector of N ones. Initially known labels determine the initial probability distribution \mathbf{s} . For a binary classification problem with only two labels, the resulting labels are determined by the distribution $\tilde{\mathbf{s}} = \mathbf{A}^P \mathbf{s}$. If $\tilde{s}_n \leq 1/2$, node v_n is assigned one label, and otherwise the other. The number P of propagations is determined heuristically.

Our DSP_G approach has two major distinctions from the original label propagation. First, we do not require \mathbf{A} to be a stochastic matrix. We only assume that edge weights $\mathbf{A}_{k,m} \geq 0$ are non-negative and indicate similarity or dependency between nodes. In this case, nodes with positive labels $\tilde{s}_n > 0$ are assigned to one class, and with negative labels to another. Second, instead of propagating labels as in a Markov chain, we construct a filter $h(\mathbf{A})$ that produces labels

$$\tilde{\mathbf{s}} = h(\mathbf{A})\mathbf{s}. \quad (31)$$

The following example illustrates our approach. Consider a set of $N = 1224$ political blogs on the Web that we wish to classify as “conservative” or “liberal” based on their context [69]. Reading and labeling each blog is very time-consuming. Instead, we read and label only a few blogs, and use these labels to adaptively build a filter $h(\mathbf{A})$ in (31).

Let signal \mathbf{s} contain initially known labels, where “conservative,” “liberal,” and unclassified blogs are represented by values $s_n = +1, -1$, and 0 , respectively. Also, let signal \mathbf{t} contain *training* labels, a subset of known labels from \mathbf{s} . Both \mathbf{s} and \mathbf{t} are represented by a graph $G = (\mathcal{V}, \mathbf{A})$, where node v_n containing the label of the n th blog, and edge $\mathbf{A}_{n,m} = 1$ if and only if there is a hyperlink reference from the n th to the m th blog; hence the graph is directed. Observe that the discovery of hyperlink references is a fast, easily automated task, unlike reading the blogs. An example subgraph for 50 blogs is shown in Fig. 1(d).

Recall that the graph shift \mathbf{A} replaces each signal coefficient with a weighted combination of its neighbors. In this case, processing training labels \mathbf{t} with the filter

$$\mathbf{I}_N + h_1 \mathbf{A} \quad (32)$$

produces new labels $\tilde{\mathbf{t}} = \mathbf{t} + h_1 \mathbf{A} \mathbf{t}$. Here, every node label is adjusted by a scaled sum of its neighbors’ labels. The parameter h_1 can be interpreted as the “confidence” in our knowledge of current labels: the higher the confidence h_1 , the

Blog selection method	Fraction of initially known labels		
	2%	5%	10%
Random	87%	93%	95%
Most hyperlinks	93%	94%	95%

TABLE I
ACCURACY OF BLOG CLASSIFICATION USING ADAPTIVE FILTERS.

more neighbors’ labels should affect the current labels. We restrict the value of h_1 to be positive.

Since the sign of each label indicates its class, label \tilde{t}_n is incorrect if its sign differs from s_n , or $\tilde{t}_n s_n \leq 0$ for $s_n \neq 0$. We determine the optimal value of h_1 by minimizing the total error, given by the number of incorrect and undecided labels. This is done in linear time proportional to the number of initially known labels $s_n \neq 0$, since each constraint

$$\tilde{t}_n s_n = \left(t_n + h_1 \sum_{k \in \mathcal{N}_n} t_k \right) s_n \leq 0 \quad (33)$$

is a linear inequality constraint on h_1 .

To propagate labels to all nodes, we repeatedly feed them through P filters (32) of the form $h^{(p)}(\mathbf{A}) = \mathbf{I}_N + h_p \mathbf{A}$, each time optimizing the value of h_p using the greedy approach discussed above. The obtained adaptive classification filter is

$$h(\mathbf{A}) = (\mathbf{I}_N + h_P \mathbf{A})(\mathbf{I}_N + h_{P-1} \mathbf{A}) \dots (\mathbf{I}_N + h_1 \mathbf{A}). \quad (34)$$

In experiments, we set $P = 10$, since we observed that filter (34) converges quickly, and in many cases, $h_p = 0$ for $p > 10$, which is similar to the actual graph’s diameter of 8. After the filter (34) is constructed, we apply it to all known labels \mathbf{s} , and classify all N nodes based on the signs of resulting labels $\tilde{\mathbf{s}} = h(\mathbf{A})\mathbf{s}$.

In our experiments, we considered two methods for selecting nodes to be labeled initially: random selection, and selection of blogs with most hyperlinks. As Table I shows, our algorithm achieves high accuracy for both methods. In particular, assigning initial labels \mathbf{s} to only 2% of blogs with most hyperlinks leads to the correct classification of 93 % of unlabeled blogs.

Customer Behavior Prediction

The adaptive filter design discussed in the previous example can be applied to other problems as well. Moreover, the linear computational cost of the filter design makes the approach easily scalable for the analysis of large datasets. Consider an example of a mobile service provider that is interested in keeping its customers. The company wants to predict which users will stop using their services in the near future, and offer them incentives for staying with the provider (improved call plan, discounted phones, etc.). In particular, based on their past behavior, such as number and length of calls within the network, the company wants to predict whether customers will stop using the services in the next month.

This problem can be formulated similarly to the previous example. In this case, the value at node v_n of the representation

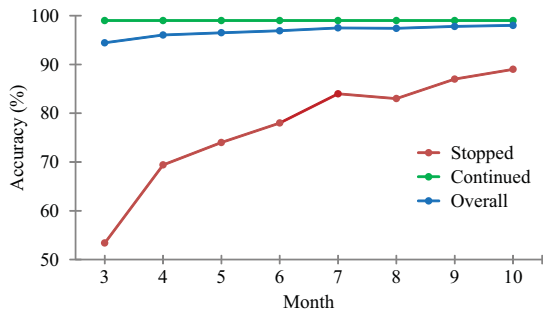


Fig. 7. The accuracy of behavior prediction for customers of a mobile provider. Predictions for customers who stopped using the provider and those who continued are evaluated separately, and then combined into the overall accuracy.

graph $G = (\mathcal{V}, \mathbf{A})$ indicates the probability that the n th customer will not use the provider services in the next 30 days. The weight of a directed edge from node v_n to v_m is the fraction of time the n th customer called and talked to the m th customer; i.e., if $T_{n,m}$ indicates the total time the n th customer called and talked to the m th customer in the past until the present moment, then

$$\mathbf{A}_{n,m} = \frac{T_{n,m}}{\sum_{k \in \mathcal{N}_n} T_{n,k}}.$$

The initial input signal \mathbf{s} has $s_n = 1$ if the customer has already stopped using the provider, and $s_n = 0$ otherwise. As in the previous example, we design a classifier filter (34); we set $P = 10$. We then process the entire signal \mathbf{s} with the designed filter obtaining the output signal $\tilde{\mathbf{s}}$ of the predicted probabilities; we conclude that the n th customer will stop using the provider if $\tilde{s}_n \geq 1/2$, and will continue if $\tilde{s}_n < 1/2$.

In our preliminary experiments, we used a ten-month-long call log for approximately 3.5 million customers of a European mobile service provider, approximately 10% of whom stopped using the provider during this period⁶. Fig. 7 shows the accuracy of predicting customer behavior for months 3-10 using filters with at most $L \leq 10$ taps. The accuracy reflects the ratio of correct predictions for all customers, the ones who stop using the service and the ones who continue; it is important to correctly identify both classes, so the provider can focus on the proper set of customers. As can be seen from the results, the designed filters achieve high accuracy in the prediction of customer behavior. Unsurprisingly, the prediction accuracy increases as more information becomes available, since we optimize the filter for month K using cumulative information from preceding $K - 1$ months.

VII. CONCLUSIONS

We have proposed DSP_G, a novel DSP theory for datasets whose underlying similarity or dependency relations are represented by arbitrary graphs. Our framework extends fundamental DSP structures and concepts, including shift, filters, signal and filter spaces, spectral decomposition, spectrum, Fourier transform, and frequency response, to such datasets

⁶We use a large dataset on Call Detailed Records (CDRs) from a large mobile operator in one European country, which we call EURMO for short.

by viewing them as signals indexed by graph nodes. We demonstrated that DSP_G is a natural extension of the classical time-series DSP theory, and traditional definitions of the above DSP concepts and structures can be obtained using a graph representing discrete time series. We also provided example applications of DSP_G to various social science applications, and our experimental results demonstrated the effectiveness of using the DSP_G framework for datasets of different nature.

Acknowledgment

We thank EURMO, CMU Prof. Pedro Ferreira, and the iLab at CMU Heinz College for granting us access to EURMO CDR database and related discussions.

APPENDIX A: MATRIX DECOMPOSITION AND PROPERTIES

We review relevant properties of the Jordan normal form, and the characteristic and minimal polynomial of a matrix $\mathbf{A} \in \mathbb{C}^{N \times N}$; for a thorough review, see [57], [58].

Jordan Normal Form

Let $\lambda_0, \dots, \lambda_{M-1}$ denote $M \leq N$ distinct eigenvalues of \mathbf{A} . Let each eigenvalue λ_m have D_m linearly independent eigenvectors $\mathbf{v}_{m,0}, \dots, \mathbf{v}_{m,D_m-1}$. The D_m is the *geometric multiplicity* of λ_m . Each eigenvector $\mathbf{v}_{m,d}$ generates a *Jordan chain* of $R_{m,d} \geq 1$ linearly independent *generalized eigenvectors* $\mathbf{v}_{m,d,r}$, $0 \leq r < R_{m,d}$, where $\mathbf{v}_{m,d,0} = \mathbf{v}_{m,d}$, that satisfy

$$(\mathbf{A} - \lambda_m \mathbf{I})\mathbf{v}_{m,d,r} = \mathbf{v}_{m,d,r-1}. \quad (35)$$

For each eigenvector $\mathbf{v}_{m,d}$ and its Jordan chain of length $R_{m,d}$, we define a *Jordan block* matrix of dimension $R_{m,d}$ as

$$\mathbf{J}_{r_{m,d}}(\lambda_m) = \begin{pmatrix} \lambda_m & 1 & & \\ & \lambda_m & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_m \end{pmatrix} \in \mathbb{C}^{R_{m,d} \times R_{m,d}}. \quad (36)$$

Thus, each eigenvalue λ_m is associated with D_m Jordan blocks, each with dimension $R_{m,d}$, $0 \leq d < D_m$. Next, for each eigenvector $\mathbf{v}_{m,d}$, we collect its Jordan chain into a $N \times R_{m,d}$ matrix

$$\mathbf{V}_{m,d} = (\mathbf{v}_{m,d,0} \quad \dots \quad \mathbf{v}_{m,d,R_{m,d}-1}). \quad (37)$$

We concatenate all blocks $\mathbf{V}_{m,d}$, $0 \leq d < D_m$ and $0 \leq m < M$, into one block matrix

$$\mathbf{V} = (\mathbf{V}_{0,0} \quad \dots \quad \mathbf{V}_{M-1,D_{M-1}}), \quad (38)$$

so that $\mathbf{V}_{m,d}$ is at position $\sum_{k=0}^{m-1} D_k + d$ in this matrix. Then, matrix \mathbf{A} has the *Jordan decomposition*

$$\mathbf{A} = \mathbf{V} \mathbf{J} \mathbf{V}^{-1}, \quad (39)$$

where the block-diagonal matrix

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{R_{0,0}}(\lambda_0) & & \\ & \ddots & \\ & & \mathbf{J}_{R_{M-1,D_{M-1}}}(\lambda_{M-1}) \end{pmatrix} \quad (40)$$

is called the *Jordan normal form* of \mathbf{A} .

Minimal and Characteristic Polynomials

The *minimal polynomial* of matrix \mathbf{A} is the monic polynomial of smallest possible degree that satisfies $m_{\mathbf{A}}(\mathbf{A}) = \mathbf{0}_N$. Let $R_m = \max\{R_{m,0}, \dots, R_{m,D_m-1}\}$ denote the maximum length of Jordan chains corresponding to eigenvalue λ_m . Then the minimal polynomial $m_{\mathbf{A}}(x)$ is given by

$$m_{\mathbf{A}}(x) = (x - \lambda_0)^{R_1} \dots (x - \lambda_{M-1})^{R_{M-1}}. \quad (41)$$

The *index* of λ_m is R_m , $1 \leq m < M$. Any polynomial $p(x)$ that satisfies $p(\mathbf{A}) = \mathbf{0}_N$, is a polynomial multiple of $m_{\mathbf{A}}(x)$, i.e., $p(x) = q(x)m_{\mathbf{A}}(x)$. The degree of the minimal polynomial satisfies

$$\deg m_{\mathbf{A}}(x) = N_{\mathbf{A}} = \sum_{m=0}^{M-1} R_m \leq N. \quad (42)$$

The *characteristic polynomial* of the matrix \mathbf{A} is defined as

$$p_{\mathbf{A}}(x) = \det(\lambda \mathbf{I} - \mathbf{A}) = (x - \lambda_0)^{A_0} \dots (x - \lambda_{M-1})^{A_{M-1}}. \quad (43)$$

Here: $A_m = R_{m,0} + \dots + R_{m,D_m-1}$ for $0 \leq m < M$, is the *algebraic multiplicity* of λ_m ; $\deg p_{\mathbf{A}}(x) = N$; $p_{\mathbf{A}}(x)$ is a multiple of $m_{\mathbf{A}}(x)$; and $p_{\mathbf{A}}(x) = m_{\mathbf{A}}(x)$ if and only if the geometric multiplicity of each λ_m , $D_m = 1$, i.e., each eigenvalue λ_m has exactly one eigenvector.

APPENDIX B: PROOF OF THEOREM 2

We will use the following lemma to prove Theorem 2.

Lemma 1: For polynomials $h(x)$, $g(x)$, and $p(x) = h(x)g(x)$, and a Jordan block $J_r(\lambda)$ as in (36) of arbitrary dimension r and eigenvalue λ , the following equality holds:

$$h(\mathbf{J}_r(\lambda))g(\mathbf{J}_r(\lambda)) = p(\mathbf{J}_r(\lambda)). \quad (44)$$

Proof: The (i, j) th element of $h(\mathbf{J}_r(\lambda))$ is

$$h(\mathbf{J}_r(\lambda))_{i,j} = \frac{1}{(j-i)!} h^{(j-i)}(\lambda) \quad (45)$$

for $j \geq i$ and 0 otherwise, where $h^{(j-i)}(\lambda)$ is the $(j-i)$ th derivative of $h(\lambda)$ [58]. Hence, the (i, j) th element of $h(\mathbf{J}_r(\lambda))g(\mathbf{J}_r(\lambda))$ for $j < i$ is zero and for $j \geq i$ is

$$\begin{aligned} & \sum_{k=i}^j h(\mathbf{J}_r(\lambda))_{i,k} g(\mathbf{J}_r(\lambda))_{k,j} \\ &= \sum_{k=i}^j \frac{1}{(k-i)!} h^{(k-i)}(\lambda) \frac{1}{(j-k)!} g^{(j-k)}(\lambda) \\ &= \frac{1}{(j-i)!} \sum_{k=i}^j \binom{j-i}{k-i} h^{(k-i)}(\lambda) g^{(j-k)}(\lambda) \\ &= \frac{1}{(j-i)!} \sum_{m=0}^{j-i} \binom{j-i}{m} h^{(m)}(\lambda) g^{(j-i-m)}(\lambda) \\ &= \frac{1}{(j-i)!} (h(\lambda)g(\lambda))^{(j-i)}. \end{aligned} \quad (46)$$

Matrix equality (44) follows by comparing (46) with (45). ■

As before, let $\lambda_0, \dots, \lambda_{M-1}$ denote distinct eigenvalues of \mathbf{A} . Consider the Jordan decomposition (39) of \mathbf{A} . For each $0 \leq m < M$, select distinct numbers $\tilde{\lambda}_{m,0}, \dots, \tilde{\lambda}_{m,D_m-1}$, so

that all $\tilde{\lambda}_{m,d}$ for $0 \leq d < D_m$ and $0 \leq m < M$ are distinct. Construct the block-diagonal matrix

$$\tilde{\mathbf{J}} = \begin{pmatrix} \mathbf{J}_{R_{0,0}}(\tilde{\lambda}_{0,0}) & & \\ & \ddots & \\ & & \mathbf{J}_{R_{M-1,D_{M-1}}}(\tilde{\lambda}_{M-1,D_{M-1}-1}) \end{pmatrix}.$$

The Jordan blocks on the diagonal of $\tilde{\mathbf{J}}$ match the sizes of the Jordan blocks of \mathbf{J} in (40), but their elements are different.

Consider a polynomial $r(x) = r_0 + r_1x + \dots + r_{N-1}x^{N-1}$, and assume that $r(\tilde{\mathbf{J}}) = \mathbf{J}$. By Lemma 1, this is equivalent to

$$\begin{cases} r(\tilde{\lambda}_{m,d}) = \lambda_m, \\ r^{(1)}(\tilde{\lambda}_{m,d}) = 1 \\ r^{(i)}(\tilde{\lambda}_{m,d}) = 0, \text{ for } 2 \leq i < D_m \end{cases}$$

for all $0 \leq d < D_m$ and $0 \leq m < M$. This is a system of N linear equations with N unknowns r_0, \dots, r_{N-1} that can be uniquely solved using inverse polynomial interpolation [58].

Using (39), we obtain $\mathbf{A} = \mathbf{V} \mathbf{J} \mathbf{V}^{-1} = \mathbf{V} r(\tilde{\mathbf{J}}) \mathbf{V}^{-1} = r(\mathbf{V} \tilde{\mathbf{J}} \mathbf{V}^{-1}) = r(\tilde{\mathbf{A}})$. Furthermore, since all $\tilde{\lambda}_{m,d}$ are distinct numbers, their geometric multiplicities are equal to 1. As discussed in Appendix A, this is equivalent to $p_{\tilde{\mathbf{A}}}(x) = m_{\tilde{\mathbf{A}}}(x)$.

APPENDIX C: PROOF OF THEOREM 4

Lemma 1 leads to the construction procedure of the inverse polynomial $g(x)$ of $h(x)$, when it exists, and whose matrix representation satisfies $g(\mathbf{A})h(\mathbf{A}) = \mathbf{I}_N$. Observe that this condition, together with (44), is equivalent to

$$\begin{cases} h(\lambda_m)g(\lambda_m) = 1, \text{ for } 0 \leq m \leq M-1 \\ (h(\lambda_m)g(\lambda_m))^{(i)} = 0, \text{ for } 1 \leq i < R_m. \end{cases} \quad (47)$$

Here, R_m is the degree of the factor $(x - \lambda_m)^{R_m}$ in the minimal polynomial $m_{\mathbf{A}}(\lambda)$ in (41). Since values of $h(x)$ and its derivatives at λ_m are known, (47) amount to $N_{\mathbf{A}}$ linear equations with $N_{\mathbf{A}}$ unknowns. They have a unique solution if and only if $h(\lambda_m) \neq 0$ for all λ_m , and the coefficients $g_0, \dots, g_{M_{\mathbf{A}}-1}$ are then uniquely determined using inverse polynomial interpolation [58].

REFERENCES

- [1] C. Chamley, *Rational Herds: Economic Models of Social Learning*, Cambridge Univ. Press, 2004.
- [2] M. Jackson, *Social and Economic Networks*, Princeton Univ., 2008.
- [3] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*, Cambridge Univ. Press, 2010.
- [4] M. Newman, *Networks: An Introduction*, Oxford Univ. Press, 2010.
- [5] J. Whittaker, *Graphical Models in Applied Multivariate Statistics*, Wiley, 1990.
- [6] S. L. Lauritzen, *Graphical Models*, Oxford Univ. Press, 1996.
- [7] F. V. Jensen, *Bayesian Networks and Decision Graphs*, IEEE Comp. Soc. Press, 2001.
- [8] M. I. Jordan, "Graphical models," *Statistical Science (Special Issue on Bayesian Statistics)*, vol. 19, no. 1, pp. 140–155, 2004.
- [9] M. J. Wainwright and M. I. Jordan, *Graphical Models, Exponential Families, and Variational Inference*, Now Publishers Inc., 2008.
- [10] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.
- [11] D. Edwards, *Introduction to Graphical Modelling*, Springer, 2000.
- [12] J. Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer, 2nd edition, 2009.
- [13] R. Kindermann and J. L. Snell, *Markov Random Fields and Their Applications*, American Mathematical Society, 1980.

- [14] A.S. Willsky, "Multiresolution Markov models for signal and image processing," *Proc. IEEE*, vol. 90, no. 8, pp. 1396–1458, 2002.
- [15] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. Royal Stat. Soc.*, vol. 36, no. 2, pp. 192–236, 1974.
- [16] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*, Chapman & Hall, 1964.
- [17] D. Vats and J. M. F. Moura, "Finding Non-overlapping Clusters for Generalized Inference Over Graphical Models," *IEEE Trans. Signal Proc.*, vol. 60, no. 12, pp. 6368–6381, 2012.
- [18] M.I. Jordan, E.B. Sudderth, M. Wainwright, and A.S. Willsky, "Major advances and emerging developments of graphical models," *IEEE Signal Proc. Mag.*, vol. 27, no. 6, pp. 17–138, 2010.
- [19] J. F. Tenenbaum, V. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.
- [20] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.
- [21] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comp.*, vol. 15, no. 6, pp. 1373–1396, 2003.
- [22] D. L. Donoho and C. Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," *Proc. Nat. Acad. Sci.*, vol. 100, no. 10, pp. 5591–5596, 2003.
- [23] F. R. K. Chung, *Spectral Graph Theory*, AMS, 1996.
- [24] M. Belkin and P. Niyogi, "Using manifold structure for partially labeled classification," 2002.
- [25] M. Hein, J. Audibert, and U. von Luxburg, "From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians," in *COLT*, 2005, pp. 470–485.
- [26] E. Gine and V. Koltchinskii, "Empirical graph Laplacian approximation of Laplace–Beltrami operators: Large sample results," *IMS Lecture Notes Monograph Series*, vol. 51, pp. 238–259, 2006.
- [27] M. Hein, J. Audibert, and U. von Luxburg, "Graph Laplacians and their convergence on random neighborhood graphs," *J. Machine Learn.*, vol. 8, pp. 1325–1370, June 2007.
- [28] R. R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. J. Warner, and S. W. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps," *Proc. Nat. Acad. Sci.*, vol. 102, no. 21, pp. 7426–7431, 2005.
- [29] R. R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. J. Warner, and S. W. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Multiscale methods," *Proc. Nat. Acad. Sci.*, vol. 102, no. 21, pp. 7432–7437, 2005.
- [30] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Appl. Comp. Harm. Anal.*, vol. 21, no. 1, pp. 53–94, 2006.
- [31] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *IPSN*, 2004, pp. 1–10.
- [32] D. Ganesan, B. Greenstein, D. Estrin, J. Heidemann, and R. Govindan, "Multiresolution storage and search in sensor networks," *ACM Trans. Storage*, vol. 1, pp. 277–315, 2005.
- [33] R. Wagner, H. Choi, R. G. Baraniuk, and V. Delouille, "Distributed wavelet transform for irregular sensor network grids," in *IEEE SSP Workshop*, 2005, pp. 1196–1201.
- [34] R. Wagner, A. Cohen, R. G. Baraniuk, S. Du, and D.B. Johnson, "An architecture for distributed wavelet analysis and processing in sensor networks," in *IPSN*, 2006, pp. 243–250.
- [35] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *J. Appl. Comp. Harm. Anal.*, vol. 30, no. 2, pp. 129–150, 2011.
- [36] S. K. Narang and A. Ortega, "Local two-channel critically sampled filter-banks on graphs," in *ICIP*, 2010, pp. 333–336.
- [37] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Proc.*, vol. 60, no. 6, pp. 2786–2799, 2012.
- [38] R. Wagner, V. Delouille, and R. G. Baraniuk, "Distributed wavelet denoising for sensor networks," in *Proc. CDC*, 2006, pp. 373–379.
- [39] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Proc. ICASSP*, 2012, pp. 3921–3924.
- [40] A. Agaskar and Y. M. Lu, "Uncertainty principles for signals defined on graphs: Bounds and characterizations," in *Proc. ICASSP*, 2012.
- [41] A. Agaskar and Y. Lu, "A spectral graph uncertainty principle," Submitted for publication., June 2012.
- [42] M. Püschel and J. M. F. Moura, "The algebraic approach to the discrete cosine and sine transforms and their fast algorithms," *SIAM J. Comp.*, vol. 32, no. 5, pp. 1280–1316, 2003.
- [43] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory," <http://arxiv.org/abs/cs.IT/0612077>.
- [44] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory: Foundation and 1-D time," *IEEE Trans. Signal Proc.*, vol. 56, no. 8, pp. 3572–3585, 2008.
- [45] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory: 1-D space," *IEEE Trans. Signal Proc.*, vol. 56, no. 8, pp. 3586–3599, 2008.
- [46] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory: Cooley-Tukey type algorithms for DCTs and DSTs," *IEEE Trans. Signal Proc.*, vol. 56, no. 4, pp. 1502–1521, 2008.
- [47] A. Sandryhaila, J. Kovacevic, and M. Püschel, "Algebraic signal processing theory: Cooley-Tukey type algorithms for polynomial transforms based on induction," *SIAM J. Matrix Analysis and Appl.*, vol. 32, no. 2, pp. 364–384, 2011.
- [48] M. Püschel and M. Rötteler, "Algebraic signal processing theory: 2-D hexagonal spatial lattice," *IEEE Trans. on Image Proc.*, vol. 16, no. 6, pp. 1506–1521, 2007.
- [49] A. Sandryhaila, J. Kovacevic, and M. Püschel, "Algebraic signal processing theory: 1-D Nearest-neighbor models," *IEEE Trans. on Signal Proc.*, vol. 60, no. 5, pp. 2247–2259, 2012.
- [50] A. Sandryhaila, S. Saba, M. Püschel, and J. Kovacevic, "Efficient compression of QRS complexes using Hermite expansion," *IEEE Trans. on Signal Proc.*, vol. 60, no. 2, pp. 947–955, 2012.
- [51] A. Sandryhaila and J. M. F. Moura, "Nearest-neighbor image model," in *Proc. ICIP*, 2012, to appear.
- [52] B. A. Miller, N. T. Bliss, and P. J. Wolfe, "Toward signal processing theory for graphs and non-Euclidean data," in *Proc. ICASSP*, 2010, pp. 5414–5417.
- [53] B. A. Miller, M. S. Beard, and N. T. Bliss, "Matched filtering for subgraph detection in dynamic networks," in *Proc. SSP*, 2011, pp. 509–512.
- [54] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Graph Fourier transform," submitted for publication.
- [55] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Graph filters," submitted for publication.
- [56] A. V. Oppenheim, R. W. Schaffer, and J. R. Buck, *Discrete-Time Signal Processing*, Prentice Hall, 2nd edition, 1999.
- [57] F. R. Gantmacher, *Matrix Theory*, vol. I, Chelsea, 1959.
- [58] P. Lancaster and M. Tismenetsky, *The Theory of Matrices*, Academic Press, 2nd edition, 1985.
- [59] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing*, Prentice Hall, 1983.
- [60] A. Bovik, *Handbook of Image and Video Processing*, Academic Press, 2nd edition, 2005.
- [61] P. P. Vaidyanathan, *The Theory of Linear Prediction*, Morgan and Claypool, 2008.
- [62] "National climatic data center," 2011, <ftp://ftp.ncdc.noaa.gov/pub/data/g sod>.
- [63] "NCDC NOAA 1981-2010 climate normals," 2011, ncdc.noaa.gov/oa/climate/normal s/usnormals.html.
- [64] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.
- [65] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley, 2nd edition, 2000.
- [66] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. ICML*, 2003, pp. 58–65.
- [67] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," in *Proc. ICML*, 2006, pp. 985–992.
- [68] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, 4th edition, 2002.
- [69] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. election: Divided they blog," in *LinkKDD*, 2005.