CHAPTER

GRAPH SIGNAL PROCESSING 8

José M.F. Moura[a*]

*Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, United States**

8.1 INTRODUCTION

It is by now a cliché that data is everywhere, and we are and will be inundated by data. According to a 2014 study by IDC [1], in 2020 alone, all digital data created, replicated, and consumed will amount to 44 zetabytes. A zetabyte is 10^{21} bytes, so we will produce yearly data quantified by the number 44 followed by 21 zeros. To have a more physical feeling for this staggering amount of data, we compare it to a coarse estimate of the entire collection of books, reports, written texts, maps, images, audio recordings, and videos in the US Library of Congress (LOC) (charged with maintaining a copy of every book ever printed in the United States). This estimate may vary, but we consider it to be three petabytes [2]. Then, the 44 zetabytes of data produced worldwide in the year 2020 will be roughly equivalent to 15 million LOCs. IDC has recently updated this estimate to 163 zetabytes by 2025 [3,4]. But these data will be very different from the data we are traditionally concerned with in disciplines such as statistics, signal or image processing, computer vision, or machine learning. Beyond traditional time series, speech, audio, radio, radar, biomedical signals, or images and videos, these data arise from the 11 billion internet-of-things (IOT) devices in 2016 that are estimated to triplicate to 30 billion in 2020, from the activity of billions of cell phone users of the many service providers; from public and private urban transportation systems; in health care from the digital records of patients, providers, visits, exams, tests, results, costs, insurance, hospital procedures; from interactions among social network agents, corporate financial data, hyperlinked blogs, or tweeters, metabolic networks, protein interaction networks, just to mention some examples. These *Big Data* are often characterized by three, five, or seven V: *V*ariety, *V*olume, *V*elocity, *V*eracity, *V*ariability, *V*alue, and *V*isualization. In many contexts, these data are produced by scattered sources such as the thousands of webcams monitoring traffic in urban centers, i.e., the data are *distributed*. Besides numerical, the data can be Boolean, ordinal, or categorical. Finally, the data is unlikely to fit neatly in a table; in other words, the data are *un*structured. The goal of this chapter is to introduce data analytic tools to process these data that are much like the classical tools used with time series, images, or videos, but now applied to the variety of *un*structured *Big Data* of today.

[a] This work is partially supported by NSF grant CCF # 1513936.

Cooperative and Graph Signal Processing. https://doi.org/10.1016/B978-0-12-813677-5.00008-0

While this chapter focuses on the analytics of these unstructured data, which we will call *Data Science*, we start by contrasting the goal of the chapter and Data Science with two other close topics, namely, *Network Science* and *Network Processes*.

Network Science. In the last decade and a half, graphs have permeated the study of complex social, biological, and technological systems [5–8] of high dimensional data, or of the dynamics of networked processes such as epidemics, to refer to a few application domains. These studies often lead to first representing the application data by a graph, and then addressing the questions of interest by focusing on and analyzing the structure of the graph. This is the domain of *Network Science* [9] that has received considerable attention [6–8]. For an illustrative Network Science approach to problems, consider the rise and dominance of the Medici elite family in 15th century Florence through marriage, economics, and the patronage of the Medicean networks [10]. Marriage network data in [10] is partially shown [11] by the graph in Fig. 8.1 where nodes are families and edges are marital linkages among families. Reference [10] shows that the "in-betweenness centrality" [12] for the Medici family marriage network is $C_b = 0.362$ while for oligarch families it is $C_b = 0.184$ (see footnote 31 in [10]). Similarly, for economic relations, it is $C_b = 0.429$ for Medici families and $C_b = 0.198$ for other elite families [10]. Further, as noted in footnote 32 in [10], the Medici family is more like a star or spoke network, with sparse ties among families tied by marriage or economic relations to the Medici patriarch (average degree of these neighbors of the Medici being 2). This contrasts with oligarch families where the

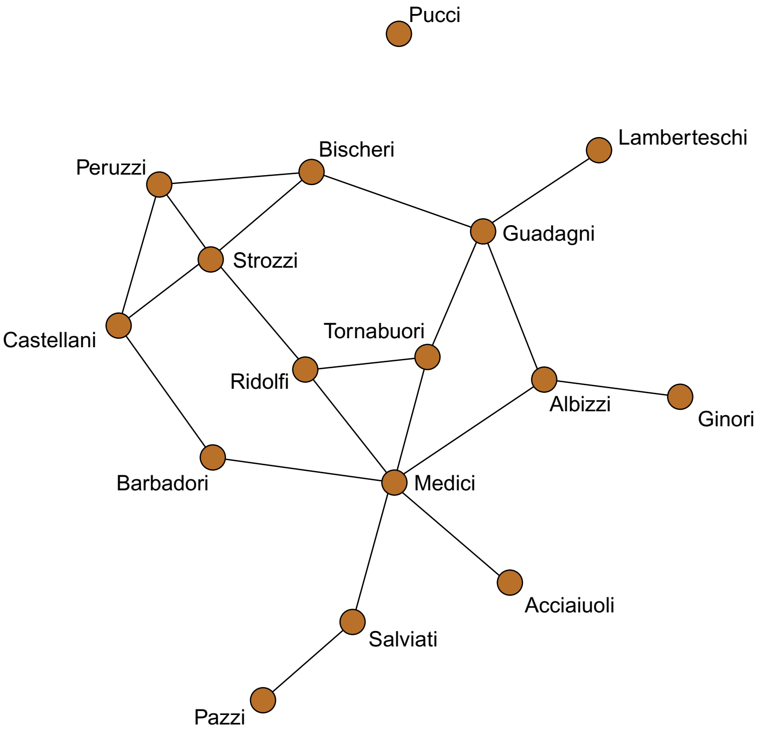

FIG. 8.1

Fifteenth-century Florentine marriages [11].

average degree of their neighbors could be as high as 6.5. This centrality explains the dependence of Medici partners on the Medici that played a significant intermediary role as facilitator of transactions among them. The point here is that, in Network Science, a graph represents the data, and, once the graph is available, the questions of interest are not addressed by analyzing the data, but rather by analyzing the structure of the graph. Network Science focuses on topological and structural properties of the network with quantities of interest such as in- and out-degrees of nodes, degree distribution, number of connected components, size of the giant component, graph diameter, and average shortest path length as well as parameters such as the clustering coefficient that measures network cohesiveness or various other centrality measures. Network models include the Erdös Rényi random graphs, small world graphs, preferential attachment and scale-free random graphs [13], regular graphs, and hub networks [6–8].

Network Processes. The spread of epidemics, fake news, memes, diffusion of opinions, and voting patterns are all examples of *network processes* of great interest in the social and life sciences, for example. These processes collect the state of all the agents (nodes) in the network. Their dynamics as the process evolves in time are governed by local rules of interaction among the agents. For example, in epidemics, a healed node becomes infected by coming into contact with another infected node [14–18]. Studies of these processes consider the emergence of global behaviors from these local interactions. These are difficult questions because the network state space is very large—for an interacting system of N agents and a binary state for each agent, the cardinality of this state space is 2^N. For a small network of $N = 200$, this is $2^{200} = 10^{60}$. This is a staggering large number that makes the analysis of general processes over general networks an almost impossible task. Often, the literature studies these systems for large time, large scale or network size (fluid dynamics) limit, restricting attention to global process parameters such as the fraction of nodes in a given state, say, the fraction of infected nodes, or the fraction of nodes adopting a particular opinion, or fraction of voters voting in the same candidate or party. Further, the literature frequently analyzes these systems under simple conditions that abstract out the network topology—full mixing or complete network where an agent can interact with any other agent. Corresponding limiting models lead to systems of ordinary differential equations governing the dynamics of the global quantities of interest. Our recent work in this area studies the dynamics of network processes for arbitrary network topologies. References [19,20] introduce the scaled network process under which the network process is a reversible Markov process [21], and for which we find its equilibrium probability distribution (time asymptotics) that explicitly exhibits the network topology through its adjacency matrix. From this equilibrium distribution, we can address global questions such as which configuration, i.e., network state, in the long run is more likely to occur, or structural questions such as which parts or substructures of the network are more likely to be infected. These questions depend on both the topology of the network and the local rules of interaction among agents.

In [22,23], we consider the qualitative behavior of these processes in the limit of large networks. Under this limit and appropriate conditions, the dynamics of global statistics, for example the fraction of infected nodes, follow nonlinear ordinary differential equations (NLODE). The qualitative behavior of these equations, requiring establishing the basins of attraction for the limit points of NLODEs, addresses issues such as which opinion or virus may survive when several compete for attention or multiple viruses cohabit the network [22,24].

Data Science. To remove the *un* from *un*structured, we assume that the data is indexed by the N nodes of a graph $G = (V, E)$. The edges in E of G represent dependencies among the data. The nodes in V can be thought of as the sources of data, for example, agents in a social network, individuals in a population, tweeters, or bloggers in networks of hyperlinked blogs [25]. The data at each node of V can itself be a time series, an image, a video, or other features collected or produced by the agent or characterizing the node. For example, we may be interested in a college cohort, say from Carnegie Mellon. We may abstract a network where the nodes are the students in the class and the edges express direct relations among them, for example, their friendships or degree of acquaintance. The data can be their grades, extracurricular activities, time spent studying, the particular dormitory where they live. These data relate to each student, and so the data are indexed by the nodes of the friendship network. We may then pursue the question of predicting the grade point average of current juniors in the fall of their senior year. Coupled to the available data associated with the lifestyle and course grades of each student, the underlying cohort network graph should lead to better predictions than treating the data independently of the social relations among students. We consider a second example. Fig. 8.2 on the left represents a small fragment (20,000 users) of a much larger network of subscribers of a cell phone service provider in a given month, say, month m. Nodes are users and their level of activity (who calls whom) is represented by edges among them. The data in this case indicate simply whether a user maintains service in a given month or drops the service; with reference to the left of Fig. 8.2, this binary valued data is given by the two colors labeling the nodes of the graph—blue users keep their service and red users drop or churn the service. Given the past history of subscribers who have churned in each month, the service provider is interested in predicting which subscribers are at risk of churning, i.e., which blue nodes in month m become red in month $m+1$. Because the churn rate is very small, this detection problem is extremely challenging, like finding a needle in a haystack. A successful solution (high probability of detecting churners while keeping the probability of false alarm very small) requires taking into account the underlying graph structure. In [26], we find a set of network features that are input to a classifier, leading to a probability of churn detection of 71% with a positive false alarm rate of 0.08%; see the right of Fig. 8.2. In other words, if the user base is 100 million and the average monthly churn rate is 1%, of the 1 million expected churners, the detector in [26] correctly finds about 700,000 of these potential churners, missing 300,000 of them while misclassifying only 80,000 as churners of the remaining 99,000,000 subscribers. The analytics of the data indexed by nodes of a graph G is the purview of *Data Science* that we consider in this chapter, in particular, we study graph signal processing (GSP) that extends to graph-based data the methods developed over the last 60 years for time and image signals. We introduce GSP by building it from classical discrete signal processing (DSP). To do this, we will provide in section 8.3 a quick abridged review of DSP concepts. But first we briefly review related literature.

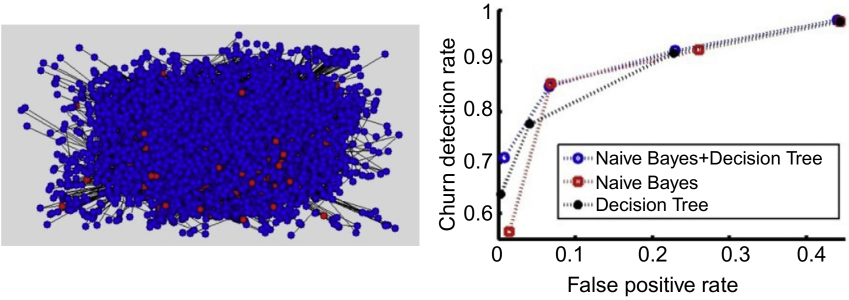

FIG. 8.2

Left: Snippet of a cell phone service provider user network in a given month: *blue dots* (dark gray in print versions) keep the service and *red dots* (light gray in print versions) drop the service in the month of service. Right: Churn detection error performance from [26].

8.2 BRIEF REVIEW OF THE LITERATURE

GSP studies data indexed by graphs. Other approaches that process data indexed by graphs include Markov random fields and, in particular, Gauss-Markov random fields [27–32], graphical models [33–37], data reduction approaches such as [38–41], spectral graph clustering [42–45], or geometric diffusion [46–48], to mention a few. Work on extending wavelets to irregularly spaced data and in sensor networks [49–51] is closer to the perspective of GSP. Reference [52] studies compressed sensing for networked data and [53–56] explore a number of issues such as wavelets on graphs, denoising, and sampling on graphs.

In [57–60], we developed algebraic signal processing for time signals and space signals. The algebraic signal processing theory introduces a signal model as a triplet of an algebra of filters, a signal module, and a linear mapping. We showed that this signal model can be defined under certain conditions by an appropriate definition of the shift filter. The algebraic signal processing leads in a principled way to GSP as we developed in [61–63]. An approach to GSP that uses the graph Laplacian is in [64].

To avoid a too mathematical introduction to GSP, we introduce GSP by illustrating how to cast DSP in the context of GSP. To do this, we start with a brief refresher on DSP in the next section.

8.3 DSP: A QUICK REFRESHER

Discrete signal processing (DSP) [65–69] studies time or image signals.[1] We briefly review time signals and their representations, filtering, delay and impulse response, frequency, spectrum, Fourier, z-transform, and related concepts.

We start with N (complex valued) numbers $\mathcal{S} = \left\{s_{\alpha_0}, \ldots, s_{\alpha_{N-1}}\right\}$. To define a time signal, we need an ordering of these numbers so that we know which number precedes and which number succeeds a given number. Let such ordering be given by the N-tuple $s = (s_0, \ldots, s_{N-1})$. Tuples are ordered. We refer to the indexing parameter n as time, and time takes values $n = 0, \ldots, N-1$. Entry s_n of s is the signal sample s_n at time n that precedes the sample s_{n+1} and succeeds sample s_{n-1}. The N-tuple is usually referred to as the (finite) discrete time signal or sequence. Common (abuse of) notation refers to the signal as simply the samples s_n or $s[n]$, where the time interval $n = 0, \ldots, N-1$ is taken for granted.

Remark. To avoid details with the finiteness of time, we assume the signal is extended to the left of $n = 0$ and right of $n = N-1$. Of the many possible extensions, we consider periodic extensions whereby we reduce $s_N = s_0$ and more generally $s_n = s_{n \bmod N}$. ■

The z-transform provides a second useful representation of the signal. The z-transform of the (finite) signal s is then

$$s(z) = s_0 z^0 + s_1 z^{-1} + \cdots + s_{N-1} z^{-(N-1)}.$$

The z-transform is not to be interpreted as a polynomial in the complex variable z. The symbol z^{-1} is a place holder and stands for the delay. The power z^{-n} is the nth delay.

[1] We consider here only linear DSP with a finite number N of samples.

When we develop GSP, we work with vector notation $\mathbf{s} = [s_0 \cdots s_{N-1}]^\top$, so, we recast DSP in vector notation also. The time signal is now interpreted as a vector in $\mathbb{C}^N$. A vector collects *all* signal samples; time is hidden or represented by the index of the coordinates of the vector.

Because $\mathbb{C}^N$ is a vector space, it has a basis with respect to which we can represent any vector as a linear combination of the vectors in the basis. We choose first the standard basis $E = \{e_0, \ldots, e_{N-1}\}$. This is an ordered basis. Using this standard basis in the vector space $\mathbf{C}^N$, the signal $\mathbf{s}$ is written as

$$\mathbf{s} = s_0 \underbrace{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\mathbf{e}_1} + \cdots + s_{N-1} \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}}_{\mathbf{e}_N},$$

where the basis vector e_n is represented in the standard basis by the standard vector $\mathbf{e}_{n+1}$.[2] This equation explicitly indicates that, in the vector representation of the signal, the sampled signal at time $n = 0$ is $s_0\,[1\,0 \cdots 0]^\top$, and more generally at time n it is $s_n\mathbf{e}_n$. The vectors of the standard basis are the (unit) impulses; for example, $\mathbf{e}_n$ is the impulse signal centered at time n, zero everywhere except at time n where it is one. So, the representation above gives the signal s as a linear combination of the unit impulses $\mathbf{e}_n$. This representation in matrix-vector form is

$$\mathbf{s} = \underbrace{\left[\mathbf{e}_1\, \mathbf{e}_2\, \cdots\, \mathbf{e}_{N-1}\right]}_{\mathbf{E}=\mathbf{I}_N} \underbrace{\begin{bmatrix} s_0 \\ s_1 \vdots \\ s_{N-1} \end{bmatrix}}_{\mathbf{s}_E},$$

where $\mathbf{E}$ is the matrix of the standard basis vectors that as expected is $\mathbf{I}_N$, the N-dimensional identity matrix. Using the notation in [70], the vector of the representation (called component vector) with respect to the standard basis is $\mathbf{s}_E = \mathbf{s}$.

Because the signal[3] s is an element of the vector space $\mathbf{C}^N$, it can be represented as a linear combination of the vectors of any other ordered basis $B = \{b_0, \ldots, b_{N-1}\}$. These basis' vectors are themselves time signals. To determine $\mathbf{s}_B$ that represents the signal s in the new basis B, let the component vectors representing the vectors of the basis B in the standard basis E be $\mathbf{b}_n$, $n = 0, \ldots, N-1$, and let

$$\mathbf{B} = \left[\mathbf{b}_0 \cdots \mathbf{b}_{N-1}\right].$$

The component vector $\mathbf{b}_n$ collects the (ordered) samples of the signal b_n. Then, the signal s is represented in the basis B as

$$\mathbf{s} = \mathbf{B}\,\mathbf{s}_B. \tag{8.1}$$

[2] Representing a basis vector e_n by a vector $\mathbf{e}_{n+1}$ is definitely confusing notation. It will become clear why when we consider, for example, the z-transform of signals.

[3] We emphasize that a signal s is an ordered set of (complex valued) numbers or (complex valued) N-tuple. The vector representation $\mathbf{s}$ of s assumes a reference basis in $\mathbf{C}^N$.

On the left of Eq. (8.1), we have the component vector $\mathbf{s}$ of the signal s with respect to the standard basis in $\mathbf{C}^N$; on the right, we have the component vector $\mathbf{s}_B$ of the same signal s, now with respect to the basis B. Then,

$$\mathbf{s}_B = \mathbf{B}^{-1}\,\mathbf{s}. \tag{8.2}$$

For example, if $B = \left\{\exp\left(j\frac{2\pi}{N}kn\right), k = 0, \ldots, N-1\right\}_{0\leq n\leq N-1}$ is the Fourier basis of N complex exponentials, then $\mathbf{s}_B$ is the vector of Fourier coefficients of the signal s and Eqs. (8.1) and (8.2) are the Fourier synthesis and Fourier analysis expressions of s (inverse Fourier transform and Fourier transform of the signal).

Filters process signals. The simplest filter is the *delay*

$$s_{n-1} = z^{-1}s_n.$$

Consider a signal s_{in} and its delayed version s_{out}. Let their component vectors be $\mathbf{s}_{\text{in}}$ and $\mathbf{s}_{\text{out}}$, respectively. Because delay is a linear operation, we have

$$\mathbf{s}_{\text{out}} = \mathbf{A}\mathbf{s}_{\text{in}}. \tag{8.3}$$

The matrix $\mathbf{A}$ representing the periodic shift is

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix}. \tag{8.4}$$

This matrix is the cyclic shift. By replacing it in Eq. (8.3),

$$\begin{aligned} \mathbf{s}_{\text{out}} &= \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \mathbf{s}_{\text{in}} \\ &= \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{N-1} \end{bmatrix} \\ &= \begin{bmatrix} s_{N-1} \\ s_0 \\ \vdots \\ s_{N-2} \end{bmatrix}. \end{aligned}$$

A generic filter h is given by its z-transform

$$h(z) = h_0 z^0 + h_1 z^{-1} + \cdots + h_{N-1} z^{-(N-1)}.$$

In vector notation, and with respect to the standard basis E, the filter is represented by the matrix $\mathbf{H}$, a polynomial in the cyclic shift

$$\begin{aligned}\mathbf{H} &= h(\mathbf{A}) \\ &= h_0\mathbf{A}^0 + h_1\mathbf{A}^1 + \cdots + h_{N-1}\mathbf{A}^{(N-1)} \\ &= h_0\mathbf{I} + h_1\mathbf{A} + \cdots + h_{N-1}\mathbf{A}^{(N-1)} \\ &= \begin{bmatrix} h_0 & h_{N-1} & \cdots & h_2 & h_1 \\ h_1 & h_0 & h_{N-1} & & h_2 \\ \vdots & h_1 & h_0 & \ddots & \vdots \\ h_{N-2} & & \ddots & \ddots & h_{N-1} \\ h_{N-1} & h_{N-2} & \cdots & h_1 & h_0 \end{bmatrix}.\end{aligned}$$

This is a circulant matrix. It is of course specified by the filter coefficients that can also be read out from the first row of the filter $\mathbf{H}$.

Filters are *shift invariant*. This can be seen from the z-transform representation

$$z \cdot h(z) = h(z) \cdot z$$

or from the matrix representation

$$\mathbf{A} \cdot h(\mathbf{A}) = h(\mathbf{A}) \cdot \mathbf{A}.$$

Either of these equations states that filtering first by $h(z)$ (or $h(\mathbf{A})$) a signal and then delaying the filtered output leads to the same signal as delaying first the signal by z (or $\mathbf{A}$) and then filtering by $h(z)$ (or $h(\mathbf{A})$) the delayed signal. Finally, we observe that, from the Cayley-Hamilton Theorem [71,72], $\mathbf{A}$ satisfies its characteristic polynomial $\Delta(\mathbf{A})$, where $\Delta(\lambda)$ is the determinant of $\lambda\mathbf{I}-\mathbf{A}$. The characteristic polynomial $\Delta(\mathbf{A})$ has degree N, so, in DSP, as described so far, linear filters are (matrix) polynomials with degree at most $N-1$.

We consider the *Fourier Transform*. Without presenting details, we observe that the shift and filters are circulant matrices. As such, they are diagonalizable and diagonalized by the (inverse) of the discrete Fourier transform matrix (DFT). Because the DFT matrix is unitary and symmetric, its inverse is simply its conjugate DFT*. In other words,

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & \cdots & 1 \\ 1 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & \cdots & 1 & 0 \end{bmatrix} = \mathrm{DFT}^* \cdot \Lambda \cdot \mathrm{DFT} \tag{8.5}$$

$$= \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{j\frac{2\pi}{N}} & e^{j\frac{2\pi}{N}2} & \cdots & e^{j\frac{2\pi}{N}(N-1)} \\ 1 & e^{j\frac{2\pi}{N}2} & e^{j\frac{2\pi}{N}4} & \cdots & e^{j\frac{2\pi}{N}2(N-1)} \\ \vdots & & \vdots & & \vdots \\ 1 & e^{j\frac{2\pi}{N}(N-1)} & e^{j\frac{2\pi}{N}(N-1)2} & \cdots & e^{j\frac{2\pi}{N}(N-1)(N-1)} \end{bmatrix} \tag{8.6}$$

$$\cdot \begin{bmatrix} 1 & & & \\ & e^{-j\frac{2\pi}{N}} & & \\ & & \ddots & \\ & & & e^{-j\frac{2\pi}{N}(N-1)} \end{bmatrix} \tag{8.7}$$

$$\cdot \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{-j\frac{2\pi}{N}} & e^{-j\frac{2\pi}{N}2} & \cdots & e^{-j\frac{2\pi}{N}(N-1)} \\ 1 & e^{-j\frac{2\pi}{N}2} & e^{-j\frac{2\pi}{N}4} & \cdots & e^{-j\frac{2\pi}{N}2(N-1)} \\ \vdots & & \vdots & & \vdots \\ 1 & e^{-j\frac{2\pi}{N}(N-1)} & e^{-j\frac{2\pi}{N}(N-1)2} & \cdots & e^{-j\frac{2\pi}{N}(N-1)(N-1)} \end{bmatrix}, \tag{8.8}$$

where $(\cdot)^*$ stands for complex conjugate. We remark:

1. **Spectral components**: The columns of the inverse DFT* of the DFT matrix

$$\mathbf{v}_n = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 \\ e^{j\frac{2\pi}{N}n} \\ e^{j\frac{2\pi}{N}2n} \\ \vdots \\ e^{j\frac{2\pi}{N}(N-1)n} \end{bmatrix}, \quad n = 0, \ldots, N-1$$

are the N eigenvectors of $\mathbf{A}$ and form a complete orthonormal basis for $\mathbf{C}^N$. These vectors are the spectral components.

2. **Frequencies**: The diagonal entries of Λ are the eigenvalues of the time shift $\mathbf{A}$. In Physics and in operator theory, these eigenvalues are the frequencies of the signal. In DSP it is more common to call frequencies

$$\begin{aligned} \Omega_n &= -\frac{1}{2\pi j} \ln \lambda_n \\ &= -\frac{1}{2\pi j} \ln e^{-j\frac{2\pi}{N}n} \\ &= \frac{1}{N} n, \quad n = 0, \ldots, N-1. \end{aligned}$$

The N (time) frequencies Ω_n are all distinct, positive, equally spaced, and increasing from 0 to $\frac{N-1}{N}$. The spectral components are the complex exponential sinusoidal functions. For example, corresponding to the zero frequency is the DC spectral component (a vector whose entries are constant and all equal to $\frac{1}{\sqrt{N}}$).

This completes our brief review of DSP. It recasts well known DSP facts in a vector formulation that is appropriate for immediate generalization to GSP, as we show in the next section.

8.4 GRAPH SIGNAL PROCESSING

As we observed in Section 8.3, DSP [65–69] studies time or image signals. GSP[4] [61,62,73] generalizes DSP to graph signals, i.e., signals whose samples are indexed by nodes $v \in V$ of a generic directed, undirected, or mixed graph $G = (V, E)$, rather than necessarily by time instants or pixels of an image as with DSP. We follow the presentation in [61,62]. GSP as introduced in these references and in this chapter recovers DSP when we apply GSP to time signals or images. This is satisfying because when we extend a theory such as DSP to a broader context such as GSP, the new theory should revert to the original theory when applied to the original problem. In other words, GSP as described here if applied to time signals or images should recover the DSP presented in Section 8.3. This is not the case with other versions of GSP.

The main points in Section 8.3 when presenting DSP can be summarized as:

1. **Set of numbers**: Start with a set of N (complex valued) unordered numbers.
2. **Graph signal**: Order the numbers to get a signal (ordered tuple).
3. **Shift**: Define a shift.
4. **Shift invariance**: Assume filters are shift invariant.
5. **Graph spectral analysis**: Diagonalize the shift.

8.4.1 GRAPH SIGNALS

We first discuss graph signals. We start with an unordered set of complex valued numbers, possibly repeated, $\mathcal{S} = \left\{s_{\alpha_0}, \ldots, s_{\alpha_{N-1}}\right\} \subset \mathbb{C}$. We assume that a graph $G = (V, E)$ is given. In G, $V = \{0, \ldots, N-1\}$ is the set of N nodes and E is the set of edges. We further assume that each of the N data in $\mathcal{S}$ is assigned to a node of the graph G. The data in $\mathcal{S}$ is now ordered by the assumed ordering of the nodes of the graph and is given by an N-tuple $s = (s_0, \ldots, s_n, \ldots, s_{N-1})$. We can think of $\mathcal{S}$ as a scrambled version of the samples of the graph signal s. The component s_n of the N-tuple graph signal s, the nth sample of graph signal s, is now indexed by node $n \in V$ of the graph G.

The graph G can be arbitrary, i.e., it may be directed, undirected, or mixed (having both directed and undirected edges). For example, a directed edge from node i to node j captures dependency of the data s_j in node j, or sample s_j, on the data s_i in node i. This dependency will be made clear in Section 8.4.2.

As an example, we consider a time signal and its associated graph. Assume the time signal $s = (s_0, \ldots, s_n, \ldots, s_{N-1})$ where, see Section 8.3, the samples are indexed by time. We construct a graph, see Fig. 8.3, where the N nodes are the time instants 0 through $N-1$ and the label of the nodes are the numerical values of the time samples s_0 through s_{N-1}. Because sample s_{n+1} at time $n+1$ follows the sample s_n at time n, node n is connected to node $n+1$ by a directed edge. Assuming a cyclic or periodic

[4] Like for DSP, we consider here only linear graph signal processing and finite graph signals, i.e., graphs with a finite number N of nodes.

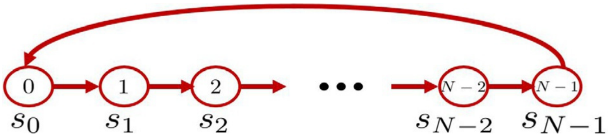

FIG. 8.3

Time cyclic graph.

signal extension $s_n = s_{n \mod N}$, see remark at the beginning of Section 8.3, node $N-1$ is connected back to node 0. This is a directed cyclic graph and it is associated with time signals.

One way to define a graph is through its adjacency matrix $\mathbf{A}$ [74]. The rows and columns of the adjacency matrix are labeled by its nodes from 0 to $N-1$. Entry $A_{ij} \neq 0$ of $\mathbf{A}$ is a weighted in-edge to node i from node j (or equivalently a weighted out-edge from node j to node i). Node j is an in-neighbor of node i, and node i is an out-neighbor of node j. The set of in-neighbors of i is its in-neighborhood $\mathcal{N}_i$, and similarly for its out-neighborhood. If $A_{ij} = A_{ji} \neq 0$, the edge (i,j) is undirected. If $A_{ij} = 0$, there is no in-edge to node i from node j. The graph may have cycles. Because A_{ij} can be arbitrarily valued, the graph adjacency matrix is a weighted matrix. Returning to time signals and the associated graph in Fig. 8.3, we note that the adjacency matrix of the time cyclic graph is the cyclic shift matrix in Eq. (8.4).

Because the graph signal assigns to each node $n \in V$ one of the complex numbers in $\mathcal{S}$, a graph signal s is a map

$$s : V \to \mathbb{C}.$$

As mentioned in Section 8.3, we work with vectors rather than tuples. With respect to the standard basis in $\mathbb{C}^N$, the graph signal is given by the component vector

$$\mathbf{s} = \begin{bmatrix} s_0 \\ \vdots \\ s_{N-1} \end{bmatrix} \in \mathbb{C}^N.$$

As an example, the impulse graph signal centered at node n is given by

$$\delta_n = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix},$$

which is zero everywhere, except at node n where it is one.

8.4.2 GRAPH SHIFT

The most elementary filtering operation in DSP is the delay given by the time shift z^{-1}. We saw in Section 8.3 that the matrix representation $\mathbf{A}$ of the time shift z^{-1} is given by the cyclic shift matrix given in Eq. (8.4). But, as we observed after Fig. 8.3, this matrix is the adjacency matrix of the time graph illustrated in Fig. 8.3. In other words, the shift matrix $\mathbf{A}$ representing the time shift z^{-1} is the adjacency matrix of the graph associated with the time signal.

We adopt here (in reverse) this insight and propose as graph shift $\mathbf{A}$ for graph signals s indexed by nodes of an arbitrary graph $G = (V, E)$ the adjacency matrix of the graph. In other words, the shifted version of the sample s_n of the graph signal is given by a weighted sum of the samples of the graph signal at the in-neighbors of node n

$$\begin{aligned} s_n &= \sum_{i=0}^{N-1} A_{ni} s_i \\ &= \sum_{i \in \mathcal{N}_n} A_{ni} s_i, \end{aligned}$$

where the weights A_{ni} are the entries of the adjacency matrix $\mathbf{A}$ and $\mathcal{N}_n$ is the in-neighborhood of node n. Collecting all the data in the signal vector $\mathbf{s}$, we get that the output $\mathbf{s}_{\text{out}}$ of the graph shift $\mathbf{A}$ to the input $\mathbf{s}_{\text{in}} = \mathbf{s}$ is given by

$$\mathbf{s}_{\text{out}} = \mathbf{A}\mathbf{s}_{\text{in}}, \tag{8.9}$$

where, we emphasize again, the graph shift is the adjacency $\mathbf{A}$ of the graph.

Remark. The samples of the graph signal are indexed by the nodes of the graph. Relabeling the nodes changes the graph signal component vector and the shift matrix. Let π be a permutation of the nodes of the graph. Then, for example, Eq. (8.9) becomes

$$\pi\, \mathbf{s}_{\text{out}} = \pi\, \mathbf{A}\pi^T \pi \mathbf{s}_{\text{in}},$$

where recall $\pi^{-1} = \pi^T$. Relabeling the nodes, correspondingly permutes the entries of the graph signal and the shift matrix is conjugated by the same permutation π, i.e., the rows of $\mathbf{A}$ are reordered by π and the columns reordered by the inverse of π. ■

8.4.3 GRAPH FILTERS AND GRAPH CONVOLUTION

A graph filter h is represented by a matrix $\mathbf{H}$, and *graph filtering* or *graph filtering convolution* by h is matrix-vector multiplication

$$\mathbf{s}_{\text{out}} = \mathbf{H} \cdot \mathbf{s}_{\text{in}}.$$

Shift Invariance. In DSP, shift invariant filters play an important role. In this chapter, we restrict attention to shift invariant filters. Shift invariance means that we can commute the operations of shifting and filtering. In other words, the filter is shift invariant if first shifting the input signal s_{in} and then filtering the shifted input by the graph filter h or first filtering s_{in} by h and then shifting the output signal leads to the same graph signal. In matrix-vector product form

$$\underbrace{\mathbf{A}\cdot\underbrace{(\mathbf{H}\cdot\mathbf{s}_{\text{in}})}_{\text{filter input}}}_{\text{shift filtered input}} = \underbrace{\mathbf{H}\cdot\underbrace{(\mathbf{A}\cdot\mathbf{s}_{\text{in}})}_{\text{shift input}}}_{\text{filter shifted input}} .$$

Because this is true for every graph signal, we conclude that for shift invariant filters

$$\mathbf{A}\cdot\mathbf{H} = \mathbf{H}\cdot\mathbf{A}, \tag{8.10}$$

and the graph filter h and the graph shift commute.

As explained in [58,61,62,72], if $\Delta(\mathbf{A}) = m(\mathbf{A})$, where $\Delta(\mathbf{A})$ and $m(\mathbf{A})$ are the characteristic polynomial and the minimal polynomial of the shift $\mathbf{A}$, then the shift invariant filters h are polynomials of the shift. In other words, if Eq. (8.10) holds for graph filter $\mathbf{H}$, then

$$\mathbf{H} = h(\mathbf{A}),$$

where $h(\cdot)$ is, by the Cayley-Hamilton Theorem [75], a polynomial of degree at most $N-1$. Also, if the shift is diagonalizable, shift invariant filters and the shift share the same (complete set of orthonormal) eigenvectors. This means that if $\mathbf{v}_n$ is a spectral component (eigenvector of $\mathbf{A}$) with corresponding eigenvalue λ_n

$$\mathbf{H}\cdot\mathbf{v}_n = h(\lambda_n)\,\mathbf{v}_n,$$

i.e., although graph filtering by $\mathbf{H}$ is in general a matrix-vector multiplication, graph filtering of a spectral component reduces to scalar multiplication by the polynomial filter evaluated at the eigenvalue λ_n. In words, the spectral components are *invariant* to graph filters.

8.4.4 GRAPH FOURIER TRANSFORM AND GRAPH SPECTRAL DECOMPOSITION

We now consider the graph Fourier transform (GFT). For simplicity, we focus on shifts $\mathbf{A}$ with N distinct eigenvalues λ_n, $n = 0, \ldots, N-1$. These shifts $\mathbf{A}$ are diagonalizable and their characteristic and minimal polynomials are equal, $\Delta(\mathbf{A}) = m(\mathbf{A})$. As argued in Section 8.4.3, shift invariant filters are then polynomials of the shift. The general case of $k < N$ distinct eigenvalues and nondiagonalizable shifts is considered in [61–63] and fully treated in [76]; we refer the reader to these references for full details.

In an analogy with classical DSP and Section 8.3, see Eq. (8.5), the inverse of the graph Fourier transform is the matrix of eigenvectors of the shift

$$\mathbf{A} = \underbrace{\left[\mathbf{v}_1 \cdots \mathbf{v}_{N-1}\right]}_{\mathbf{GFT}^{-1}} \operatorname{diag}\left[\lambda_1, \ldots, \lambda_{N-1}\right] \underbrace{\left[\mathbf{v}_1 \cdots \mathbf{v}_{N-1}\right]^{-1}}_{\mathbf{GFT}} \tag{8.11}$$

$$= \mathbf{GFT}^{-1}\Lambda\,\mathbf{GFT} \tag{8.12}$$

$$= \mathbf{GFT}^{H}\Lambda\,\mathbf{GFT}, \tag{8.13}$$

where $\mathbf{GFT}^{-1}$ is the matrix of eigenvectors of $\mathbf{A}$ and diag is a diagonal matrix. Because $\mathbf{A}$ is diagonalizable, it has a complete eigenbasis. So the graph Fourier transform $\mathbf{GFT}$ is unitary and $\mathbf{GFT}^{-1} = \mathbf{GFT}^H$.

In the sequel, we also use the following representation for the GFT.

$$\begin{aligned}\mathbf{GFT} = \mathbf{W}^H &= \begin{bmatrix} \mathbf{w}_1^H \\ \vdots \\ \mathbf{w}_{N-1}^H \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{v}_1^H \\ \vdots \\ \mathbf{v}_{N-1}^H \end{bmatrix},\end{aligned}$$

because the $\mathbf{GFT}$ is unitary.

The N distinct eigenvalues $\{\lambda_n\}_{0\le n\le N-1}$ of $\mathbf{A}$ are the graph frequencies and the eigenvectors $\{\mathbf{v}_n\}_{0\le n\le N-1}$ of $\mathbf{A}$ that are the columns of $\mathbf{GFT}^H$ are the graph spectral components.

Given a signal s, with vector component $\mathbf{s}$, its graph Fourier transform is then

$$\begin{aligned}\hat{\mathbf{s}} &= \mathbf{GFT} \cdot \mathbf{s} \\ &= \begin{bmatrix} \mathbf{v}_1^H \cdot \mathbf{s} \\ \vdots \\ \mathbf{v}_{N-1}^H \cdot \mathbf{s} \end{bmatrix}.\end{aligned}$$

This is the analysis formula of the graph Fourier transform. The synthesis of the original signal from its graph Fourier decomposition is

$$\begin{aligned}\mathbf{s} &= \mathbf{GFT}^H \cdot \hat{\mathbf{s}} \\ &= \sum_{n=0}^{N-1} \hat{s}_n \mathbf{v}_n \\ &= \sum_{n=0}^{N-1} \left(\mathbf{v}_n^H \cdot \mathbf{s}\right) \mathbf{v}_n \\ &= \sum_{n=0}^{N-1} \langle \mathbf{v}_n^H, \mathbf{s}\rangle\, \mathbf{v}_n,\end{aligned} \tag{8.14}$$

where $\mathbf{v}_n$ are the spectral components. In Eq. (8.14), $\langle\cdot,\cdot\rangle$ is the dot or scalar product. This is the graph signal equivalent expansion of the time domain Fourier series representation of the signal.

Remark. The vectors $\{\mathbf{w}_n\}_{0\le n\le N-1}$ are the left graph eigenvectors of the graph shift $\mathbf{A}$. Because the graph shift $\mathbf{A}$ is assumed to be diagonalizable, the left and right eigenvectors $\mathbf{w}_n$ and $\mathbf{v}_n$ can be chosen to be the same. ■

Graph Filter Frequency Response. Given a shift invariant filter h, its graph frequency response is

$$\left(h\left(\lambda_0\right), \ldots, h\left(\lambda_{N-1}\right)\right),$$

the polynomial filter h evaluated at the graph frequencies $\lambda_n, n = 0, \ldots, N-1$.

Graph Convolution Theorem. Following [61], see Equation (27) therein, we have the following result for filtering with shift invariant filters

$$\begin{aligned}
\mathbf{s}_{\text{out}} &= \mathbf{H} \cdot \mathbf{s}_{\text{in}} \\
&= h(\mathbf{A}) \cdot \mathbf{s}_{\text{in}} \\
&= h\left(\mathbf{GFT}^H \Lambda\, \mathbf{GFT}\right) \cdot \mathbf{s}_{\text{in}} \\
&= \underbrace{\mathbf{GFT}^H \cdot \underbrace{\left[h(\Lambda) \cdot \underbrace{\mathbf{GFT} \cdot \mathbf{s}_{\text{in}}}_{\text{GFT of } \mathbf{s}} \right]}_{\text{Filtering in spectral domain}}}_{\text{Inverse GFT}} .
\end{aligned} \tag{8.15}$$

This is the graph signal processing version of the Convolution Theorem of classical DSP. Filtering can be accomplished in the node domain as a matrix-vector multiplication. In alternative, as per Eq. (8.15), (1) we first GFT the input graph signal by $\hat{\mathbf{s}}_{\text{in}} = \mathbf{GFT} \cdot \mathbf{s}_{\text{in}}$, then (2) graph filter in the spectral domain as a pointwise multiplication of $\hat{\mathbf{s}}_{\text{in}}$, the GFT of the graph signal $\mathbf{s}_{\text{in}}$, with the graph frequency response $h(\Lambda)$ of the graph filter h, followed, finally, by (3) the inverse GFT of the graph filtered output in the graph spectral domain as $\mathbf{GFT}^H \cdot [h(\Lambda) \cdot \mathbf{GFT} \cdot \mathbf{s}_{\text{in}}]$.

Graph Frequency Ordering. To define the concepts of low, band, and high pass signals and filters, we need to order the graph frequencies. Because the underlying graph most likely will contain direct edges, the graph shift $\mathbf{A}$ is in general nonsymmetric and hence the graph frequencies may be complex valued. Because the complex numbers $\mathbb{C}$ is not an ordered field, there remains then the question of how to order the graph frequencies. In [63], we chose to order the graph frequencies through the total variation of the spectral components.

For time signals, the total variation of a signal $\mathbf{s}$ is

$$\text{TV}(\mathbf{s}) = \sum_{n=0}^{N-1} \left| s_n - s_{n-1 \mod N} \right| .$$

The total variation TV($\mathbf{s}$) of the time signal $\mathbf{s}$ measures how the signal varies over time. In [63], see equation below Fig. 3, we compute the TV for the time spectral components as

$$\text{TV}\left(\mathbf{v}_n\right) = \left|1 - \cos\frac{2\pi n}{N}\right| + \left|\sin\frac{2\pi n}{N}\right| .$$

This expression gives that $\text{TV}\left(\mathbf{v}_n\right) = \text{TV}\left(\mathbf{v}_{N-n}\right)$. The total variation for the DC-spectral component of time signals is zero. This agrees with our intuition—because the DC-spectral component of a time signals is a constant signal, its total variation across time is zero. This corresponds to the lowest time frequency $\Omega_0 = 0$. We can also compute from the above expression that the $\text{TV}\left(\mathbf{v}_n\right)$ is strictly increasing from $n = 0$ to $n = \frac{N}{2}$, if N is even, or $\frac{N-1}{2}$ if N is odd. So, with time signals, the frequencies can be ordered by ordering the associated spectral components according to their TV.

We use this method of ordering time frequencies to order the graph frequencies. From [63], the graph total variation of spectral component $\mathbf{v}$ is

$$\mathrm{TV}_{\mathrm{G}}(\mathbf{v}) = \left\| \mathbf{v} - \mathbf{A}^{\mathrm{norm}}\mathbf{v} \right\|_1 ,$$

where the normalized graph shift is

$$\mathbf{A}^{\mathrm{norm}} = \frac{1}{|\lambda_{\max}|}\mathbf{A},$$

and $\lambda_{\max}$ is the eigenvalue of $\mathbf{A}$ with largest magnitude, i.e., $|\lambda_{\max}| \geq |\lambda_n|, \forall n$. This normalization is used to guarantee that the shifted signal is properly scaled for comparison with the original nonshifted graph signal. It follows that, for spectral component $\mathbf{v}$,

$$\mathrm{TV}_{\mathrm{G}}(\mathbf{v}) = \left| 1 - \frac{\lambda}{|\lambda_{\max}|} \right| \|\mathbf{v}\|_1 ,$$

where $\|\cdot\|_1$ is the vector ℓ_1 norm. From here, it follows Theorem 2 in [63] that states that, given two distinct complex eigenvalues or graph frequencies λ_m and λ_n of the graph shift $\mathbf{A}$ with eigenvectors or spectral components $\mathbf{v}_m$ and $\mathbf{v}_n$, respectively, the total variation of these spectral components satisfies

$$\mathrm{TV}_G(\mathbf{v}_m) < \mathrm{TV}_G(\mathbf{v}_n)$$

if the graph frequency λ_m is located closer to the value $|\lambda_{\max}|$ on the complex plane than the graph frequency λ_n. Using these results we can order the graph frequencies and from this ordering define low, band, and high pass graph signals and graph filters; see [63] where we illustrate these concepts with the dataset of hyperlinked political blogs [25].

8.4.5 FURTHER TOPICS AND APPLICATIONS

There is significant work that extends GSP to many other areas in signal and data processing. Sampling is considered in, for example [55,77–80], and interpolation in [73]. Uncertainty principles are in [81–83]. Reference [84] considers approximation of signals supported on graphs. References [85,86] extend classical multirate signal processing theory to graphs. In [87–89], the authors consider interpolation of graph signals, spectral estimation of graph processes, and blind identification of graph filters. A recent topic of interest is graph learning from data; see a sample of work related to this topic in [90–94].

There is an increasing set of applications of GSP. In our original work [61,62,73], we illustrated the application of GSP in filter design for denoising, for classification, for lossy compression of graph signals, and for analysis of graph signals, for detection of malfunctioning in sensor networks. Reference [95] applies GSP to image inpainting and other problems of graph signal recovery by variation minimization. In [96], GSP is applied to semisupervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring and [97] applies GSP to fast resampling of three-dimensional point clouds via graphs. In [98], multiresolution representations for piecewise-smooth signals on graphs are developed. In recent work [99], we apply GSP to convolutional neural networks (CNN), replacing the image or regular lattice-like convolution step in CNN by graph convolution, leading to noticeable increased performance over other existing approaches.

In this chapter, we studied GSP assuming the graph shift is the adjacency matrix of the graph. There is a healthy literature that develops GSP with other definitions for the graph shift. In [64], GSP is developed adopting the graph Laplacian as shift and several subsequent works have adopted this shift. In [100,101], variations on the adjacency matrix as shift are discussed with different trade-offs with respect to adopting the adjacency matrix as shift as done here and in our original work [61]. We note that, following the algebraic signal processing [57–60,102], there is no "optimal" or "best choice" of shift. Different shifts lead to different signal models, and the question is which of these is better suited and leads to better results in the context of a specific application.

A final note on the spectral decomposition of graph signals. In [76], we show that in many applications the shift has repeated eigenvalues or is not diagonalizable. The graph Fourier transform is then dependent on choices of basis associated with corresponding invariant or spectral subspaces. So, care should be taken to completely specify these choices before claiming results on graph Fourier decompositions. Also, the concepts of frequency and spectral components are then to be handled with care. Further, if the shift is not diagonalizable, we work with the Jordan form of the shift that is numerically sensitive and the spectral components are (oblique) projections on subspaces of dimension larger than one. We refer to [76] for further details.

8.5 CONCLUSION

This chapter considered GSP that extends classical DSP to data that is indexed by the nodes of a graph. We studied linear GSP with an underlying graph with a finite number of nodes. We introduced the main concepts of classic DSP but for graph signals, i.e., data indexed by the nodes of the graph G. The shift plays an important role in GSP. We worked here with the shift that is identified with the adjacency matrix of the graph. Shift invariant filters are polynomials of the shift. We discussed the graph Fourier transform that introduces the graph frequency, and, by a suitable ordering of the graph frequencies, we were able to define low, band, and high pass graph signals and graph filters.

REFERENCES

[1] Turner V. The digital universe of opportunities: rich data and the increasing value of the internet of things. tech. rep.. IDC; 2014. https://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm.

[2] Johnston L. A "Library of Congress" worth of data: it's all in how you define it; 2012. https://blogs.loc.gov/thesignal/2012/04/a-library-of-congress-worth-of-data-its-all-in-how-you-define-it/ [Recovered 17 December 2017].

[3] Reinsel D, Gantz J, Rydning J. Data age 2025: the evolution of data to life-critical don't focus on big data; focus on the data that's big. tech. rep.. IDC; 2017. An IDC White Paper, Sponsored by Seagate. https://www.seagate.com/files/www-content/our-story/trends/files/Seagate-WP-DataAge2025-March-2017.pdf.

[4] Weissberger A. IDC directions 2016: IOT (internet of things) outlook vs current market assessment. IEEE Communications Society, ComSoc blog; 2016. http://techblog.comsoc.org/2016/03/09/idc-directions-2016-iot-internet-of-things-outlook-vs-current-market-assessment/.

[5] Börner K, Sanyal S, Vespignani A. Network science. Ann Rev Inf Sci Technol 2007;41(1):537–607.

[6] Newman M. Networks: an introduction. Oxford University Press; 2010.

[7] Jackson MO. Social and economic networks. Princeton University Press; 2010.

[8] Easley D, Kleinberg J. Networks, crowds, and markets: reasoning about a highly connected world. Cambridge University Press; 2010.

[9] National Research Council. Network science. Washington, DC: The National Academies Press; 2005. ISBN 978-0-309-10026-7. https://doi.org/10.17226/11516. https://www.nap.edu/catalog/11516/network-science.

[10] Padgett JF, Ansell CK. Robust action and the rise of the Medici, 1400-1434. Am J Sociol 1993;98(6):1259–319.

[11] Bbuuggzz. 15th century Florentine marriages data from Padgett and Ansell.pdf. File licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license; 2013. https://commons.wikimedia.org/wiki/File:15th_Century_Florentine_Marriges_Data_from_Padgett_and_Ansell.pdf [Retrieved 8 December 2017].

[12] Freeman LC. Centrality in social networks conceptual clarification. Soc Networks 1978;1(3):215–39.

[13] Barabási AL. Network science. Cambridge University Press; 2016.

[14] Barrat A, Barthelemy M, Vespignani A. Dynamical processes on complex networks. Cambridge University Press; 2008.

[15] Liggett TM. Interacting particle systems, vol. 276. Springer Science & Business Media; 2012.

[16] Kipnis C, Landim C. Scaling limits of interacting particle systems, vol. 320. Springer Science & Business Media; 2013.

[17] Pastor-Satorras R, Castellano C, Van Mieghem P, Vespignani A. Epidemic processes in complex networks. Rev Mod Phys 2015;87(3):925.

[18] Lanchier N. Interacting particle systems. In: Stochastic modeling. Springer; 2017. p. 235–44.

[19] Zhang J, Moura JMF. Diffusion in social networks as SIS epidemics: beyond full mixing and complete graphs. IEEE J Sel Top Signal Process 2014;8(4):537–51.

[20] Zhang J, Moura JMF. Role of subgraphs in epidemics over finite-size networks under the scaled SIS process. J Complex Networks 2015;3(4):584–605.

[21] Kelly FP. Reversibility and stochastic networks. Cambridge University Press; 2011.

[22] Santos A, Moura JMF, Xavier JM. Bi-virus SIS epidemics over networks: qualitative analysis. IEEE Trans Netw Sci Eng 2015;2(1):17–29.

[23] Santos AA, Kar S, Moura JMF, Xavier J. Thermodynamic limit of interacting particle systems over dynamical networks. In: 2016 50th Asilomar conference on signals, systems and computers. IEEE; 2016. p. 997–1000.

[24] Santos A, Moura JMF, Xavier JM. Sufficient condition for survival of the fittest in a bi-virus epidemics. In: 2015 49th Asilomar conference on signals, systems and computers. IEEE; 2015. p. 1323–7.

[25] Adamic LA, Glance N. The political blogosphere and the 2004 U.S. election: divided they blog. In: Proceedings of the 3rd international workshop on link discovery. ACM; 2005. p. 36–43.

[26] Deri JA, Moura JMF. Churn detection in large user networks. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2014. p. 1090–4.

[27] Kindermann R, Snell JL. Markov random fields and their applications, vol. 1. American Mathematical Society; 1980.

[28] Rue H, Held L. Gaussian Markov random fields: theory and applications. CRC Press; 2005.

[29] Besag J. Spatial interaction and the statistical analysis of lattice systems. J R Stat Soc Ser B Methodol 1974:192–236.

[30] Chellappa R, Jain A, editors. Markov random fields. Theory and application. Boston, MA: Academic Press; 1993.

[31] Moura JMF, Balram N. Recursive structure of noncausal Gauss-Markov random fields. IEEE Trans Inf Theory 1992;38(2):334–54.

[32] Balram N, Moura JMF. Noncausal Gauss Markov random fields: parameter structure and estimation. IEEE Trans Inf Theory 1993;39(4):1333–55.

[33] Lauritzen SL. Graphical models, vol. 17. Clarendon Press; 1996.

[34] Wainwright MJ, Jordan MI. Graphical models, exponential families, and variational inference. In: Foundations and trends® in machine learning, vol. 1(1–2); 2008. p. 1–305.

[35] Koller D, Friedman N. Probabilistic graphical models: principles and techniques. MIT Press; 2009.

[36] Jordan M, Sudderth E, Wainwright M, Willsky A. Major advances and emerging developments of graphical models [from the guest editors]. IEEE Signal Process Mag 2010;27(6):17–138.

[37] Sudderth EB, Ihler AT, Isard M, Freeman WT, Willsky AS. Nonparametric belief propagation. Commun ACM 2010;53(10):95–103.

[38] Tenenbaum JF, Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. Science 2000;290:2319–23.

[39] Roweis S, Saul L. Nonlinear dimensionality reduction by locally linear embedding. Science 2000;290:2323–6.

[40] Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput 2003;15(6):1373–96.

[41] Donoho DL, Grimes C. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. Proc Natl Acad Sci U S A 2003;100(10):5591–6.

[42] Belkin M, Niyogi P. Using manifold structure for partially labeled classification. In: Advances in neural information processing systems (NIPS); 2002. p. 953–60.

[43] Belkin M, Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in neural information processing systems; 2002. p. 585–91.

[44] Ng AY, Jordan MI, Weiss Y. On spectral clustering: analysis and an algorithm. In: Advances in neural information processing systems; 2002. p. 849–56.

[45] Schaeffer SE. Graph clustering. Comput Sci Rev 2007;1(1):27–64.

[46] Coifman RR, Lafon S, Lee A, Maggioni M, Nadler B, Warner FJ, Zucker SW. Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. Proc Nat Acad Sci 2005;102(21):7426–31.

[47] Coifman RR, Lafon S, Lee A, Maggioni M, Nadler B, Warner FJ, Zucker SW. Geometric diffusions as a tool for harmonic analysis and structure definition of data: multiscale methods. Proc Natl Acad Sci U S A 2005;102(21):7432–7.

[48] Coifman RR, Maggioni M. Diffusion wavelets. Appl Comput Harmon Anal 2006;21(1):53–94.

[49] Ganesan D, Greenstein B, Estrin D, Heidemann J, Govindan R. Multiresolution storage and search in sensor networks. ACM Trans Storage 2005;1:277–315.

[50] Wagner R, Cohen A, Baraniuk RG, Du S, Johnson DB. An architecture for distributed wavelet analysis and processing in sensor networks. In: Information processing in sensor networks (IPSN); 2006. p. 243–50.

[51] Wagner R, Delouille V, Baraniuk RG. Distributed wavelet de-noising for sensor networks. In: Proc. IEEE conference on decision and control (CDC); 2006. p. 373–9.

[52] Haupt J, Bajwa WU, Rabbat M, Nowak R. Compressed sensing for networked data. IEEE Signal Process Mag 2008;25(2):92–101.

[53] Narang SK, Ortega A. Local two-channel critically sampled filter-banks on graphs. In: IEEE international conference on image processing (ICIP); 2010. p. 333–6.

[54] Hammond DK, Vandergheynst P, Gribonval R. Wavelets on graphs via spectral graph theory. J Appl Comput Harmon Anal 2011;30(2):129–50.

[55] Narang SK, Ortega A. Downsampling graphs using spectral theory. In: IEEE international conference on acoustics, speech and signal processing (ICASSP); 2011. p. 4208–11.

[56] Narang SK, Ortega A. Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Trans Signal Process 2012;60(6):2786–99.

[57] Püschel M, Moura JMF. The algebraic approach to the discrete cosine and sine transforms and their fast algorithms. SIAM J Comp 2003;32(5):1280–316.

[58] Püschel M, Moura JMF. Algebraic signal processing theory; 2006. 67 p. http://arxiv.org/abs/cs.IT/0612077.
[59] Püschel M, Moura JMF. Algebraic signal processing theory: foundation and 1-D time. IEEE Trans Signal Process 2008;56(8):3572–85.
[60] Püschel M, Moura JMF. Algebraic signal processing theory: 1-D space. IEEE Trans Signal Process 2008;56(8):3586–99.
[61] Sandryhaila A, Moura JMF. Discrete signal processing on graphs. IEEE Trans Signal Process 2013;61(7):1644–56.
[62] Sandryhaila A, Moura JMF. Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process Mag 2014;31(5):80–90.
[63] Sandryhaila A, Moura JMF. Discrete signal processing on graphs: frequency analysis. IEEE Trans Signal Process 2014;62(12):3042–54.
[64] Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P. The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag 2013;30:83–98.
[65] Oppenheim AV, Schafer RW. Digital signal processing. Englewood Cliffs, NJ: Prentice-Hall; 1975.
[66] Oppenheim AV, Willsky AS. Signals and systems. Englewood Cliffs, NJ: Prentice-Hall; 1983.
[67] Siebert WM. Circuits, signals, and systems. Cambridge, MA: The MIT Press; 1986.
[68] Oppenheim AV, Schafer RW. Discrete-time signal processing. Englewood Cliffs, NJ: Prentice-Hall; 1989.
[69] Mitra SK. Digital signal processing. A computer-based approach. New York: McGraw Hill; 1998.
[70] Bendersky E. Change of basis in linear algebra (posted on web); 2015. https://eli.thegreenplace.net/2015/change-of-basis-in-linear-algebra/ [Accessed 11 December 2017].
[71] Gantmacher FR. Matrix theory. New York: Chelsea; 1959. p. 21.
[72] Lancaster P, Tismenetsky M. The theory of matrices: with applications. Elsevier; 1985.
[73] Narang SK, Gadde A, Ortega A. Signal processing techniques for interpolation in graph structured data. In: 2013 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2013. p. 5445–9.
[74] Chung FRK. Spectral graph theory. American Mathematical Society; 1996.
[75] Gantmacher FR, Brenner JL. Applications of the theory of matrices. Courier Corporation; 2005.
[76] Deri JA, Moura JMF. Spectral projector-based graph Fourier transforms. IEEE J Sel Top Signal Process 2017;11(6):785–95. https://doi.org/10.1109/JSTSP.2017.2731599.
[77] Anis A, Gadde A, Ortega A. Towards a sampling theorem for signals on arbitrary graphs. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2014. p. 3864–8.
[78] Chen S, Varma R, Sandryhaila A, Kovačević J. Discrete signal processing on graphs: sampling theory. IEEE Trans Signal Process 2015;63(24):6510–23.
[79] Puy G, Tremblay N, Gribonval R, Vandergheynst P. Random sampling of bandlimited signals on graphs. Appl Comput Harmon Anal 2018;44(2):446–75.
[80] Anis A, Gadde A, Ortega A. Efficient sampling set selection for bandlimited graph signals using graph spectral proxies. IEEE Trans Signal Process 2016;64(14):3775–89.
[81] Agaskar A, Lu YM. Uncertainty principles for signals defined on graphs: bounds and characterizations. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2012. p. 3493–6.
[82] Agaskar A, Lu YM. A spectral graph uncertainty principle. IEEE Trans Inf Theory 2013;59(7):4338–56.
[83] Pasdeloup B, Alami R, Gripon V, Rabbat M. Toward an uncertainty principle for weighted graphs. In: 2015 23rd European signal processing conference (EUSIPCO). IEEE; 2015. p. 1496–500.
[84] Zhu X, Rabbat M. Approximating signals supported on graphs. In: IEEE international conference on acoustics, speech and signal processing (ICASSP); 2012. p. 3921–24.
[85] Teke O, Vaidyanathan P. Extending classical multirate signal processing theory to graphs—Part I: Fundamentals. IEEE Trans Signal Process 2017;65(2):409–22.

[86] Teke O, Vaidyanathan P. Extending classical multirate signal processing theory to graphs—Part II: M-channel filter banks. IEEE Trans Signal Process 2017;65(2):423–37.

[87] Segarra S, Marques AG, Leus G, Ribeiro A. Interpolation of graph signals using shift-invariant graph filters. In: 2015 23rd European signal processing conference (EUSIPCO). IEEE; 2015. p. 210–4.

[88] Marques AG, Segarra S, Leus G, Ribeiro A. Stationary graph processes and spectral estimation; 2016. arXiv preprint arXiv:160304667.

[89] Segarra S, Mateos G, Marques AG, Ribeiro A. Blind identification of graph filters. IEEE Trans Signal Process 2017;65(5):1146–59.

[90] Mei J, Moura JMF. Fitting graph models to big data. In: 2015 49th Asilomar conference on signals, systems and computers. IEEE; 2015. p. 387–90.

[91] Mei J, Moura JMF. Signal processing on graphs: estimating the structure of a graph. In: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2015. p. 5495–9.

[92] Mei J, Moura JMF. Signal processing on graphs: performance of graph structure estimation. In: 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2016. p. 6165–9.

[93] Mei J, Moura JMF. Signal processing on graphs: causal modeling of unstructured data. IEEE Trans Signal Process 2017;65(8):2077–92.

[94] Egilmez HE, Pavez E, Ortega A. Graph learning from data under Laplacian and structural constraints. IEEE J Sel Top Signal Process 2017;11(6):825–41.

[95] Chen S, Sandryhaila A, Moura JMF, Kovačević J. Signal recovery on graphs: variation minimization. IEEE Trans Signal Process 2015;63(17):4609–24.

[96] Chen S, Cerda F, Rizzo P, Bielak J, Garrett JH, Kovačević J. Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring. IEEE Trans Signal Process 2014;62(11):2879–93.

[97] Chen S, Tian D, Feng C, Vetro A, Kovačević J. Fast resampling of 3D point clouds via graphs. IEEE Trans Signal Process 2017;66(3):666–81.

[98] Chen S, Singh A, Kovačević J. Multiresolution representations for piecewise-smooth signals on graphs. https://arxiv.org/abs/1803.02944; March 2018.

[99] Du J, Zhang S, Wu G, Moura JMF, Kar S. Topology adaptive graph convolutional networks; 2017. ArXiv preprint arXiv:1710.10370.

[100] Girault B, Gonçalves P, Fleury E. Translation on graphs: an isometric shift operator. IEEE Signal Process Lett 2015;22(12):2416–20. https://doi.org/10.1109/LSP.2015.2488279.

[101] Gavili A, Zhang XP. On the shift operator, graph frequency, and optimal filtering in graph signal processing. IEEE Trans Signal Process 2017;65(23):6303–18. https://doi.org/10.1109/TSP.2017.2752689.

[102] Püschel M, Moura JMF. Algebraic signal processing theory: Cooley-Tukey type algorithms for DCTs and DSTs. IEEE Trans Signal Process 2008;56(4):1502–21.