Foundations and Trends® in Machine Learning

Data Analytics on Graphs Part II: Signals on Graphs

Suggested Citation: Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides (2020), "Data Analytics on Graphs Part II: Signals on Graphs", Foundations and Trends® in Machine Learning: Vol. 13, No. 2–3, pp 158–331. DOI: 10.1561/2200000078-2.

Ljubiša Stanković
University of Montenegro
Montenegro
ljubisa@ucg.ac.me

Danilo Mandic
Imperial College London
UK
d.mandic@imperial.ac.uk

Miloš Daković
University of Montenegro
Montenegro
milos@ucg.ac.me

Miloš Brajović
University of Montenegro
Montenegro
milosb@ucg.ac.me

Bruno Scalzo
Imperial College London
UK
bruno.scalzo-dees12@imperial.ac.uk

Shengxi Li
Imperial College London
UK
shengxi.li17@imperial.ac.uk

Anthony G. Constantinides
Imperial College London
UK
a.constantinides@imperial.ac.uk

now
the essence of knowledge
Boston — Delft

Contents

Data Analytics on Graphs Part II: Signals on Graphs

Ljubiša Stanković[1], Danilo Mandic[2], Miloš Daković[3], Miloš Brajović[4], Bruno Scalzo[5], Shengxi Li[6] and Anthony G. Constantinides[7]

[1] *University of Montenegro, Montenegro; ljubisa@ucg.ac.me*
[2] *Imperial College London, UK; d.mandic@imperial.ac.uk*
[3] *University of Montenegro, Montenegro; milos@ucg.ac.me*
[4] *University of Montenegro, Montenegro; milosb@ucg.ac.me*
[5] *Imperial College London, UK; bruno.scalzo-dees12@imperial.ac.uk*
[6] *Imperial College London, UK; shengxi.li17@imperial.ac.uk*
[7] *Imperial College London, UK; a.constantinides@imperial.ac.uk*

ABSTRACT

The area of Data Analytics on graphs deals with information processing of data acquired on irregular but structured graph domains. The focus of Part I of this monograph has been on both the fundamental and higher-order graph properties, graph topologies, and spectral representations of graphs. Part I also establishes rigorous frameworks for vertex clustering and graph segmentation, and illustrates the power of graphs in various data association tasks. Part II embarks on these concepts to address the algorithmic and practical issues related to data/signal processing on graphs, with the focus on the analysis and estimation of both deterministic and random data on graphs. The fundamental ideas related to graph signals are introduced through a simple and intuitive, yet general enough case study of multisensor temperature field estimation. The concept of systems on graph

Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides (2020), "Data Analytics on Graphs Part II: Signals on Graphs", Foundations and Trends® in Machine Learning: Vol. 13, No. 2–3, pp 158–331. DOI: 10.1561/2200000078-2.

is defined using graph signal shift operators, which generalize the corresponding principles from traditional learning systems. At the core of the spectral domain representation of graph signals and systems is the Graph Fourier Transform (GFT), defined based on the eigendecomposition of both the adjacency matrix and the graph Laplacian. Spectral domain representations are then used as the basis to introduce graph signal filtering concepts and address their design, including Chebyshev series polynomial approximation. Ideas related to the sampling of graph signals, and in particular the challenging topic of data dimensionality reduction through graph subsampling, are presented and further linked with compressive sensing. The principles of time-varying signals on graphs and basic definitions related to random graph signals are next reviewed. Localized graph signal analysis in the joint vertex-spectral domain is referred to as the vertex-frequency analysis, since it can be considered as an extension of classical time-frequency analysis to the graph serving as signal domain. Important aspects of the local graph Fourier transform (LGFT) are covered, together with its various forms including the graph spectral and vertex domain windows and the inversion conditions and relations. A link between the LGFT with a varying spectral window and the spectral graph wavelet transform (SGWT) is also established. Realizations of the LGFT and SGWT using polynomial (Chebyshev) approximations of the spectral functions are further considered and supported by examples. Finally, energy versions of the vertex-frequency representations are introduced, along with their relations with classical time-frequency analysis, including a vertex-frequency distribution that can satisfy the marginal properties. The material is supported by illustrative examples.

Keywords: graph theory; random data on graphs; big data on graphs; signal processing on graphs; machine learning on graphs; graph

topology learning; systems on graphs; vertex-frequency estimation; graph neural networks; graphs and tensors.

1

Introduction

Graphs are structures, often irregular, constructed in a way to represent the observed data and to account, in a natural way, the specific interrelationships between the data sources. However, traditional approaches have been established outside Machine Learning and Signal Processing, with which largely focus on analyzing the underlying graphs rather than dealing with signals on graphs. Moreover, given the rapidly increasing availability of multisensor and multinode measurements, likely recorded on irregular or ad-hoc grids, it would be extremely advantageous to analyze such structured data as "signals on graphs" and thus benefit from the ability of graphs to account for spatial sensing awareness, physical intuition and sensor importance, together with the inherent "local versus global" sensor association. The aim of Part II of this monograph is therefore to establish a common language between graph signals which are observed on irregular signal domains, and some of the fundamental paradigms in Learning Systems, Signal Processing and Data Analytics, such as spectral analysis, system transfer function, digital filter design, parameter estimation, and optimal denoising.

In classical Data Analytics and Signal Processing, the signal domain is determined by equidistant time instants or by a set of spatial sensing

points on a uniform grid. However, increasingly the actual data sensing domain may not even be related to the physical dimensions of time and/or space, and it typically does exhibit various forms of irregularity, as, for example, in social or web-related networks, where the sensing points and their connectivity pertain to specific objects/nodes and ad-hoc topology of their links. It should be noted that even for the data acquired on well defined time and space domains, the introduction of new relations between the signal samples, through graphs, may yield new insights into the analysis and provide enhanced data processing (for example, based on local similarity, through neighborhoods). We therefore set out to demonstrate that the advantage of graphs over classical data domains is that graphs account naturally and comprehensively for irregular data relations in the problem definition, together with the corresponding data connectivity in the analysis (Chen *et al.*, 2014; Ekambaram, 2014; Gavili and Zhang, 2017; Hamon *et al.*, 2016; Moura, 2018; Sandryhaila and Moura, 2013; Shuman *et al.*, 2013; Vetterli *et al.*, 2014).

To build up the intuition behind the fundamental ideas of signals/data on graphs, a simple yet general example of multisensor temperature estimation is first considered in Section 2. Basic concepts regarding the signals and systems on graphs are presented in Section 3, including basic definitions, operations and transforms, which generalize the foundations of traditional signal processing. Systems on graphs are interpreted starting from a comprehensive account of the existing and the introduction of a novel, isometric, graph signal shift operator. Further, graph Fourier transform is defined based on both the adjacency matrix and the graph Laplacian and it serves as the basis to introduce graph signal filtering concepts. Various ideas related to the sampling of graph signals, and particularly, the challenging topic of their subsampling, are reviewed in Section 4. Sections 6 and 7 present the concepts of time-varying signals on graphs and introduce basic definitions related to random graph signals. Localized graph signal behavior can be simultaneously characterized in the vertex-frequency domain, which is discussed in Section 8. This section also covers the important topics of local graph Fourier transform, various forms of its inversion, relations with the frames and links with the graph wavelet transform. Energy

versions of the vertex-frequency representations are also considered, along with their relations with classical time-frequency analysis.

2

Problem Statement: An Illustrative Example

Consider a multi-sensor setup for measuring a temperature field in a region of interest. The temperature sensing locations are chosen according to the significance of a particular geographic area to local users, with $N = 16$ sensing points in total, as shown in Figure 2.1(a). The temperature field is denoted by $\{x(n)\}$, with n as the sensor index, while a snapshot of its values is given in Figure 2.1(b). Each measured sensor signal can then be mathematically expressed as

$$x(n) = s(n) + \varepsilon(n), \quad n = 0, 1, \ldots, 15, \tag{2.1}$$

where $s(n)$ is the true temperature that would have been obtained in ideal measuring conditions and $\varepsilon(n)$ comprises the adverse effects of the local environment on sensor readings or faulty sensor activity, and is referred to as "noise" in the sequel. For illustrative purposes, in our study each $\varepsilon(n)$ was modeled as a realization of white, zero-mean, Gaussian process, with standard deviation $\sigma_\varepsilon = 2$, that is, $\varepsilon(n) \sim \mathcal{N}(0, 4)$. It was added to the signal, $s(n)$, to yield the signal-to-noise ratio in $x(n)$ of $\text{SNR}_{in} = 14.2$ dB.

Remark 1: Classical data analytics requires a rearrangement of the quintessentially irregular spatial temperature sensing arrangement in Figure 2.1(a) into a linear structure shown in Figure 2.1(b). Obviously,

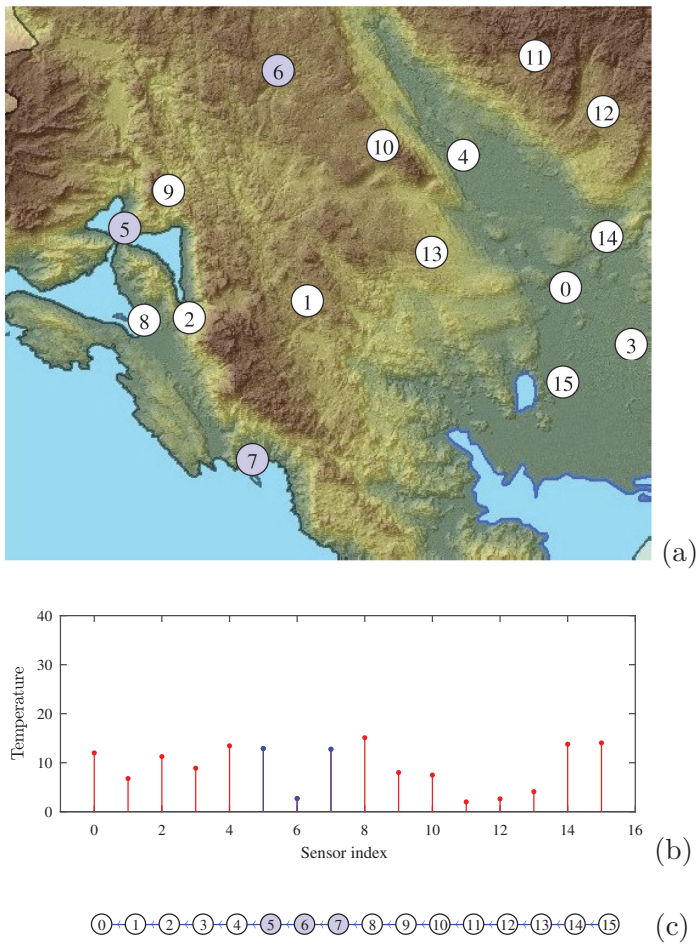


Figure 2.1: Temperature sensing as a classic data analytics problem. (a) Sensing locations in a geographic region along the Adriatic sea. (b) Temperatures measured at $N = 16$ sensing locations. In standard data estimation, the spatial sensor index is used for the horizontal axis and serves as the data domain. This domain can be interpreted as a directed path graph structure, shown in the bottom panel (c). Observe that the consecutive samples (vertices) on this path graph offer no physical intuition or interpretation, as in this "brute force" arrangement, for example, vertex 6 is located on a high mountain, whereas its neighboring vertices 5 and 7 are located along the sea; despite the consecutive index numbers these sensors are physically distant, as indicated by their very different temperature measurements.

such "lexicographic" ordering is not amenable to exploiting the information related to the actual sensor locations, which is inherently dictated by the terrain. This renders classical analyses of this multisensor temperature field inapplicable (or at best suboptimal), as the performance critically depends on the chosen sensor ordering scheme. This exemplifies that even a most routine multisensor measurement setup requires a more complex estimation structure than the standard linear one corresponding to the classical signal processing framework, shown in Figure 2.1(b).

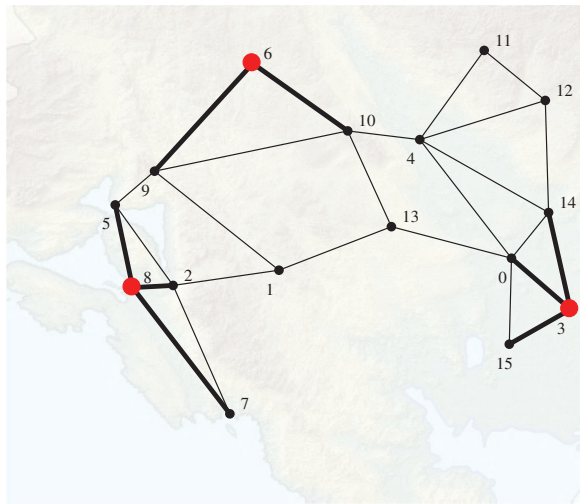
To introduce a "situation-aware" noise reduction scheme for the temperature field in Figure 2.1, we proceed to explore a graph-theoretic framework to this problem, starting from a local signal average operator. In classical analysis, this may be achieved through a moving average operator, e.g., by averaging across the neighboring data samples, or equivalently neighboring sensors in the linear data setup in Figure 2.1(b), and for each sensing point. Physically, such local neighborhood should include close neighboring sensing points but only those which also exhibit similar meteorological properties defined by the sensor distance, altitude difference, and other terrain specific properties. In other words, since the sensor network in Figure 2.1 measures a set of related temperatures from irregularly spaced sensors, an effective estimation strategy should include domain knowledge – not possible to achieve with standard methods (linear path graph).

To illustrate the advantages of approaches based on local information (neighborhood based), consider the neighborhoods for the sensing points $n = 3$ (low land), $n = 6$ (mountains), and 8 (coast), shown in Figure 2.2(a). The cumulative temperature for each sensing point is then given by

$$y(n) = \sum_{m \text{ at and around } n} x(m),$$

so that the local average temperature for a sensing point n may be obtained by dividing the cumulative temperature, $y(n)$, with the number of included sensing points (size of local neighborhood). For example, for the sensing points $n = 3$ and $n = 6$, presented in Figure 2.2(a), the

(a)



(b)

Figure 2.2: Temperature sensing setup as a graph signal estimation problem. (a) Local neighborhood for the sensing points $n = 3$, 6, and 8. These neighborhoods are chosen using "domain knowledge" dictated by the local terrain and by taking into account the sensor distance and altitude. Neighboring sensors for each of these sensing locations (vertices) are chosen in a physically meaningful way and their relation is indicated by the connectivity lines, that is, graph edges. (b) Local neighborhoods for all sensing vertices, presented in a graph form (thick lines indicate the edges from (a)).

"domain knowledge aware" local estimation takes the form

$$y(3) = x(3) + x(0) + x(14) + x(15) \tag{2.2}$$

$$y(6) = x(6) + x(9) + x(10). \tag{2.3}$$

For convenience, the full set of relations among the sensing points can now be arranged into a matrix form, to give

$$\mathbf{y} = \mathbf{x} + \mathbf{A}\mathbf{x}, \tag{2.4}$$

where the **adjacency matrix A**, given in (2.5), indicates the connectivity structure of the sensing locations; this local connectivity structure should be involved in the calculation of each $y(n)$.

This simple real-world example can be interpreted within the graph signal processing framework as follows:

- Sensing points where the signal is measured are designated as the **graph vertices**, as in Figure 2.1.

$$\mathbf{A} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \end{array} \\ \left[\begin{array}{cccccccccccccccc} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{2.5}$$

$$
\mathbf{W} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \end{array}
\begin{array}{c}
\begin{array}{cccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \end{array} \\
\left[\begin{array}{cccccccccccccccc}
0 & 0 & 0 & 0.97 & 0.91 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.05 & 0.90 & 0.94 \\
0 & 0 & 0.03 & 0 & 0 & 0 & 0 & 0 & 0 & 0.37 & 0 & 0 & 0.78 & 0 & 0 \\
0 & 0.03 & 0 & 0 & 0 & 0.96 & 0 & 0.95 & 0.98 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.97 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.88 & 0.96 \\
0.91 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.06 & 0.01 & 0.01 & 0.94 & 0 \\
0 & 0 & 0.96 & 0 & 0 & 0 & 0 & 0 & 0.97 & 0.01 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.62 & 0.40 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.95 & 0 & 0 & 0 & 0 & 0 & 0.94 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.98 & 0 & 0 & 0.97 & 0 & 0.94 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.37 & 0 & 0 & 0 & 0.01 & 0.62 & 0 & 0 & 0 & 0.25 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.06 & 0 & 0.40 & 0 & 0 & 0.25 & 0 & 0 & 0.85 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.01 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.92 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.01 & 0 & 0 & 0 & 0 & 0 & 0 & 0.92 & 0 & 0 & 0.01 \\
0.05 & 0.78 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0 & 0 & 0 \\
0.90 & 0 & 0 & 0.88 & 0.94 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.01 & 0 & 0 \\
0.94 & 0 & 0 & 0.96 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right]
\end{array}
\tag{2.6}
$$

$$
\mathbf{L} = \left[\begin{array}{cccccccccccccc}
3.77 & 0 & 0 & -0.97 & -0.91 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.05 & -0.90 & -0.94 \\
0 & 1.19 & -0.03 & 0 & 0 & 0 & 0 & 0 & -0.37 & 0 & 0 & 0 & -0.78 & 0 & 0 \\
0 & -0.03 & 2.93 & 0 & 0 & -0.96 & 0 & -0.95 & -0.98 & 0 & 0 & 0 & 0 & 0 & 0 \\
-0.97 & 0 & 0 & 2.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.88 & -0.96 \\
-0.91 & 0 & 0 & 0 & 1.94 & 0 & 0 & 0 & 0 & 0 & -0.06 & -0.01 & -0.01 & -0.94 & 0 \\
0 & 0 & -0.96 & 0 & 0 & 1.94 & 0 & 0 & -0.97 & -0.01 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1.02 & 0 & 0 & -0.62 & -0.40 & 0 & 0 & 0 & 0 \\
0 & 0 & -0.95 & 0 & 0 & 0 & 0 & 1.89 & -0.94 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -0.98 & 0 & 0 & -0.97 & 0 & -0.94 & 2.89 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -0.37 & 0 & 0 & 0 & -0.01 & -0.62 & 0 & 0 & 1.24 & -0.25 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.06 & 0 & -0.40 & 0 & 0 & -0.25 & 1.56 & 0 & -0.85 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.01 & 0 & 0 & 0 & 0 & 0 & 0 & 0.93 & -0.92 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.01 & 0 & 0 & 0 & 0 & 0 & -0.92 & 0.94 & 0 & -0.01 \\
-0.05 & -0.78 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.85 & 0 & 0 & 1.68 & 0 \\
-0.90 & 0 & 0 & -0.88 & -0.94 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.01 & 0 & 2.74 & 0 \\
-0.94 & 0 & 0 & -0.96 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.91
\end{array}\right] \tag{2.7}
$$

- Vertex-to-vertex lines which indicate physically meaningful connectivity among the sensing points become the **graph edges**, as in Figure 2.2(a).
- The vertices and edges form **a graph**, as in Figure 2.2(b), a new very structurally rich signal domain.
- The graph, rather than a standard vector of sensing points, is then used for analyzing and processing data, as it exhibits both spatial and physical domain awareness.
- The measured temperatures are now interpreted as **signal samples on graph**, as shown in Figure 2.3.
- Similar to traditional signal processing, this new **graph signal** may have many realizations on the same graph and may comprise noise.
- Through relation (2.4), we have therefore introduced a simple **system on a graph** for physically and spatially aware signal averaging (a linear first-order system on a graph).

To emphasize our trust in a particular sensor (i.e., to model sensor relevance), a weighting scheme may be imposed, in the form

$$y(n) = x(n) + \sum_{m \neq n} W_{nm} x(m), \tag{2.8}$$

where W_{nm} are the elements of the weighting matrix, $\mathbf{W}$.

There are three classes of approaches to the definition of graph edges and their corresponding weights, W_{nm}:

- already physically well defined edges and weights,
- definition of edges and weights based on the geometry of vertex positions,
- data similarity based methods for learning the underlying graph topology.

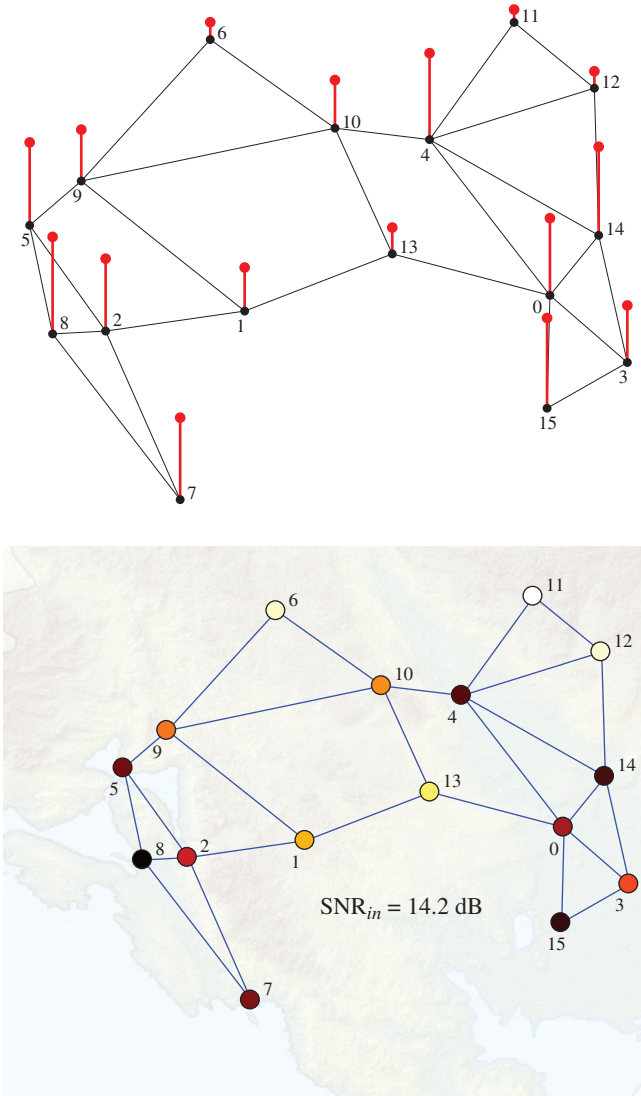


Figure 2.3: From a multi-sensor temperature measurement to a graph signal. The temperature field is represented on a graph that combines the spatially unaware measurements in Figure 2.1(b) and the physically relevant graph topology in Figure 2.2(b). The graph signal values are represented in two ways: (top) by vertical lines for which the length is proportional to the signal values, and (bottom) by using a "hot" colormap to designate the signal values at the vertices.

All the three approaches to define the edge weights are covered in detail in Part III of this monograph.

Since in our case of geographic temperature measurements, the graph weights do not belong to the class of obvious and physically well defined edges and weights, we will employ the "geometry of the vertices" based approach for the definition of the edges and weights. In this way, the weight elements, W_{nm}, for the neighboring vertices are calculated based on the horizontal vertex distance, r_{mn}, and the altitude difference, h_{mn}, as

$$W_{mn} = e^{-\alpha r_{mn} - \beta h_{mn}}, \tag{2.9}$$

where α and β are suitable constants. The so obtained weight matrix, $\mathbf{W}$, is given in (2.6).

Based on (2.4), a weighted graph signal estimator of cumulative temperature now becomes

$$\mathbf{y} = \mathbf{x} + \mathbf{W}\mathbf{x}. \tag{2.10}$$

In order to produce unbiased estimates, instead of the cumulative sums in (2.4) and (2.8), the weighting coefficients within the estimate for each $y(n)$ should sum up to unity. This can be achieved through a normalized form of (2.10), given by

$$\mathbf{y} = \frac{1}{2}(\mathbf{x} + \mathbf{D}^{-1}\mathbf{W}\mathbf{x}), \tag{2.11}$$

where the elements of the diagonal normalization matrix, $\mathbf{D}$, are equal to **the degree matrix** elements, $D_{nn} = \sum_m W_{nm}$, while $\mathbf{D}^{-1}\mathbf{W}$ is a **random walk (diffusion) shift operator** (Stanković *et al.*, 2018b, 2019).

Now that we have defined the graph vertices and edge weights we may resort to the data-agnostic clustering approaches, given in Part I – Section 4.3, to cluster the vertices of this graph based on the graph topology. Figure 2.4 shows the clustering result based on the three smoothest eigenvectors, $\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{u}_3$ (excluding the constant eigenvector, $\mathbf{u}_0$), of the graph Laplacian matrix, $\mathbf{L} = \mathbf{D} - \mathbf{W}$, given in (2.7). Notice that even such a simple graph clustering scheme was capable of identifying different physically meaningful geographic regions.

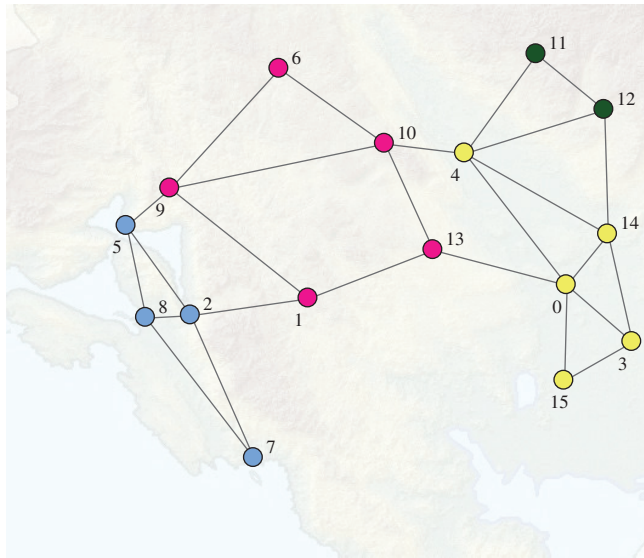


Figure 2.4: Clustering of the graph from Figure 2.2(b) based on the graph Laplacian eigenvectors, $\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{u}_3$. Observe the correct clustering of the graph into the clusters that belong to the seaside area (blue), low mountains (red), low land (yellow), and high mountains (green).

This also means that temperature estimation can roughly be performed within each cluster, which may even be treated as an independent graph (see graph segmentation and graph cuts in Part I, Section 4), rather than over the whole sensor network.

The above-introduced graph data estimation framework is quite general and admits application to many different scenarios where, after identifying a suitable graph topology, we desire to perform estimation on data acquired on such graphs, the subject of this part of the monograph.

3

Signals and Systems on Graphs

In classical data analytics, a signal is sampled at successive, equally spaced, time instants. This then dictates the ordering of signal samples, with $x(n)$ being preceded by $x(n-1)$ and succeeded by $x(n+1)$. The "time distance" between data samples is therefore an inherent parameter in standard data processing algorithms. The relation between sampling instants can also be represented in a graph form, whereby the vertices that correspond to the instants when the signal is sampled and the corresponding edges define the linear sampling (vertex) ordering. The equally spaced nature of sampling instants in classical scenarios can then be represented with equal weights for all edges (for example, normalized to 1), as shown in Figure 3.1.

Algorithms defined in discrete time (like, for example, those based on the DFT or other similar data transforms), usually assume periodicity of the analyzed signals, which means that sample $x(N-1)$ is succeeded by sample $x(0)$, in a perpetual sequence. Notice that this case corresponds to the circular graph, shown in Figure 3.2, which allows us to use this model in many standard data transforms, such as the DFT, DCT, wavelets, and to define graph-counterparts of other processing algorithms, based on these transforms.

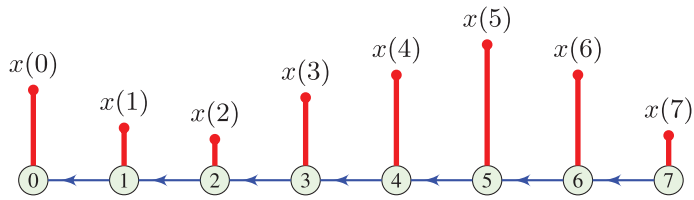


Figure 3.1: Directed path graph representation of a classical time-domain signal defined on an equidistant discrete-time grid.

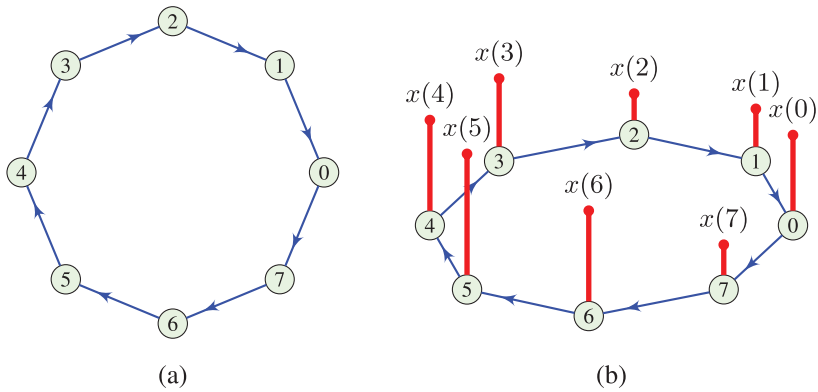


Figure 3.2: Graph representation of periodic data. (a) A directed circular graph. (b) A periodic signal measured on a circular graph. Signal values, $x(n)$, are designated by vertical lines at the corresponding vertex, n.

A signal on a general (including also circular) undirected graph is defined by associating real (or complex) data values, $x(n)$, to each vertex, as shown in Figures 3.3 and 3.4. Such signal values can be arranged in a vector form

$$\mathbf{x} = [x(0), x(1), \ldots, x(N-1)]^T,$$

so that a graph may be considered as a generalized signal domain.

This allows, in general, for any linear processing scheme for a graph signal observed at a vertex, n, to be defined as a linear combination of the signal value, $x(n)$, at this vertex and the signal samples, $x(m)$, at the neighboring vertices, that is

$$y(n) = x(n)h(n,n) + \sum_{m \in \mathcal{V}_n} x(m)h(m,n), \tag{3.1}$$

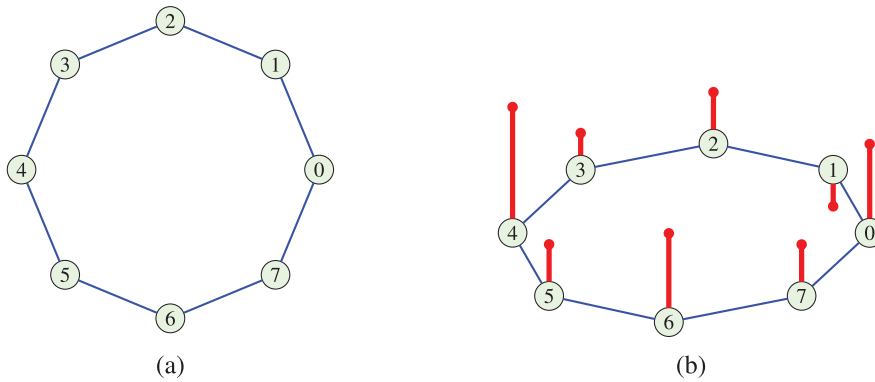


Figure 3.3: Undirected circular graph (a) and signal on the graph (b). Signal values, $x(n)$, are presented as vertical lines at the corresponding vertex, n.

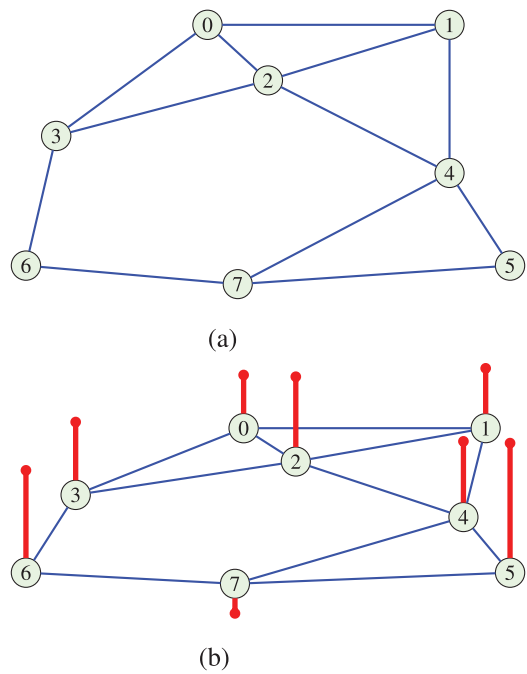


Figure 3.4: Arbitrary undirected graph (a) and signal on graph (b). Signal values, $x(n)$, are presented as vertical lines at the corresponding vertex, n.

where $\mathcal{V}_n$ is the set of vertices in the neighborhood of vertex n, and $h(m,n)$ are the scaling coefficients.

Remark 2: The estimation form in (3.1) is highly vertex-dependent; it is vertex-invariant only in a very specific case of regular graphs, where $\mathcal{V}_n$ is a K-neighborhood of the vertex n, with $h(n,m) = h(n-m)$.

We now proceed to define various forms of vertex-invariant filtering functions, using shifts on a graph. These will then be used to introduce efficient graph signal processing methods (Agaskar and Lu, 2013; Sandryhaila and Moura, 2014a,b; Segarra and Ribeiro, 2016; Venkitaraman *et al.*, 2016; Wang *et al.*, 2016; Yan *et al.*, 2017).

3.1 Adjacency Matrix and Graph Signal Shift

Consider a graph signal, $\mathbf{x}$, for which $x(n)$ is the observed sample at a vertex n. A signal shift on a graph can be defined as movement of the signal sample, $x(n)$, from its original vertex, n, along all walks of length one, that is $K = 1$, that start at vertex n. If the signal shifted in this way is denoted by $\mathbf{x}_1$, then its values can be defined using the graph adjacency matrix, $\mathbf{A}$, as

$$\mathbf{x}_1 = \mathbf{A}\mathbf{x}. \tag{3.2}$$

Example 1: As an illustration of a graph signal and its shifted version, consider the signal on a circular graph from Figure 3.2(a). The original signal, $\mathbf{x}$, is shown in Figure 3.5(a), and its shifted version, $\mathbf{x}_1$, in Figure 3.5(b). Another simple signal on the undirected graph from Figure 3.4(a) is presented in Figure 3.6(a), with its shifted version, $\mathbf{x}_1 = \mathbf{A}\mathbf{x}$, shown in Figure 3.6(b).

A signal shifted by two graph shifts is obtained by further shifting $\mathbf{x}_1 = \mathbf{A}\mathbf{x}$ by one shift. The resulting, twice shifted, graph signal is then given by

$$\mathbf{x}_2 = \mathbf{A}\mathbf{x}_1 = \mathbf{A}(\mathbf{A}\,\mathbf{x}) = \mathbf{A}^2\,\mathbf{x}.$$

Therefore, in general, an m times shifted signal on graph is given by

$$\mathbf{x}_m = \mathbf{A}\mathbf{x}_{m-1} = \mathbf{A}^m\,\mathbf{x}.$$

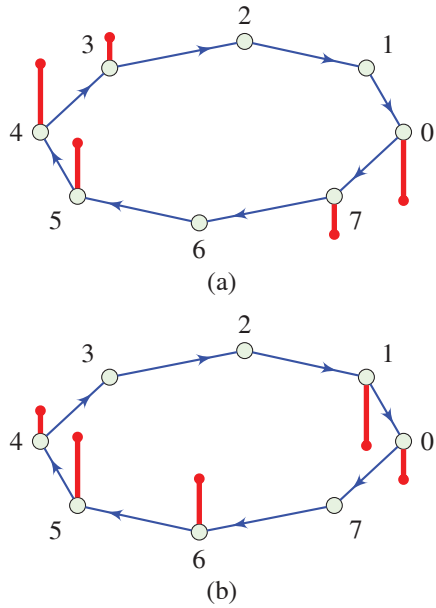


Figure 3.5: Graph shift operator on a directed graph (classical circular shift). (a) Elements of a signal, $\mathbf{x}$, shown as red lines on a directed circular graph. (b) The shifted version, $\mathbf{Ax}$, of the graph signal from (a). The adjacency matrix for this graph is given in (2.14) in Part I.

Remark 3: Like the standard shift operator, the second order shift of a graph signal is obtained by shifting the already once shifted signal. The role of the shift operator is assumed by the adjacency matrix, $\mathbf{A}$.

Remark 4: While this section considers unweighted graphs with the adjacency matrix, $\mathbf{A}$, used as a shift operator, all presented results can be directly applied to the more general class of weighted graphs, where the shift is implemented by the weight matrix, $\mathbf{W}$. The graph Laplacian as a shift operator will be considered in the next section. We will also summarize the various possible shift operators, including those based on the normalized Laplacian and random walk matrices.

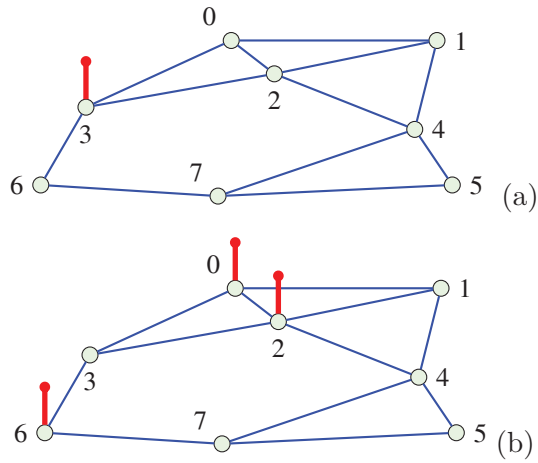


Figure 3.6: Graph signal shift on an undirected graph. (a) A simple signal, $\mathbf{x}$, on an undirected graph. (b) Shifted version, $\mathbf{Ax}$, of the graph signal from (a).

3.2 Systems Based on Graph Shifted Signals

Very much like in standard linear shift-based systems, a system on a graph can be implemented as a linear combination of a graph signal, $\mathbf{x}$, and its graph shifted versions, $\mathbf{A}^m\mathbf{x}$, $m = 1, 2, \ldots, M-1$. The output signal from a system on a graph can then be written as

$$\mathbf{y} = h_0\mathbf{A}^0\mathbf{x} + h_1\mathbf{A}^1\mathbf{x} + \cdots + h_{M-1}\mathbf{A}^{M-1}\mathbf{x} = \sum_{m=0}^{M-1} h_m\mathbf{A}^m\mathbf{x} \qquad (3.3)$$

where $\mathbf{A}^0 = \mathbf{I}$, by definition, and h_0, h_1, $\ldots$, h_{M-1} are the system coefficients. For a circular (classical linear system) graph, this relation reduces to the well known Finite Impulse Response (FIR) filter, given by,

$$y(n) = h_0x(n) + h_1x(n-1) + \cdots + h_{M-1}x(n-M+1). \qquad (3.4)$$

Keeping in mind that the matrix $\mathbf{A}^m$ describes walks of the length $K = m$ in a graph (see Property M_2 in Part I), the output graph signal, $y(n)$, is calculated as a linear combination of the input graph

signal values and the signal values observed at vertices belonging to the $(M-1)$-neighborhood of the considered vertex n.

Remark 5: When the minimal and characteristic polynomials are of the same degree, a physically meaningful system order $(M-1)$ should be lower than the number of vertices N, that is, $M \leq N$. The corresponding condition in classical signal analysis would be that the number, M, of the system impulse response coefficients, h_m, in (3.4) should be lower or equal to the total number of signal samples, N (for the graph in Figure 3.5 it means that the meaningful graph signal shifts are $m = 0, 1, 2, \ldots, N-1$, since the shift for $m = N$ reduces to the shift for $m = 0$, the shift for $m = N+1$ is equivalent to the shift for $m = 1$, and so on). Therefore, in general, the system order $(M-1)$ should be lower than the degree N_m of the minimal polynomial of the adjacency matrix $\mathbf{A}$. For more detail see Part I, Section 3.1.

Remark 6: Any system of order $M - 1 \geq N_m$ can be reduced to a system of order $N_m - 1$.

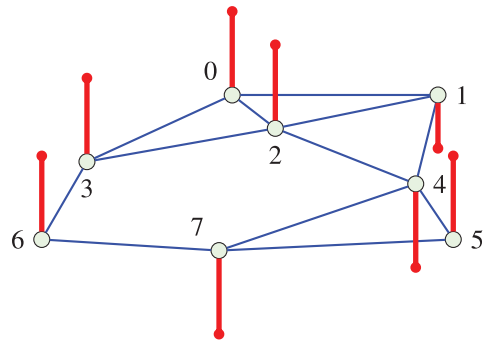
Remark 7: If the system order is greater than or equal to the degree of the minimal polynomial, $M - 1 \geq N_m$, then there exist more than one system producing the same output signal for a given input signal. All such systems on a graph are called equivalent.

The statements in the last three remarks will be addressed in more detail in Section 3.5, with their proofs also provided.

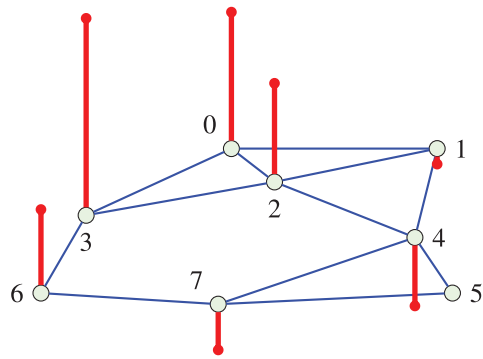
Example 2: Consider a signal on the graph from Figure 3.4(a), given in Figure 3.7(a), and a linear system which operates on this graph, defined by the coefficients $h_0 = 1$, $h_1 = 0.5$. Observe that this system on a graph corresponds to a simple classical first-order weighted moving average system. The output graph signal then represents a weighted average of the signal value at a vertex n and the signal values at its $K = 1$ neighborhood. The output graph signal is shown in Figure 3.7(b).

General system on graph. A system on a graph may be defined in the vertex domain as

$$\mathbf{y} = H(\mathbf{A})\mathbf{x}, \tag{3.5}$$



input signal $\mathbf{x}$ (a)



output signal $\mathbf{y}$ (b)

Figure 3.7: Example of vertex domain signal filtering. (a) An arbitrary graph signal. (b) The output signal obtained through a first-order (averaging) system on a graph, defined as $\mathbf{y} = \mathbf{x} + 0.5\ \mathbf{A}\mathbf{x}$.

where $H(\mathbf{A})$ is a vertex domain system (filter) function. A system on a graph is then linear and shift invariant if it satisfies the following properties of:

1. Linearity
$$H(\mathbf{A})(a_1\mathbf{x}_1 + a_2\mathbf{x}_2) = a_1\mathbf{y}_1 + a_2\mathbf{y}_2.$$

2. Shift invariance
$$H(\mathbf{A})[\mathbf{A}\mathbf{x}] = \mathbf{A}[H(\mathbf{A})\mathbf{x}] = \mathbf{A}\mathbf{y}.$$

Remark 8: A system on a graph defined by

$$H(\mathbf{A}) = h_0\mathbf{A}^0 + h_1\mathbf{A}^1 + \cdots + h_{M-1}\mathbf{A}^{M-1} \tag{3.6}$$

is linear and shift invariant since $\mathbf{A}\mathbf{A}^m = \mathbf{A}^m\mathbf{A}$.

3.3 Graph Fourier Transform (GFT), Adjacency Matrix Based Definition

Classical exploratory data analysis often employs estimation of signals in the spectral (Fourier) domain; this has led to a number of simple and efficient algorithms. While standard spectral analysis employs an equidistant grid in both time and frequency, following the ideas of a system on a graph, we next show that spectral domain representations of graph signals are naturally based on spectral decompositions of the adjacency matrix or graph Laplacian.

The graph Fourier transform of a signal, $\mathbf{x}$, is defined as

$$\mathbf{X} = \mathbf{U}^{-1}\mathbf{x} \tag{3.7}$$

where $\mathbf{X}$ denotes a vector of the GFT coefficients, and $\mathbf{U}$ is a matrix whose columns represent the eigenvectors of the adjacency matrix, $\mathbf{A}$. Denote the elements of the vector $\mathbf{X}$ by $X(k)$, for $k = 0, 1, \ldots, N-1$, and recall that for undirected graphs, the adjacency matrix is symmetric, that is, $\mathbf{A}^T = \mathbf{A}$, and that the eigenmatrices of a symmetric matrix satisfy the property

$$\mathbf{U}^{-1} = \mathbf{U}^T.$$

Remark 9: In the analysis of directed graphs, it is usually assumed that the adjacency matrix, $\mathbf{A}$, (for unweighted graphs) or the weight matrix, $\mathbf{W}$, (for weighted graphs) are diagonizable. However, these matrices are not always diagonizable, and we have to resort to using the standard Jordan normal form (Sandryhaila and Moura, 2014b). A recently proposed pragmatic approach to address this issue is to first employ the Jordan–Chevalley decomposition of a nondiagonizable matrix ($\mathbf{A}$ or $\mathbf{W}$) into its diagonalizable and nilpotent parts, and subsequently use the diagonizable part (corresponding to the diagonal of the Jordan normal form) to define shifts on a (modified) graph (Misiakos *et al.*, 2020).

The element, $X(k)$, of the graph Fourier transform vector, $\mathbf{X}$, therefore represents a projection of the considered graph signal, $x(n)$, onto the k-th eigenvector of $\mathbf{A}$ (a basis function), given by

$$X(k) = \sum_{n=0}^{N-1} x(n) u_k(n). \tag{3.8}$$

In this way, the graph Fourier transform can be interpreted as a set of projections (signal decomposition) onto the set of eigenvectors, $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}$, which serve as orthonormal basis functions.

The inverse graph Fourier transform is then straightforwardly obtained from (3.7) as

$$\mathbf{x} = \mathbf{U}\,\mathbf{X}, \tag{3.9}$$

or element-wise

$$x(n) = \sum_{k=0}^{N-1} X(k) u_k(n). \tag{3.10}$$

Observe that, for example, for a circular graph from Figure 3.2, the graph Fourier transform pair in (3.8) and (3.10) reduces to the standard discrete Fourier transform (DFT) pair. For this reason, the transform in (3.8) and its inverse in (3.10) are referred to as the *graph Fourier transform* (GFT) and the *inverse graph Fourier transform* (IGFT).

3.4 System on a Graph in the GFT Domain

Consider a general system on a graph defined in (3.6),

$$\mathbf{y} = H(\mathbf{A})\mathbf{x} = (h_0\mathbf{A}^0 + h_1\mathbf{A}^1 + \cdots + h_{M-1}\mathbf{A}^{M-1})\mathbf{x}. \tag{3.11}$$

Upon employing the spectral representation of the adjacency matrix, $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$, we have

$$\begin{aligned}\mathbf{y} &= (h_0\mathbf{U}\mathbf{\Lambda}^0\mathbf{U}^{-1} + h_1\mathbf{U}\mathbf{\Lambda}^1\mathbf{U}^{-1} + \cdots + h_{M-1}\mathbf{U}\mathbf{\Lambda}^{M-1}\mathbf{U}^{-1})\mathbf{x} \\ &= \mathbf{U}(h_0\mathbf{\Lambda}^0 + h_1\mathbf{\Lambda}^1 + \cdots + h_{M-1}\mathbf{\Lambda}^{M-1})\mathbf{U}^{-1}\,\mathbf{x} \\ &= \mathbf{U}\,H(\mathbf{\Lambda})\mathbf{U}^{-1}\,\mathbf{x},\end{aligned} \tag{3.12}$$

with the system on a graph transfer function

$$H(\mathbf{\Lambda}) = h_0\mathbf{\Lambda}^0 + h_1\mathbf{\Lambda}^1 + \cdots + h_{M-1}\mathbf{\Lambda}^{M-1}, \tag{3.13}$$

where $\mathbf{\Lambda}$ is the matrix of eigenvalues of $\mathbf{A}$.

A pre-multiplication of this relation with $\mathbf{U}^{-1}$, yields

$$\mathbf{U}^{-1}\mathbf{y} = H(\mathbf{\Lambda})\mathbf{U}^{-1}\,\mathbf{x}. \tag{3.14}$$

From (3.7), the terms $\mathbf{U}^{-1}\,\mathbf{y}$ and $\mathbf{U}^{-1}\,\mathbf{x}$ are respectively the GFTs of the output graph signal, $\mathbf{y}$, and the input graph signal, $\mathbf{x}$, so that the spectral domain system on a graph relation becomes

$$\mathbf{Y} = H(\mathbf{\Lambda})\,\mathbf{X}. \tag{3.15}$$

The output graph signal in the vertex domain can then be calculated as

$$\mathbf{y} = H(\mathbf{A})\mathbf{x} = \text{IGFT}\{H(\mathbf{\Lambda})\,\mathbf{X}\}. \tag{3.16}$$

The element-wise form of the system on a graph in (3.15) is of the form

$$Y(k) = (h_0 + h_1\lambda_k + \cdots + h_{M-1}\lambda_k^{M-1})X(k),$$

where λ_k denotes the kth eigenvalue of the adjacency matrix, $\mathbf{A}$. From (3.13) and the above equation, we can now define the *transfer function of a system on a graph* in the form

$$H(\lambda_k) = \frac{Y(k)}{X(k)} = h_0 + h_1\lambda_k + \cdots + h_{M-1}\lambda_k^{M-1}. \tag{3.17}$$

Remark 10: The classical linear system in (3.4) can be obtained directly from its graph counterpart in (3.17) when the graph is directed and circular. This is because the adjacency matrix of a directed circular graph has eigenvalues $\lambda_k = e^{-j2\pi k/N}$ (see Part I, Section 3.2 for more detail on directed circular graphs), which are identical to the samples on the unit circle in classical DFT.

Similar to the z-transform in classical signal processing, for systems on graphs we can also introduce the system transfer function in the z-domain.

The z-domain transfer function of a system on a graph is defined as

$$H(z^{-1}) = \mathcal{Z}\{h_n\} = h_0 + h_1 z^{-1} + \cdots + h_{M-1}z^{-(M-1)}, \tag{3.18}$$

for $n = 0, 1, \ldots, M-1$. Obviously, from (3.17), we have

$$H(\lambda_k) = H(z^{-1})\big|_{z^{-1}=\lambda_k}.$$

However, the definition of the z-transform for arbitrary graph signals, $x(n)$ and $y(n)$, that would satisfy the relation $Y(z^{-1}) = H(z^{-1})X(z^{-1})$ is not straightforward, which limits the application of the z-transform on graphs. This will be discussed in more detail in Section 3.10.

3.5 Graph Signal Filtering in the Spectral Domain of the Adjacency Matrix

The energy of a graph shifted signal is given by

$$\|\mathbf{x}_1\|_2^2 = \|\mathbf{A}\mathbf{x}\|_2^2 .$$

However, as shown in Figure 3.6, in general, the energy of a shifted signal is not the same as the energy of the original signal, that is

$$\|\mathbf{A}\mathbf{x}\|_2^2 \neq \|\mathbf{x}\|_2^2 .$$

On the other hand, in graph signal processing it is often desirable that a graph shift does not increase signal energy. One such graph shift operator is introduced below.

Remark 11: Using the matrix two-norm it is straightforward to show that the ratio of energies of the graph shifted signal, $\mathbf{A}\mathbf{x}$, and the original graph signal, $\mathbf{x}$, satisfies the relation

$$\max\left\{\frac{\|\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right\} = \max\left\{\frac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\|\mathbf{x}\|_2^2}\right\} = \lambda_{\max}^2, \tag{3.19}$$

where $\lambda_{\max} = \max_k |\lambda_k|,\ k = 0, 1, \ldots, N-1$.

Normalization of the Adjacency Matrix

From (3.19), for the energy of a graph shifted signal, $\|\mathbf{A}\mathbf{x}\|_2^2$, not to exceed the energy of the original graph signal, $\|\mathbf{x}\|_2^2$, we may employ *the normalized adjacency matrix*, defined as

$$\mathbf{A}_{norm} = \frac{1}{\lambda_{\max}}\mathbf{A} \tag{3.20}$$

as a graph shift operator within any system on a graph. While this kind of normalization still does not make the shift on a graph isometric, the

energy of the signal shifted in this way is guaranteed not to be bigger than the energy of the original graph signal, since

$$\|\mathbf{A}_{norm}\mathbf{x}\|_2^2 \leq \|\mathbf{x}\|_2^2 .$$

The equality holds only for a very specific signal which is proportional to the eigenvector that corresponds to $\lambda_{\max}$.

The basic shift on a graph, system on a graph, and graph spectral domain representations can be implemented with the normalized adjacency matrix in (3.20) in the same way as with the original adjacency matrix. *An important property which does not apply to standard adjacency matrices is that the normalization of adjacency matrix yields a simpler eigenvector and eigenvalue ordering scheme, as shown next.*

Spectral Ordering of Eigenvectors of the Adjacency Matrix

For physically meaningful low-pass and high-pass filtering on a graph, we need to establish the notion of spectral order. This, in turn, requires a criterion to classify the eigenvectors (corresponding to the GFT basis functions) into the slow-varying and fast-varying ones.

Remark 12: In classical Fourier analysis, the basis functions are ordered according to their frequency, whereby, for example, low-pass (slow varying) basis functions are harmonic functions characterized by low frequencies. On the other hand, the notion of frequency of the eigenvectors of the graph adjacency matrix, which serve as a basis for signal decomposition, is not defined and we have to find another criterion to classify or rank the eigenvectors. Again, we draw the inspiration from classical Fourier analysis which suggests that the energy of the "signal change" can be used instead of frequency to indicate the rate of change of an eigenvector along time.

Energy of signal change. The *first graph difference* can be defined for graph signals as a difference of the original graph signal and its graph shift, that is,

$$\boldsymbol{\Delta}\mathbf{x} = \mathbf{x} - \mathbf{x}_1 = \mathbf{x} - \mathbf{A}_{norm}\mathbf{x}.$$

In analogy to classical analysis, the energy of signal change can then be defined as the energy of the first difference of a graph signal $\mathbf{x}$, and

takes the form

$$E_{\Delta x} = \|\mathbf{x} - \mathbf{A}_{norm}\mathbf{x}\|_2^2 = \left\|\mathbf{x} - \frac{1}{\lambda_{\max}}\mathbf{A}\mathbf{x}\right\|_2^2.$$

When the graph signal assumes a specific form of an eigenvector, $\mathbf{x} = \mathbf{u}$, of the adjacency matrix, $\mathbf{A}$, the energy of this eigenvector change is equal to

$$E_{\Delta u} = \left\|\mathbf{u} - \frac{1}{\lambda_{\max}}\lambda\mathbf{u}\right\|_2^2 = \left|1 - \frac{\lambda}{\lambda_{\max}}\right|^2, \tag{3.21}$$

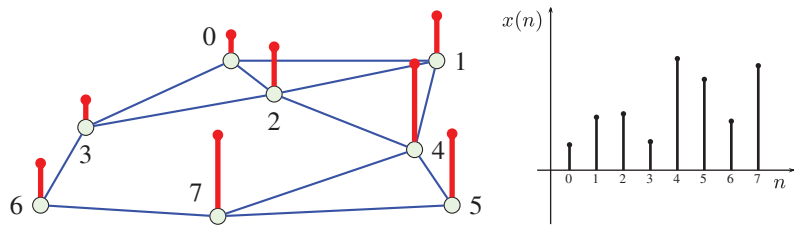
whereby the normalized adjacency matrix, $\mathbf{A}_{norm}$, is used to bound the energy of the shifted graph signal. In the derivation we have also used $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$ and $\|\mathbf{u}\|_2^2 = 1$.

Now, the lower values of $E_{\Delta u}$ indicate that $\mathbf{u}$ is slow-varying, $E_{\Delta u} = 0$ indicates that the signal is constant, while larger values of $E_{\Delta u}$ are associated with fast changes of $\mathbf{u}$ in time. The form in (3.21) is also referred to as *the two-norm total variation of a basis function/eigenvector*. Therefore, if the change in a basis function, $\mathbf{u}$, has a large energy, then the eigenvector, $\mathbf{u}$, can be considered to belong to the higher spectral content of the graph signal.

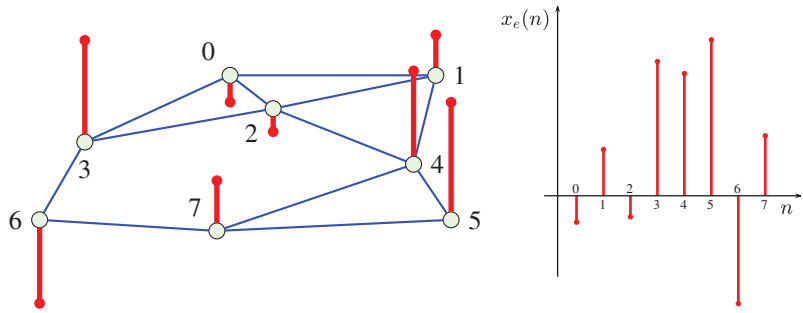
Remark 13: From (3.21), the energy of the rate of change of a graph signal is minimal for $\lambda = \lambda_{\max}$ and it increases as λ decreases (see Figure 3.1 in Part I).

Now that we have established a criterion for the ordering of eigenvectors, based on the corresponding eigenvalues, we shall proceed to define an *ideal low-pass filter on a graph*. The intuition behind low-pass filtering in the graph domain is that such a filter should pass unchanged all signal components (eigenvectors of $\mathbf{A}$) for which the rates of change are slower than that defined by the cut-off eigenvalue, λ_c (*cf.* cut-off frequency), while all signal components (eigenvectors) which exhibit variations which are faster than that defined by the cut-off eigenvalue, λ_c, should be suppressed. The ideal low-pass filter in the graph domain is therefore defined as

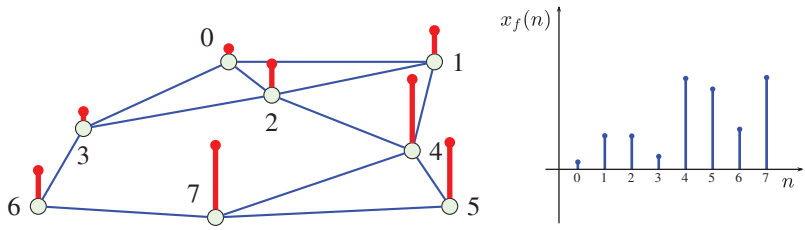
$$f(\lambda) = \begin{cases} 1, & \text{for } \lambda > \lambda_c, \\ 0, & \text{for other } \lambda. \end{cases}$$



(a) original signal, $\mathbf{x} = 3.2\mathbf{u}_7 + 2\mathbf{u}_6$



(b) noisy signal, $\mathbf{x}_\varepsilon = \mathbf{x} + \boldsymbol{\varepsilon}$



(c) filtered signal

Figure 3.8: A low-pass graph signal filtering example. (a) Original signal, $\mathbf{x} = 3.2\mathbf{u}_7 + 2\mathbf{u}_6$. (b) Noisy signal, $\mathbf{x}_\varepsilon = \mathbf{x} + \boldsymbol{\varepsilon}$, at an SNR = 2.7 dB. (c) Filtered signal, at an SNR = 18.8 dB. Ideal low-pass filtering based on the two highest eigenvalues in the pass-band was applied. Note that if $\mathbf{u}_k$ is an eigenvector then $-\mathbf{u}_k$ is also an eigenvector (eigenvectors sign ambiguity).

Example 3: Consider again the undirected graph from Figure 3.4(a) on which we observe a graph signal shown in Figure 3.8(a), which is constructed as a linear combination of two of the eigenvectors of the adjacency matrix of this graph to give $\mathbf{x} = 3.2\mathbf{u}_7 + 2\mathbf{u}_6$ (eigenvectors of the adjacency matrix of the considered graph are presented in Part I,

Figure 3.1). The signal is corrupted by additive white Gaussian noise, $\boldsymbol{\varepsilon}$, at the signal-to-noise (SNR) ratio of $\text{SNR}_{in} = 2.7$ dB and the noisy graph signal, $\mathbf{x}_\varepsilon = \mathbf{x} + \boldsymbol{\varepsilon}$, is shown in Figure 3.8(b). This noisy signal is next filtered using an ideal spectral domain graph filter with a cut-off eigenvalue of $\lambda_c = 1$. The output signal, $\mathbf{x}_f$, is shown in Figure 3.8(c). With $\text{SNR}_{out} = 18.8$ dB, an increase in signal quality of 16.1 dB is achieved with this type of filtering.

Remark 14: The energy of the rate of change of an eigenvector is consistent with the classical DFT based filtering when $\lambda_k = \exp(-j2\pi k/N)$ and $\lambda_{\max} = 1$.

Spectral Domain Filter Design

We shall denote by $G(\mathbf{\Lambda})$ the desired graph transfer function of a system defined on a graph. Then, a system with this transfer function can be implemented either in the spectral domain or in the vertex domain.

In the spectral domain, the implementation is straightforward and can be performed in the following three steps:

1. calculate the GFT of the input graph signal, $\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}$,
2. multiply the GFT of the input graph signal by the graph transfer function, $G(\mathbf{\Lambda})$, to obtain the output spectral form, $\mathbf{Y} = G(\mathbf{\Lambda})\mathbf{X}$, and
3. calculate the output graph signal as the inverse GFT of $\mathbf{Y}$ in Step 2, that is, $\mathbf{y} = \mathbf{U}\mathbf{Y}$.

This procedure may be computationally very demanding for large graphs, where it may be more convenient to implement the desired filter (or its close approximation) directly in the vertex domain.

For the implementation in the vertex domain, the task is to find the coefficients (*cf.* standard impulse response) $h_0, h_1, \ldots, h_{M-1}$ in (3.3), such that their spectral representation, $H(\mathbf{\Lambda})$, is equal (or approximately equal) to the desired $G(\mathbf{\Lambda})$. This is performed in the following way. The transfer function of the vertex domain system is given by (3.17) as $H(\lambda_k) = h_0 + h_1\lambda_k^1 + \cdots + h_{M-1}\lambda_k^{M-1}$ and should be equal to the

desired transfer function, $G(\lambda_k)$, for $k = 0, 1, \ldots, N-1$. This condition leads to a system of linear equations

$$\begin{aligned} h_0 + h_1\lambda_0^1 + \cdots + h_{M-1}\lambda_0^{M-1} &= G(\lambda_0) \\ h_0 + h_1\lambda_1^1 + \cdots + h_{M-1}\lambda_1^{M-1} &= G(\lambda_1) \\ &\vdots \\ h_0 + h_1\lambda_{N-1}^1 + \cdots + h_{M-1}\lambda_{N-1}^{M-1} &= G(\lambda_{N-1}). \end{aligned} \tag{3.22}$$

The matrix form of this system is then

$$\mathbf{V}_\lambda \, \mathbf{h} = \mathbf{g}, \tag{3.23}$$

where $\mathbf{V}_\lambda$ is the Vandermonde matrix form of the eigenvalues λ_k, given by

$$\mathbf{V}_\lambda = \begin{bmatrix} 1 & \lambda_0^1 & \cdots & \lambda_0^{M-1} \\ 1 & \lambda_1^1 & \cdots & \lambda_1^{M-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_{N-1}^1 & \cdots & \lambda_{N-1}^{M-1} \end{bmatrix} \tag{3.24}$$

and

$$\mathbf{h} = [h_0, h_1, \ldots, h_{M-1}]^T \tag{3.25}$$

is the vector of system coefficients which need to be calculated to obtain the desired

$$\mathbf{g} = [G(\lambda_0), G(\lambda_1), \ldots, G(\lambda_{N-1})]^T = \text{diag}(G(\mathbf{\Lambda})). \tag{3.26}$$

Comments on the solution in (3.22):

1. Consider the case with N vertices and with all distinct eigenvalues of the adjacency matrix (in other words, the minimal polynomial is equal to the characteristic polynomial, $P_{min}(\lambda) = P(\lambda)$).

   (a) If the filter order, M, is such that $M = N$, then the solution to (3.22) is unique, since the determinant of the Vandermonde matrix is always nonzero.

   (b) If the filter order, M, is such that $M < N$, then the system in (3.22) is overdetermined. Therefore, the solution to (3.22) can only be obtained in the least squares sense (as described later in this section).

2. If some of the eigenvalues are of a degree higher than one (minimal polynomial order, N_m, is lower than the number of vertices, N) the system in (3.22) reduces to a system of N_m linear equations (by removing multiple equations which correspond to the repeated eigenvalues λ).

   (a) If the filter order, M, is such that $N_m < M \leq N$, the system in (3.22) is underdetermined. In that case $(M - N_m)$ filter coefficients are free variables and the system has an infinite number of solutions, while all so obtained *filters are equivalent*.

   (b) If the filter order is such that $M = N_m$, the solution to the system in (3.22) is unique.

   (c) If the filter order is such that $M < N_m$, the system in (3.22) is overdetermined and the solution is obtained in the least squares sense.

3. Any filter of an order $M > N_m$ has a *unique equivalent filter* of order N_m. This equivalent filter can be obtained by setting the free variables to zero, that is, $h_i = 0$ for $i = N_m, N_m + 1, \ldots, N - 1$.

Finding the system coefficients

Exact solution. For $M = N = N_m$, that is, when the filter order is equal to the number of vertices and the order of minimal polynomial, the solution to the system in (3.22) or (3.23) is unique and is obtained from

$$\mathbf{h} = \mathbf{V}_\lambda^{-1}\mathbf{g}.$$

Least-squares solution. For the overdetermined case, when $M < N_m$, the mean-square approximation of $\mathbf{h} = [h_0, h_1, \ldots, h_{M-1}]^T$ in $\mathbf{V}_\lambda\mathbf{h} = \mathbf{g}$ is obtained by minimizing the squared error

$$e = \|\mathbf{V}_\lambda\mathbf{h} - \mathbf{g}\|_2^2 .$$

From $\partial e/\partial \mathbf{h}^T = \mathbf{0}$ we then have

$$\hat{\mathbf{h}} = (\mathbf{V}_\lambda^T\mathbf{V}_\lambda)^{-1}\mathbf{V}_\lambda^T\mathbf{g} = \text{pinv}(\mathbf{V}_\lambda)\mathbf{g}.$$

Since $M < N_m$, the obtained solution, $\hat{\mathbf{h}}$, is the least-squares approximation for $\mathbf{V}_\lambda \mathbf{h} = \mathbf{g}$. Given that this solution may not satisfy $\mathbf{V}_\lambda \mathbf{h} = \mathbf{g}$, the designed coefficient vector, $\hat{\mathbf{g}}$ (its spectrum $\hat{G}(\mathbf{\Lambda})$), obeys

$$\mathbf{V}_\lambda \hat{\mathbf{h}} = \hat{\mathbf{g}}$$

which, in general, differs from the desired system coefficients, $\mathbf{g}$ (their spectrum $G(\mathbf{\Lambda})$).

Example 4: Consider the unweighted graph from Figure 3.4(a) and the task of the synthesis of a desired filter for which the frequency response is described by

$$\mathbf{g} = [0, 0, 0, 0, 0, 0.5, 1, 1]^T.$$

This filter was designed for various filter orders $M = 1, 2, 4, 6$, using (3.22) and the results are shown in Figure 3.9. For clarity, analytically, the vertex domain realization of the filter with $M = 4$ is given by

$$\mathbf{y} = 0.1734\mathbf{A}^0\mathbf{x} + 0.3532\mathbf{A}^1\mathbf{x} + 0.0800\mathbf{A}^2\mathbf{x} - 0.0336\mathbf{A}^3\mathbf{x},$$

however, the exact frequency response $\hat{\mathbf{g}} = \mathbf{g}$ is only obtained with $M = N = 8$.

Polynomial (Chebyshev) Approximation of the System on a Graph Transfer Function

Without loss of generality, it can be considered that the desired transfer function, $\mathbf{g} = [G(\lambda_0), G(\lambda_1), \ldots, G(\lambda_{N-1})]^T$, consists of samples taken from a continuous function of λ within the interval $\lambda_{\min} \leq \lambda \leq \lambda_{\max}$, where $\lambda_{\min}$ and $\lambda_{\max}$ denote the minimum and maximum values of $\{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\}$, respectively. The variable λ of the desired transfer function, $G(\lambda)$, is continuous, and the system on graph uses only the values at discrete points $\lambda \in \{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\}$. Therefore, for a polynomial approximation, $P(\lambda)$, of the desired transfer function, $G(\lambda)$, it is important that the error at the points within the considered interval, $\lambda_{\min} \leq \lambda \leq \lambda_{\max}$, is bounded and sufficiently small.

This problem is known in algebra as the min-max approximation, and its goal is to find an approximating polynomial that has the smallest maximum absolute error from the desired function value. The min-max polynomials can be approximated by the truncated Chebyshev

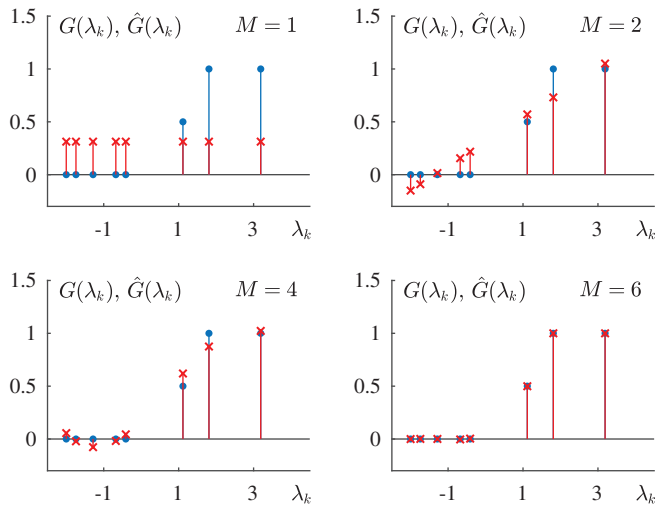


Figure 3.9: Design of a graph filter with a specified transfer function in the spectral domain (*cf.* standard frequency response). The desired spectral response, $G(\lambda_k)$, is denoted by blue circles. Red asterisks designate the spectral response of the filter designed in Example 4, denoted by $\hat{G}(\lambda_k)$, obtained with M filter coefficients, h_0, $h_1, \ldots, h_{M-1}$, in the vertex domain.

polynomials, $P(\lambda)$, which yield approximations of the desired function having almost min-max behavior.

For this the reason, the approximation of the desired transfer function, $G(\lambda)$, may be performed using the truncated Chebyshev polynomial

$$P_{M-1}(z) = \frac{c_0}{2} + \sum_{m=1}^{M-1} c_m T_m(z), \tag{3.27}$$

where $T_m(z)$ are the Chebyshev polynomials defined as

$$\begin{aligned} T_0(z) &= 1, \\ T_1(z) &= z, \\ T_2(z) &= 2z^2 - 1, \\ T_3(z) &= 4z^3 - 3z, \\ &\vdots \\ T_m(z) &= 2zT_{m-1}(z) - T_{m-2}(z), \end{aligned} \tag{3.28}$$

with the variable λ being centered and normalized as

$$z = \frac{2\lambda - (\lambda_{\max} + \lambda_{\min})}{\lambda_{\max} - \lambda_{\min}}, \tag{3.29}$$

such that $-1 \leq z \leq 1$ (required by the Chebyshev polynomial definition). The inverse mapping, from z to λ, is given by

$$\lambda = \frac{1}{2}(z(\lambda_{\max} - \lambda_{\min}) + \lambda_{\max} + \lambda_{\min}).$$

Since the Chebyshev polynomials are orthogonal, with measure $dz/\sqrt{1-z^2}$, the Chebyshev coefficients, c_m, for an expansion of the desired function, $G(z)$, into the polynomial series, $P_{M-1}(z)$, are easily obtained as

$$\begin{aligned} c_m &= \frac{2}{\pi}\int_{-1}^{1} G(z)T_m(z)\frac{dz}{\sqrt{1-z^2}} \\ &= \frac{2}{\pi}\int_{0}^{\pi} \cos(m\theta)G(\cos(\theta))d\theta. \end{aligned}$$

Example 5: Consider the unweighted graph from Figure 3.4(a) with the desired transfer function

$$G(\lambda) = \frac{1 + \text{sign}(\lambda - \lambda_5)}{2}.$$

The samples of $G(\lambda)$ at the discrete points

$$\lambda_k \in \{-2, -1.74, -1.28, -0.68, -0.41, 1.11, 1.81, 3.19\},$$

correspond to the values of $G(\lambda_k)$ in Example 4, Figure 3.9. Since the minimum and maximum eigenvalues are $\lambda_{\min} = -2$ and $\lambda_{\max} = 3.19$, this yields the desired transfer function with a variable z within a normalized interval, $-1 \leq z \leq 1$,

$$G(z) = \frac{1 + \text{sign}(z - z_5)}{2},$$

where z_5 is defined by (3.29) as

$$z_5 = \frac{2\lambda_5 - (\lambda_7 + \lambda_0)}{\lambda_7 - \lambda_0} = 0.2.$$

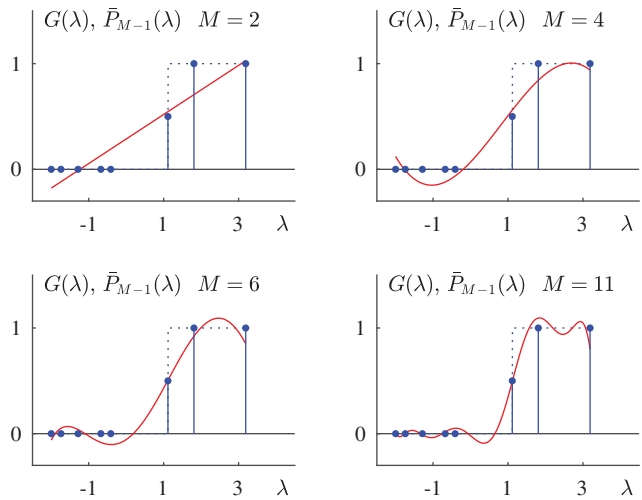


Figure 3.10: Design of a graph filter with a specified transfer function in the spectral domain using the Chebyshev polynomial approximation of order $(M-1)$ with M terms, $T_0(z)$, $T_1(z), \ldots, T_{M-1}(z)$. The desired spectral response, $G(\lambda)$, is denoted by blue dashed line and blue dots. Red lines designate the spectral response of the designed Chebyshev approximation.

The Chebyshev series for $(M-1) = 3$ is given by

$$\begin{aligned} P_{M-1}(z) &= 0.43 + 0.62T_1(z) + 0.12T_2(z) - 0.18T_3(z) \\ &= 0.31 + 1.16z + 0.24z^2 - 0.72z^3. \end{aligned}$$

Upon the change of variables, $z \to \lambda$, we obtain the form

$$\bar{P}_{M-1}(\lambda) = 0.07 + 0.36\lambda + 0.11\lambda^2 - 0.04\lambda^3.$$

The approximations of the spectral domain transfer function of the graph filter, using the Chebyshev polynomial of order $(M-1)$, with M terms, are shown in Figure 3.10, for $M = 2, 4, 6$, and 11.

Graph signal filtering can now be performed in the vertex domain using

$$\mathbf{y} = \bar{P}_{M-1}(\mathbf{A})\mathbf{x},$$

where

$$\bar{P}_{M-1}(\mathbf{A}) = 0.07 + 0.36\mathbf{A} + 0.11\mathbf{A}^2 - 0.04\mathbf{A}^3.$$

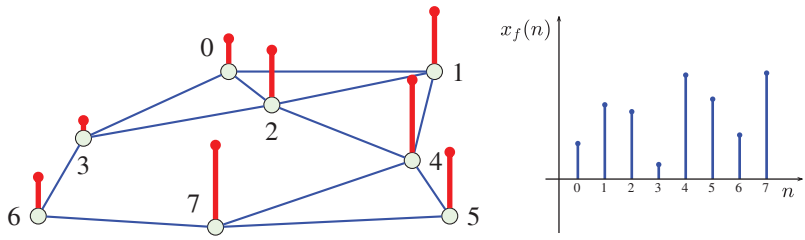


Figure 3.11: Vertex-domain filtering result for the noisy signal from Figure 3.8, using the Chebyshev approximation of the desired transfer function from Figure 3.10 with $M = 4$.

The result of the vertex domain filtering using $\bar{P}_{M-1}(\mathbf{A})$ is shown in Figure 3.11 for the noisy signal from Figure 3.8, with the SNR improvement of 16.76 dB.

Calculation complexity. If the number of nonzero elements in the adjacency matrix, $\mathbf{A}$, is $N_{\mathbf{A}}$, then the number of arithmetic operations (additions) to calculate $\mathbf{Ax}$ is of $N_{\mathbf{A}}$ order. The same number of operations is required to calculate $\mathbf{A}^2\mathbf{x} = \mathbf{A}(\mathbf{Ax})$ using the available $\mathbf{Ax}$. This means that the total number of arithmetic operations (additions) to calculate all $\mathbf{Ax}$, $\mathbf{A}^2\mathbf{x}, \ldots, \mathbf{A}^{M-1}\mathbf{x}$ is of order $MN_{\mathbf{A}}$. Adding these terms requires additional $MN_{\mathbf{A}}$ arithmetic operations (additions), while the calculation of all terms of the form $c_m\mathbf{A}^m\mathbf{x}$ requires an order of $MN_{\mathbf{A}}$ multiplications by constants c_m, $m = 0, 1, \ldots, M-1$. Therefore, to calculate the output graph signal, $\mathbf{y} = \bar{P}_{M-1}(\mathbf{A})\mathbf{x}$, an order of $2MN_{\mathbf{A}}$ additions and $MN_{\mathbf{A}}$ multiplications is needed. Notice that the eigenanalysis of the adjacency matrix, $\mathbf{A}$, requires an order of N^3 arithmetic operations. For large graphs, the advantage in calculation complexity of the vertex domain realization with the polynomial transfer function approximation, $\mathbf{y} = \bar{P}_{M-1}(\mathbf{A})\mathbf{x}$, is obvious.

As is common place in standard filter design theory, the transition intervals of the approximated transfer function, $G(\lambda)$, can be appropriately smoothed, to improve the approximation.

In general, the mapping in (3.29) from λ to z can be written as $z = a\lambda + b$, where $a = 2/(\lambda_{\max} - \lambda_{\min})$ and $b = -(\lambda_{\max} + \lambda_{\min})/(\lambda_{\max} - \lambda_{\min})$.

The Chebyshev polynomial series in λ is then of the form

$$\bar{P}_{M-1}(\lambda) = \frac{c_0}{2} + \sum_{m=1}^{M-1} c_m \bar{T}_m(\lambda), \tag{3.30}$$

with $\bar{T}_0(\lambda) = 1$, $\bar{T}_1(\lambda) = a\lambda + b$, and

$$\bar{T}_m(\lambda) = 2(a\lambda + b)\bar{T}_{m-1}(\lambda) - \bar{T}_{m-2}(\lambda),$$

for $m \geq 2$.

The same relations hold for

$$\bar{P}_{M-1}(\mathbf{A}) = \frac{c_0}{2} + \sum_{m=1}^{M-1} c_m \bar{T}_m(\mathbf{A}). \tag{3.31}$$

This change of variables admits recursive calculation, as in (3.28).

3.5.1 Inverse System on a Graph

A system on a graph, $H(\mathbf{\Lambda})$, which represents an inverse of the system on a graph, given by $G(\mathbf{\Lambda})$, can be obtained from their generic relationship

$$H(\mathbf{\Lambda})G(\mathbf{\Lambda})\mathbf{X} = \mathbf{X}.$$

According to (3.26), this in turn means that if all $G(\lambda_k) \neq 0$ and $P(\lambda) = P_{min}(\lambda)$, then $H(\lambda_k) = 1/G(\lambda_k)$ for each k.

3.6 Graph Fourier Transform Based on the Laplacian

Similar to the graph Fourier transform based on the adjacency matrix, spectral representation of a graph signal can be alternatively based on eigenvalue decomposition of the graph Laplacian, given by

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$$

or $\mathbf{LU} = \mathbf{U}\mathbf{\Lambda}$.

Although the analysis can be conducted in a unified way for spectral decompositions based on both the adjacency matrix and the graph Laplacian, due to their different behavior and scope of application, these will be considered separately.

The graph Fourier transform of a signal, $\mathbf{x}$, which employs the graph Laplacian eigenvalue decomposition to define its basis functions, is

given by

$$\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}, \tag{3.32}$$

where the matrix $\mathbf{U}$ comprises in its columns the eigenvectors of the graph Laplacian. The inverse graph Fourier transform then follows immediately in the form

$$\mathbf{x} = \mathbf{U}\,\mathbf{X}. \tag{3.33}$$

In the case of undirected circular unweighted graph, such as the graph in Figure 3.3(a), this Laplacian based spectral analysis reduces to the standard Fourier transform, but with real-valued basis functions (eigenvectors), as shown in Part I, Section 3.3.2.

3.7 Ordering and Filtering in the Laplacian Spectral Domain

As shown in Section 3.5, *the graph shift and the adjacency matrix are related to the first finite difference of eigenvectors in the vertex domain, while the rate of the eigenvector change is related to its corresponding eigenvalue (cf. standard frequency).* A similar approach can be used for the Laplacian based eigendecomposition. We have seen that for standard time domain signals, the Laplacian of a circle graph represents the second order finite difference, $y(n)$, of a signal $u(n)$, that is

$$y(n) = -u(n-1) + 2u(n) - u(n+1),$$

as shown in Section 3.3 in Part I. A compact expression for all elements of the Laplacian can then be written in a matrix form as $\mathbf{y} = \mathbf{L}\mathbf{u}$. It is now obvious that the eigenvectors, $\mathbf{u}$, which exhibit small variations should also have a small cumulative energy of the second order difference

$$E_u = \sum_n [(u(n) - u(n-1))^2 + (u(n) - u(n+1))^2]/2.$$

Recall that this expression corresponds to the quadratic form of the eigenvector, $\mathbf{u}$, defined by $E_u = \mathbf{u}^T\mathbf{L}\mathbf{u}$.

The above reasoning for the Laplacian quadratic form can also be used for graph signals. As a default case for the Laplacian analysis we will consider undirected weighted graphs, where by definition

$$\mathbf{L}\mathbf{u} = \lambda\mathbf{u}, \qquad \mathbf{u}^T\mathbf{u} = 1$$

or

$$\mathbf{u}^T\mathbf{L}\mathbf{u} = \lambda\mathbf{u}^T\mathbf{u} = \lambda = E_u.$$

This means that the quadratic form of an eigenvector, $\mathbf{u}_k$, is equal to its corresponding eigenvalue. This is elaborated in detail in Section 4.2 in Part I, where we have shown that

$$\mathbf{u}_k^T\mathbf{L}\mathbf{u}_k = \lambda_k = \frac{1}{2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} W_{nm}(u_k(n) - u_k(m))^2 \geq 0. \tag{3.34}$$

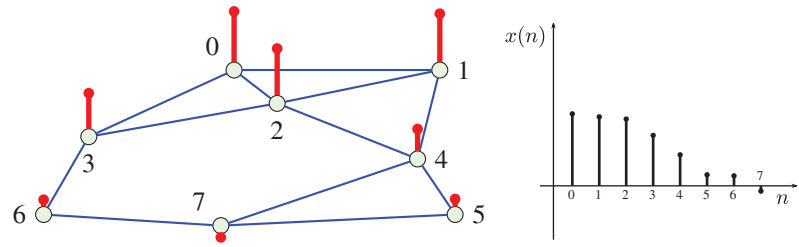
Obviously, a small $\mathbf{u}_k^T\mathbf{L}\mathbf{u}_k = \lambda_k$ implies a small variation of $W_{nm}(u_k(n) - u_k(m))^2$ in the eigenvector $\mathbf{u}_k$, and for each vertex n. Consequently, the eigenvectors corresponding to small λ_k correspond to the low-pass part of a graph signal. In other words, the smaller the smoothness index (curvature), $\mathbf{u}_k^T\mathbf{L}\mathbf{u}_k = \lambda_k$, the smoother the eigenvector, $\mathbf{u}_k$.

An *ideal low-pass filter* in the Laplacian spectrum domain, with a cut-off eigenvalue λ_c, can be therefore defined as

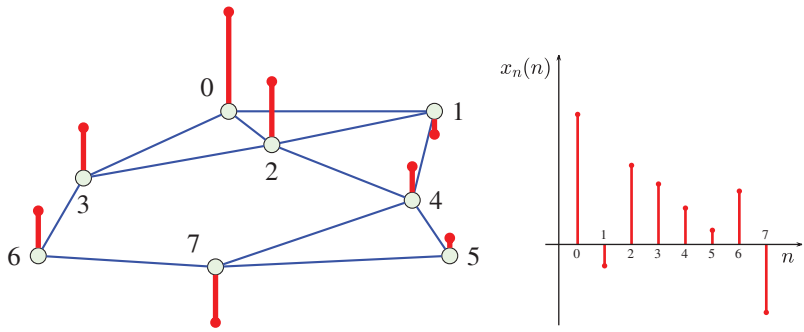
$$f(\lambda) = \begin{cases} 1, & \text{for } \lambda < \lambda_c \\ 0, & \text{for other } \lambda. \end{cases}$$

Example 6: Consider a signal on the undirected graph from Figure 2.2 in Part I, shown in Figure 3.12(a). This graph signal is generated as a linear combination of two Laplacian eigenvectors (which correspond to the slow-varying signal part), to give $\mathbf{x} = 2\mathbf{u}_0 + 1.5\mathbf{u}_1$. The Laplacian eigenvectors of the considered graph are presented in Part I, Figure 3.4, while the considered graph signal is shown in Figure 3.12(a). The original signal, $\mathbf{x}$, was then corrupted by white Gaussian noise at the signal-to-noise ratio of $\text{SNR}_{in} = -1.76$ dB, and shown in Figure 3.12(b). Next, this noisy graph signal was filtered using an ideal spectral domain graph filter, with a cut-off eigenvalue $\lambda_c = 2$, to obtain the filtered signal, $\mathbf{x}_f$, shown in Figure 3.12(c). The so achieved output SNR was $\text{SNR}_{out} = 21.29$ dB, that is, despite its simplicity, the graph filter achieved a gain in SNR of 23.05 dB, as compared to the noisy signal in Figure 3.12(b).

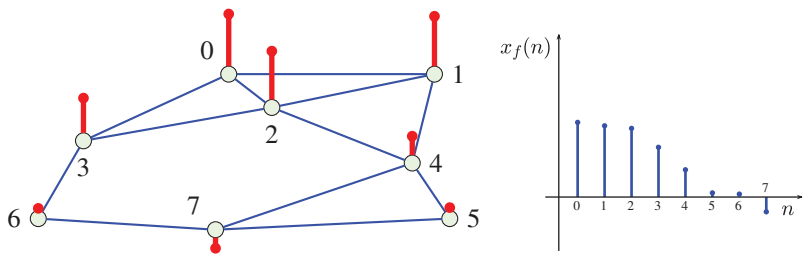
To further illustrate the principle of graph filtering, the noisy signal from Figure 2.3 was filtered using a filter with the spectral cut-off at $\lambda_c = 0.25$ and the result is shown in Figure 3.13. The same signal was



(a) original signal



(b) noisy signal



(c) filtered signal

Figure 3.12: Graph signal filtering example. (a) Original signal. (b) Noisy signal. (c) Filtered signal. Low pass filtering was performed based on the two lowest eigenvalues of the graph Laplacian.

also filtered using a polynomial approximation to the low-pass system, as illustrated in Figure 3.14.

Laplacian versus adjacency-based GFT for regular graphs. A direct relation between the adjacency-based and Laplacian-based spectral decomposition can be established for $\mathcal{J}$-regular unweighted

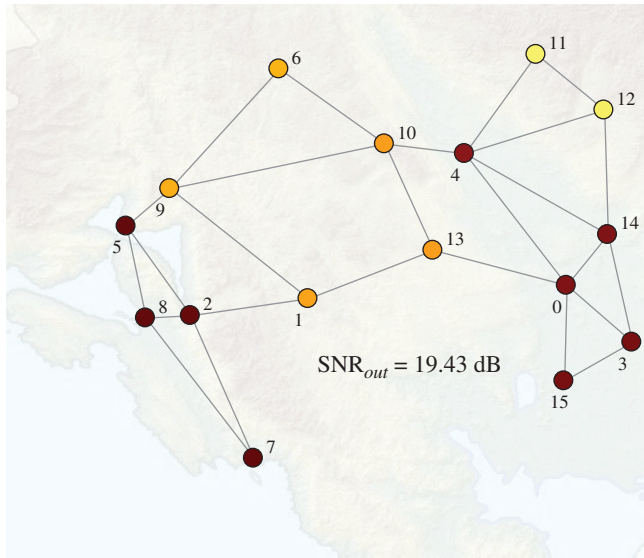


Figure 3.13: Denoising results for the noisy signal from Figure 2.3, which was filtered using a low-pass graph filter with $\lambda_c = 0.25$.

graphs (see Equation (2.13) in Part I), for which

$$\mathbf{L} = \mathcal{J}\mathbf{I} - \mathbf{A}$$

to yield

$$\lambda_k^{(A)} = \mathcal{J} - \lambda_k^{(L)},$$

where the eigenvalues of the adjacency matrix and the graph Laplacian are respectively denoted by $\lambda_k^{(A)}$ and $\lambda_k^{(L)}$, while they share the same eigenvectors.

Remark 15: Rank-ordering of the eigenvectors, $\mathbf{u}_k$, from low-pass to high-pass, which is based on the respective eigenvalues, $\lambda_k^{(A)}$ and $\lambda_k^{(L)}$, yields exactly opposite ordering for these two graph spectral decompositions. For example, the smoothest eigenvector is obtained for $\min_k \lambda_k^{(L)} = \lambda_0^{(L)} = 0$ or for $\max_k \lambda_k^{(A)} = \lambda_{\max} = \mathcal{J} - \lambda_0^{(L)} = \mathcal{J}$.

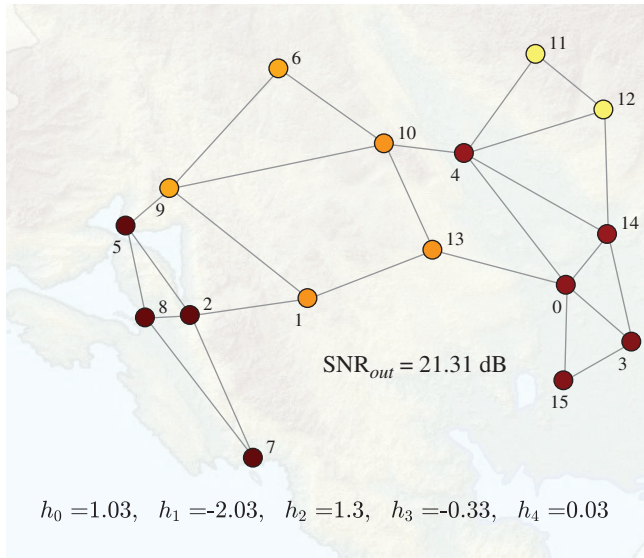


Figure 3.14: Graph filtering of a noisy signal from Figure 2.3, using a fourth-order system given by $\mathbf{y} = h_0\mathbf{L}^0\,\mathbf{x} + h_1\mathbf{L}^1\,\mathbf{x} + h_2\mathbf{L}^2 + h_3\mathbf{L}^3 + h_4\mathbf{L}^4$.

3.8 Systems on a Graph Defined Using the Graph Laplacian

Following on the discussion in Section 3.2 and Equation (3.3), a system on a graph, defined using the graph Laplacian, has the form

$$\begin{aligned}\mathbf{y} &= h_0\mathbf{L}^0\,\mathbf{x} + h_1\mathbf{L}^1\,\mathbf{x} + \cdots + h_{M-1}\mathbf{L}^{M-1}\,\mathbf{x} \\ &= \sum_{m=0}^{M-1} h_m\mathbf{L}^m\,\mathbf{x}.\end{aligned} \tag{3.35}$$

For an unweighted graph, this definition of a system on a graph can be related to the corresponding adjacency matrix form as $\mathbf{L} = \mathbf{D} - \mathbf{A}$.

The **spectral domain description of a system on a graph** is then obtained through the Laplacian eigenvalue decomposition, to yield

$$\begin{aligned}\mathbf{y} = \mathbf{U}\mathbf{Y} &= \sum_{m=0}^{M-1} h_m\mathbf{L}^m\,\mathbf{x} = H(\mathbf{L})\mathbf{x} \\ &= \mathbf{U}H(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x} = \mathbf{U}H(\mathbf{\Lambda})\mathbf{X},\end{aligned} \tag{3.36}$$

where we used the property of the eigendecomposition of matrix polynomial,

$$H(\mathbf{L}) = \sum_{m=0}^{M-1} h_m \mathbf{L}^m = \sum_{m=0}^{M-1} h_m \mathbf{U}\boldsymbol{\Lambda}^m \mathbf{U}^T = \mathbf{U}H(\boldsymbol{\Lambda})\mathbf{U}^T \tag{3.37}$$

described in Section 3.2.3 in Part I, and the notation

$$H(\boldsymbol{\Lambda}) = \sum_{m=0}^{M-1} h_m \boldsymbol{\Lambda}^m \tag{3.38}$$

to obtain

$$\mathbf{Y} = H(\boldsymbol{\Lambda})\mathbf{X}$$

or in an element-wise form

$$Y(k) = H(\lambda_k)X(k), \quad k = 0, 1, \ldots, N-1.$$

In the vertex domain, the n-th element of the output signal, $\mathbf{y} = \mathbf{U}H(\boldsymbol{\Lambda})\mathbf{U}^T\mathbf{x}$, of a system on a graph is given by

$$y(n) = \sum_{k=0}^{N-1} \sum_{i=0}^{N-1} x(i)u_k(i)H(\lambda_k)u_k(n) = \sum_{i=0}^{N-1} x(i)h_n(i), \tag{3.39}$$

for which the transfer function is defined by

$$H(\lambda_k) = h_0 + h_1\lambda_k + \cdots + h_{M-1}\lambda_k^{M-1} \tag{3.40}$$

and the *graph impulse response* is

$$h_n(i) = \sum_{k=0}^{N-1} H(\lambda_k)u_k(n)u_k(i) = \mathcal{T}_n\{h(i)\}. \tag{3.41}$$

Remark 16: The expression for $y(n)$ in (3.39) can be interpreted as a *generalized convolution on graphs*, which is performed using a generalized graph shift of the impulse response, $h_n(i)$, in the vertex domain (see also Part III).

We next proceed to describe the generalized convolution on graphs through responses to the unit delta pulses. For illustration, consider the delta function located at a graph vertex m, given by

$$\delta_m(n) = \begin{cases} 1, & \text{for } m = n \\ 0, & \text{for } m \neq n \end{cases} \tag{3.42}$$

with the corresponding GFT

$$\Delta(k) = \sum_{n=0}^{N-1} \delta_m(n) u_k(n) = u_k(m), \tag{3.43}$$

which is defined based on graph Laplacian eigenvectors.

Observe that, similar to the standard time domain, any graph signal can be written as a sum of delta functions at the graph vertices, that is

$$x(n) = \sum_{i=0}^{N-1} x(i)\delta_n(i)$$

or in a vector form

$$\mathbf{x} = \sum_{i=0}^{N-1} x(i)\boldsymbol{\delta}_i,$$

where $\boldsymbol{\delta}_i$ is a vector with elements $\delta(n-i)$, as in (3.42). Then, the system output, $\mathbf{y}$, takes the form

$$\begin{aligned}\mathbf{y} &= \sum_{m=0}^{M-1} h_m \mathbf{L}^m \mathbf{x} = \mathbf{U} H(\boldsymbol{\Lambda}) \mathbf{U}^T \mathbf{x} \\ &= \sum_{i=0}^{N-1} x(i) \mathbf{U} H(\boldsymbol{\Lambda}) \mathbf{U}^T \boldsymbol{\delta}_i\end{aligned}$$

and its elements are obtained as

$$y(n) = \sum_{i=0}^{N-1} x(i) \sum_{k=0}^{N-1} u_k(n) H(\lambda_k) u_k(i) = \sum_{i=0}^{N-1} x(i) h_n(i),$$

where $h_n(i)$ are related to $H(\lambda_k)$ as in (3.41).

Remark 17: According to (3.36), the form of the graph convolution operator for a vertex n, given in (3.39), is localized within the $(M-1)$-neighborhood of the vertex n. This localization property is even more important for large graphs.

A generalized convolution for two arbitrary graph signals will be addressed next.

3.9 Convolution of Signals on a Graph

Consider two graph signals, $x(n)$ and $h(n)$. A generalized convolution operator for these two signals on a graph is defined using their graph Laplacian spectra (Shuman *et al.*, 2016), based on the assumption that the spectrum of a convolution on a graph

$$y(n) = x(n) * h(n)$$

is equal to the product of the corresponding spectra of graph signals, $x(n)$ and $h(n)$, that is

$$Y(k) = X(k)H(k), \tag{3.44}$$

in the element-wise form. The output of the generalized graph convolution operation, $x(n) * h(n)$, is then equal to the inverse GFT of the spectral product $Y(k)$ in (3.44), that is

$$\begin{aligned} y(n) &= x(n) * h(n) \\ &= \sum_{k=0}^{N-1} Y(k)u_k(n) = \sum_{k=0}^{N-1} X(k)H(k)u_k(n), \end{aligned}$$

where

$$H(k) = \sum_{n=0}^{N-1} h(n)u_k(n). \tag{3.45}$$

Notice the difference between the definition of $H(k)$ in (3.45) and $H(\lambda_k)$ in (3.40). Both these forms will be discussed in more detail in the next section.

Shift on a graph – an alternative definition. The above framework of generalized graph convolution can also serve as a basis for a convenient definition of a shift on a graph. Consider the graph signal, $h(n)$, and the delta function located at a vertex m. Here, we will use $h_m(n)$ to denote the shifted version of the graph signal, $h(n)$, "toward" a vertex m. This kind of shifted signal will be defined following the reasoning in classical signal processing where the shifted signal is obtained as a convolution of the original signal and an appropriately shifted delta function. Therefore, a graph shifted signal is here defined through a generalized graph convolution, $h(n) * \delta_m(n)$, whose GFT is equal to

$H(k)u_k(m)$, according to (3.43) and (3.44). The graph shifted signal is then the IGFT of $H(k)u_k(m)$, that is

$$h_m(n) = h(n) * \delta_m(n) = \sum_{k=0}^{N-1} H(k)u_k(m)u_k(n). \tag{3.46}$$

The same relation follows when calculating the inverse GFT of $X(k)H(k)$, to yield

$$\begin{aligned} y(n) &= \sum_{k=0}^{N-1} X(k)H(k)u_k(n) \\ &= \sum_{k=0}^{N-1}\sum_{m=0}^{N-1} x(m)u_k(m)H(k)u_k(n) \\ &= \sum_{m=0}^{N-1} x(m)h_m(n) = x(n) * h(n), \end{aligned} \tag{3.47}$$

where

$$h_m(n) = \sum_{k=0}^{N-1} H(k)u_k(m)u_k(n) = T_m\{h(n)\} \tag{3.48}$$

is another version of graph shifted signal. Since the definition of $H(k)$ as a GFT of a signal $h(n)$ differs from that in (3.40), these produce different shift operations, which are respectively denoted by $T_m\{h(n)\}$ and $\mathcal{T}_m\{h(n)\}$.

Remark 18: Note that neither of the two shift operations, (3.41) or (3.48), satisfy the property that a shift by 0 is equal to the original signal, $h_0(n) \neq h(n)$.

Example 7: Consider a signal on the graph from Figure 3.4(a), which is defined by its graph Laplacian GFT, given by

$$H(k) = \exp(-2\lambda_k \tau),$$

with $\tau = 0.1573$. All shifted signals, $h_m(n) = T_m\{h(n)\}$, obtained using the shift operator in (3.48), are shown in Figure 3.15.

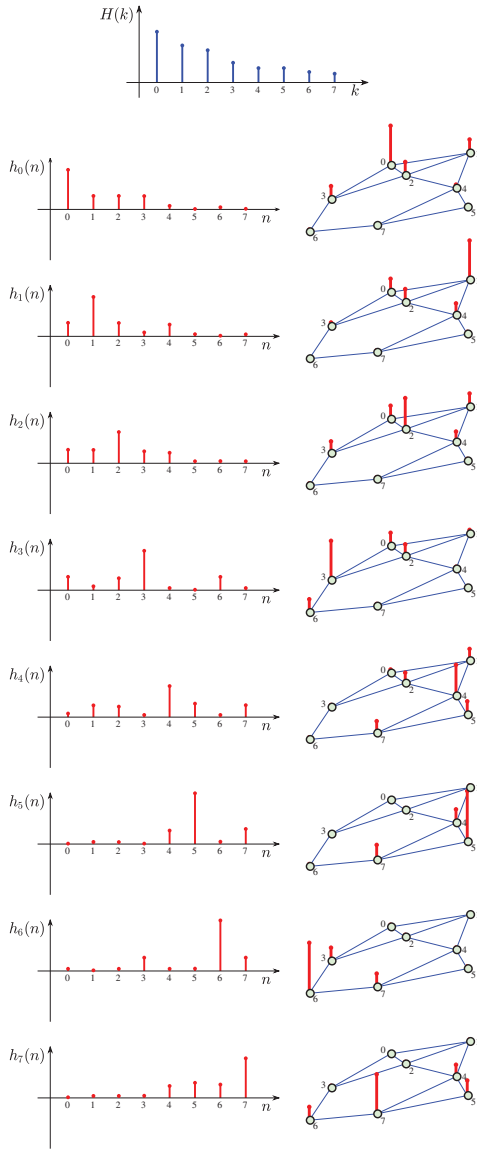


Figure 3.15: An example of graph shift operator. Top: The graph signal defined by its Laplacian GFT, given by $H(k) = \exp(-2\lambda_k \tau)$. Left and right column: The graph signals $h_m(n)$ "shifted" for $m = 0$ to $m = 7$, calculated using $h_m(n) = T_m\{h(n)\}$ in (3.48). The shifted signal is shown both on the vertex index line (left) and on the graph itself (right).

3.10 The z-Transform of a Signal on a Graph

The relation between the graph signal shift operators, $T_m\{h(n)\}$ and $\mathcal{T}_m\{h(n)\}$, which are respectively used to define the generalized convolutions in (3.40) and (3.47), can be established based on the definitions of $H(\lambda_k)$ and $H(k)$. Consider $H(\lambda_k)$, defined by (3.40), as a graph Fourier transform of signal $h(n)$. The samples of the graph signal $h(n)$ are then equal to the IGFT of $H(\lambda_k)$, that is

$$h(n) = \sum_{k=0}^{N-1} H(\lambda_k) u_k(n)$$

while the system coefficients h_n, $n = 0, 1, \ldots, M-1$, are related to $H(\lambda_k)$ by (3.40), that is

$$H(\lambda_k) = h_0 + h_1 \lambda_k + \cdots + h_{M-1} \lambda_k^{M-1}.$$

For $M = N$, the vector forms of the last two relations are

$$[h(0),\ h(1), \ldots, h(N-1)]^T = \mathbf{U} H(\mathbf{\Lambda})$$
$$H(\mathbf{\Lambda}) = \mathbf{V}_\lambda [h_0,\ h_1, \ldots, h_{N-1}]^T$$

so that the signal, $h(n)$, and the coefficients, h_n, can be related as

$$[h_0,\ h_1, \ldots, h_{N-1}]^T = \mathbf{V}_\lambda^{-1} \mathbf{U}^T [h(0),\ h(1), \ldots, h(N-1)]^T. \quad (3.49)$$

Remark 19: In classical DFT (the case of a directed circular graph and its adjacency matrix, when $\mathbf{U}^H$ should be used instead of $\mathbf{U}^T$), the signal samples, $h(n)$, which are obtained as the inverse DFT of $H(\lambda_k)$ and the system coefficients, h_n, are the same, since the eigenvalues are equal to the corresponding shift operators in the spectral domain, $\lambda_k = \exp(-j2\pi k/N)$ and $u_k(n) = \exp(j2\pi nk/N)/\sqrt{N} = \lambda_k^{-n}/\sqrt{N}$, with $h_n = h(n)/\sqrt{N}$ and

$$H(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} h(n) e^{-j2\pi nk/N}.$$

Therefore, for classical DFT analysis, the following relation holds

$$\sqrt{N} \mathbf{V}_\lambda = (\mathbf{U}^H)^{-1}.$$

This relation is obvious from (3.24) and $u_k^*(n) = \lambda_k^n/\sqrt{N}$, and will be used to define *the z-transform of a graph signal.*

The z-transform of graph signals. For a given graph signal $\mathbf{x} = [x(0), x(1), \ldots, x(N-1)]^T$, following the reasoning as in (3.49), the coefficients of a system $[x_0, x_1, \ldots, x_{N-1}]^T$ which corresponds to a system transfer function that would have the same GFT as the graph signal itself are

$$[x_0, x_1, \ldots, x_{N-1}]^T = \mathbf{V}_\lambda^{-1}\mathbf{U}^T[x(0), x(1), \ldots, x(N-1)]^T$$

or

$$[x_0, x_1, \ldots, x_{N-1}]^T = \mathbf{V}_\lambda^{-1}[X(0), X(1), \ldots, X(N-1)]^T.$$

The graph z-transform of a signal $\mathbf{x}$ is therefore equal to the classic z-transform of coefficients $[x_0, x_1, \ldots, x_{N-1}]^T$,

$$X(z^{-1}) = \mathcal{Z}\{x_n\} = x_0 + x_1 z^{-1} + \cdots + x_{N-1} z^{-(N-1)} \tag{3.50}$$

so that the following holds

$$Y(z^{-1}) = H(z^{-1})X(z^{-1}).$$

The output signal, $y(n)$, can now be obtained as

$$[y(0), y(1), \ldots, y(N-1)]^T = \mathbf{U}\mathbf{V}_\lambda[y_0,\ y_1, \ldots, y_{N-1}]^T,$$

where the output graph signal, $y(n)$, results from the inverse z-transform of the coefficients, y_n, that is

$$Y(z^{-1}) = \mathcal{Z}\{y_n\} = y_0 + y_1 z^{-1} + \cdots + y_{N-1} z^{-(N-1)}.$$

The z-transform representation in the complex valued z-domain may be of interest when the eigenvalues are complex-valued, which occurs in the decomposition of adjacency matrices of directed graphs. For example, for the graph from Figure 2.1(b) in Part I and its adjacency matrix, the eigenvalues are shown in Figure 3.16.

Definition: The analytic graph signal, $X_a(k)$, and the graph Hilbert transform, $X_h(k)$, are defined in the spectral domain as

$$\begin{aligned} X_a(k) &= (1 + \operatorname{sign}(\Im(\lambda_k)))X(k) \\ X_h(k) &= j \operatorname{sign}(\Im(\lambda_k))X(k) \\ X(k) &= X_a(k) + jX_h(k), \end{aligned}$$

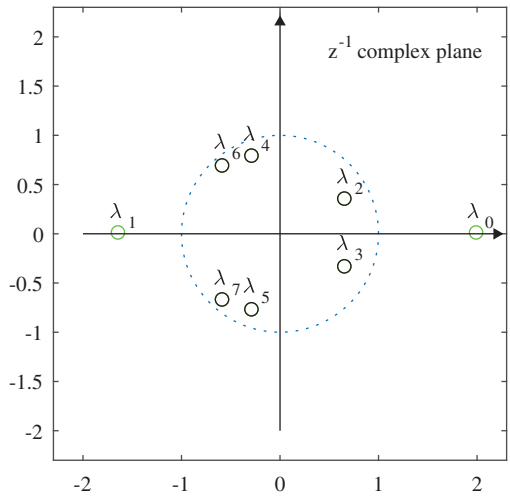


Figure 3.16: Complex eigenvalues of the adjacency matrix of a directed graph in Figure 2.1(b) in Part I.

where $\Im(\lambda_k)$ denotes imaginary part of λ_k. If these relations are applied to the standard DFT with $\lambda_k = e^{-j2\pi k/N}$ we would obtain the corresponding classical signal processing definitions.

3.11 Shift Operator in the Spectral Domain

A shift operation in the spectral domain can be defined in the same way as the shift in the vertex domain. Consider a product of two graph signals, $x(n)y(n)$, defined on an undirected graph. The GFT of this product then takes the form

$$\begin{aligned}\text{GFT}\{x(n)y(n)\} &= \sum_{n=0}^{N-1} x(n)y(n)u_k(n) \\ &= \sum_{n=0}^{N-1}\sum_{i=0}^{N-1} X(i)u_i(n)y(n)u_k(n) = \sum_{i=0}^{N-1} X(i)Y_i(k),\end{aligned}$$

where

$$Y_i(k) = \sum_{n=0}^{N-1} y(n)u_i(n)u_k(n)$$

can be considered as a shift of $Y(k)$ by i spectral indices.

Remark 20: As desired, a shift by $i = 0$ in the spectral domain produces the original value, $Y_0(k) = Y(k)$, up to a constant factor $1/\sqrt{N}$. This relation does not hold for the shift operators in the vertex domain.

3.12 Parseval's Theorem on a Graph

Consider two graph signals, $x(n)$ and $y(n)$, which are observed on an undirected graph and their spectra, $X(k)$ and $Y(k)$. Then, Parseval's theorem has the form

$$\sum_{n=0}^{N-1} x(n)y(n) = \sum_{k=0}^{N-1} X(k)Y(k) \tag{3.51}$$

and it holds for any two graph signals.

To prove Parseval's theorem on graphs, consider

$$\begin{aligned}\sum_{n=0}^{N-1} x(n)y(n) &= \sum_{n=0}^{N-1} \Big[\sum_{k=0}^{N-1} X(k)u_k(n) \Big] y(n) \\ &= \sum_{k=0}^{N-1} X(k) \sum_{n=0}^{N-1} y(n)u_k(n),\end{aligned} \tag{3.52}$$

to yield Parseval's equivalence between the energies in the original vertex and spectral domains. It has been assumed that the graphs are undirected, so that $\mathbf{U}^{-1} = \mathbf{U}^T$ holds. This theorem is quite general and applies to both the graph Laplacian and the adjacency matrix based decompositions on undirected graphs.

3.13 Optimal Denoising

Consider a measurement, $\mathbf{x}$, composed of a slow-varying graph signal, $\mathbf{s}$, and a fast changing disturbance, $\boldsymbol{\varepsilon}$, to give

$$\mathbf{x} = \mathbf{s} + \boldsymbol{\varepsilon}.$$

The aim is to design a filter for disturbance suppression (denoising), the output of which is denoted by $\mathbf{y}$.

The optimal denoising task may then be defined as a minimization of the objective function

$$J = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha \mathbf{y}^T \mathbf{L}\mathbf{y}. \tag{3.53}$$

Physically, the minimization of the first term $\frac{1}{2}\|\mathbf{y}-\mathbf{x}\|_2^2$ forces the output signal $\mathbf{y}$ to be as close as possible to the available observations $\mathbf{x}$, in terms of the energy of their Euclidean distance (minimum error energy), while the second term represent a measure of smoothness of $\mathbf{y}$ (see Section 3.7). This is also physically meaningful, as the original input, $\mathbf{s}$, was low-pass and smoother than the disturbance, $\boldsymbol{\varepsilon}$. The parameter α provides a balance between the closeness of output, $\mathbf{y}$, to $\mathbf{x}$ and the output smoothness criterion.

To solve this minimization problem, we differentiate

$$\frac{\partial J}{\partial \mathbf{y}^T} = \mathbf{y} - \mathbf{x} + 2\alpha \mathbf{L}\mathbf{y} = \mathbf{0}$$

which results in

$$\mathbf{y} = (\mathbf{I} + 2\alpha \mathbf{L})^{-1}\mathbf{x}.$$

The spectral domain form of this relation follows from $\mathbf{L} = \mathbf{U}^T\boldsymbol{\Lambda}\mathbf{U}$, $\mathbf{Y} = \mathbf{U}^T\mathbf{y}$, and $\mathbf{X} = \mathbf{U}^T\mathbf{x}$, to yield

$$\mathbf{Y} = (\mathbf{I} + 2\alpha \boldsymbol{\Lambda})^{-1}\mathbf{X}.$$

The element-wise transfer function of the above spectral input/output relation then takes the form

$$H(\lambda_k) = \frac{1}{1 + 2\alpha\lambda_k}. \tag{3.54}$$

Remark 21: For a small α, we have $H(\lambda_k) \approx 1$, that is, an all-pass behavior of (3.54), with no signal smoothing, which yields $\mathbf{y} \approx \mathbf{x}$. On the other hand, for a large α, $H(\lambda_k) \approx \delta(k)$. The resulting $\mathbf{y} \approx const.$ is maximally smooth (a constant output, without any variation).

Example 8: The noisy signal from Figure 2.3 was filtered using the optimal filter in (3.54) with $\alpha = 1$, and the result is shown in Figure 3.17. The achieved SNR was 19.16 dB.

Other cost functions. Among many possible alternatives, we will introduce two more cost functions for graph signal denoising, which exploit different constraints imposed on the solution.

Instead of enforcing the smoothness of the output signal, we may instead desire that its deviation from a linear form (which would satisfy

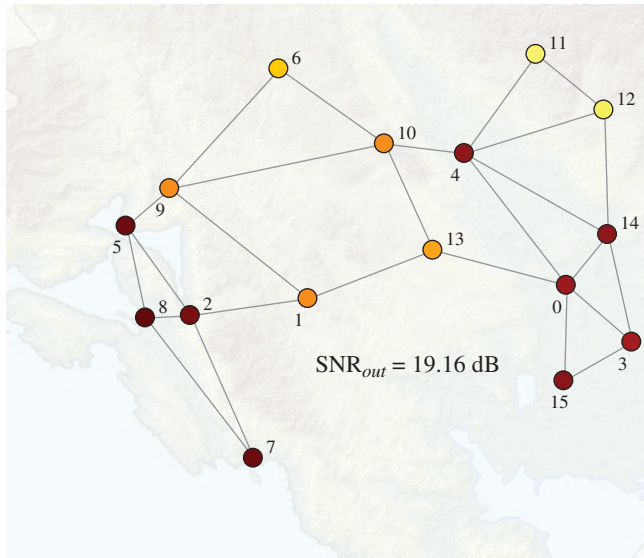


Figure 3.17: Graph signal denoising for a noisy signal from Figure 2.3, which is filtered using an optimal filter in (3.54), with $\alpha = 1$.

$\mathbf{Ly} = \mathbf{0}$) is as small as possible. This can be achieved with the cost function given by

$$J = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha\|\mathbf{Ly}\|_2^2 = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha\mathbf{y}^T\mathbf{L}^2\mathbf{y} \tag{3.55}$$

which yields a closed form denoising solution

$$\mathbf{y} = (\mathbf{I} + 2\alpha\mathbf{L}^2)^{-1}\mathbf{x}$$

with the corresponding element-wise spectral domain relation $H(\lambda_k) = 1/(1 + 2\alpha\lambda_k^2)$.

A combination of the two cost function forms in (3.53) and (3.55), may provide additional flexibility in the design of the filter transfer function, for example

$$J = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha\mathbf{y}^T\mathbf{Ly} + \beta\mathbf{y}^T\mathbf{L}^2\mathbf{y}$$

would yield the transfer function

$$H(\lambda_k) = \frac{1}{1 + 2\alpha\lambda_k + 2\beta\lambda_k^2}.$$

This transfer function form can be further fine-tuned through the choice of the parameters α and β. For example, if we desire the component corresponding to $\lambda_1 \neq 0$ not to be attenuated, we would use $\alpha+\beta\lambda_1 = 0$. Such a cost function can be straightforwardly extended to produce a transfer function for M unattenuated components.

Sparsity promoting solutions. Some applications require to promote the sparsity of the output graph signal, rather than its smoothness. Such solutions then naturally rest upon compressive sensing theory which requires the two-norm in the previous cost functions to be replaced with the norms that promote sparsity. Two examples of such cost functions are

$$J = \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \alpha\|\mathbf{L}\mathbf{y}\|_p^p \tag{3.56}$$

and

$$J = \frac{1}{2}\sum_{n=0}^{N-1}(y(n)-x(n))^2 + \alpha \sum_{n=0}^{N-1}\left(\sum_{m=0}^{N-1} W_{mn}(y(n)-y(m))^2\right)^{p/2} \tag{3.57}$$

with $0 \leq p \leq 1$.

Remark 22: The zero-norm, ℓ_0, with $p = 0$, is the best in promoting sparsity, since for $p = 0$ the second term in the cost function in (3.56) counts (and minimizes) the number of nonzero elements in $\mathbf{L}\mathbf{y}$. Minimization of the sparsity of $\mathbf{L}\mathbf{y}$ promotes constant (or linear) solutions for $\mathbf{y}$, with the smallest number of discontinuities (nonzero elements of vector $\mathbf{L}\mathbf{y}$). In the second cost function in (3.57), the zero-norm promotes the smallest possible number of nonzero elements of the term $\sum_{m=0}^{N-1} W_{mn}(y(n) - y(m))^2$; this is also known as the total variation (TV) approach. However, the minimization of such objective functions cannot be achieved in an analytic way, like in the standard MSE case of $p = 2$.

On the other hand, the choice of $p = 1$ with one-norm, ℓ_1, makes the above cost functions convex, allowing for gradient descend methods be used to arrive at the solution, while producing the same solution as with $p = 0$, under some mild conditions. The ℓ_1-norm serves as an analytic proxy to the ℓ_0-norm (Kim *et al.*, 2009).

3.14 Summary of Shift Operators for Systems on a Graph

The most common choices for the graph shift operator are: (i) adjacency matrix, $\mathbf{S} = \mathbf{A}$, and (ii) graph Laplacian, $\mathbf{S} = \mathbf{L}$.

Various other operators can and have been used as shift operators in systems on a graph, like: (a) normalized versions of the adjacency matrix, (b) normalized graph Laplacian, $\mathbf{S} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$, (c) random walk (diffusion) matrix, $\mathbf{S} = \mathbf{D}^{-1}\mathbf{W}$ (Heimowitz and Eldar, 2017; Stanković *et al.*, 2019a), signed Laplacian, and Laplacian for directed graphs.

Various shift operators produce corresponding eigenvector (signal decomposition) bases, such as those analyzed in Part I and given in Table 3.1.

A generalized form of the output from a system on a graph can then be written as

$$\mathbf{y} = h_0\mathbf{S}^0\,\mathbf{x} + h_1\mathbf{S}^1\,\mathbf{x} + \cdots + h_{M-1}\mathbf{S}^{M-1}\,\mathbf{x} = \sum_{m=0}^{M-1} h_m\mathbf{S}^m\,\mathbf{x}, \qquad (3.58)$$

where, by definition $\mathbf{S}^0 = \mathbf{I}$, while h_0, h_1, ..., h_{M-1} are the system coefficients.

In the next sections we will consider in detail the adjacency matrix of unweighted (directed and undirected graphs) and graph Laplacian of directed graphs.

Table 3.1: Summary of graph spectral basis vectors

Operator	Eigenanalysis
Graph Laplacian	$\mathbf{L}\mathbf{u}_k = \lambda_k\mathbf{u}_k$
Generalized eigenvectors of graph Laplacian	$\mathbf{L}\mathbf{u}_k = \lambda_k\mathbf{D}\mathbf{u}_k$
Normalized graph Laplacian	$\mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}\mathbf{u}_k = \lambda_k\mathbf{u}_k$
Adjacency matrix	$\mathbf{A}\mathbf{u}_k = \lambda_k\mathbf{u}_k$
Normalized adjacency matrix	$\left(\frac{1}{\lambda_{\max}}\mathbf{A}\right)\mathbf{u}_k = \lambda_k\mathbf{u}_k$
Laplacian for directed graphs $\mathbf{L} = \mathbf{D}^{in} - \mathbf{W}$	$\mathbf{S}\mathbf{u}_k = \lambda_k\mathbf{u}_k$ $\mathbf{S} = \mathbf{I} - \mathbf{L}$
Sign Laplacian $\mathbf{L}_a = \mathbf{D}_a - \mathbf{W}$	$\mathbf{L}_a\mathbf{u}_k = \lambda_k\mathbf{u}_k$ $D_a(m,m) = \sum_{n=0}^{N-1} \lvert W_{mn}\rvert$

4

Subsampling, Compressed Sensing, and Reconstruction

Graphs may comprise of a very large number of vertices, of the order of millions or even higher. The associated computational and storage issues bring to the fore the consideration of potential advantages of signal subsampling and compressive sensing defined on graphs. We here present several basic approaches to subsampling, along with their relations to classical signal processing (Anis *et al.*, 2016; Behjat *et al.*, 2016; Chen *et al.*, 2015a,b,c, 2016; Leskovec and Faloutsos, 2006; Marques *et al.*, 2016; Narang and Ortega, 2011, 2012; Nguyen and Do, 2015; Puy *et al.*, 2018; Sakiyama and Tanaka, 2014; Segarra *et al.*, 2015; Stanković, 2015; Stanković *et al.*, 2018c; Tanaka and Eldar, 2020; Tanaka and Sakiyama, 2014; Tremblay and Borgnat, 2016; Tsitsvero *et al.*, 2016; Wang *et al.*, 2015).

4.1 Subsampling of Bandlimited Graph Signals

For convenience, we shall start from the simplest case where the considered graph signal is of a bandlimited nature. Such a signal can be expressed as a linear combination of $K < N$ eigenvectors of the graph

Laplacian which exhibit the lowest smoothness indices,

$$x(n) = \sum_{k=0}^{K-1} X(k)u_k(n), \quad n = 0, 1, \ldots, N-1. \tag{4.1}$$

The GFT domain coefficients of this (K-sparse) signal in the GFT domain are of the following form

$$\mathbf{X} = [X(0), X(1), \ldots, X(K-1), 0, 0, \ldots, 0]^T. \tag{4.2}$$

Recall that a graph signal is sparse in the GFT domain if $K \ll N$. The smallest number of graph signal samples, M, needed to recover the sparse signal is therefore $M = K < N$. For stability of reconstruction, it is common to employ $K \leq M < N$ graph signal samples. The vector of available graph signal samples will be referred to as the *measurement vector*, and will be denoted by $\mathbf{y}$, while the set of vertices (a random subset of $\mathcal{V} = \{0, 1, 2, \ldots, N-1\}$) over which the samples of graph signal are available is denoted by

$$\mathbb{M} = \{n_1, n_2, \ldots, n_M\}.$$

The measurement matrix can now be defined using the IGFT, $\mathbf{x} = \mathbf{U}\,\mathbf{X}$, of which an element-wise form is given by (4.1). The equations in (4.1) corresponding to the available graph signal samples at vertices $n \in \mathbb{M} = \{n_1, n_2, \ldots, n_M\}$ then define the system

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_0(n_1) & u_1(n_1) & \ldots & u_{N-1}(n_1) \\ u_0(n_2) & u_1(n_2) & \ldots & u_{N-1}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_0(n_M) & u_1(n_M) & \ldots & u_{N-1}(n_M) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix},$$

for which the matrix form is given by

$$\mathbf{y} = \mathbf{A}_{MN}\mathbf{X}, \tag{4.3}$$

where $\mathbf{A}_{MN}$ is the *measurement matrix* and the *measurements vector*

$$\mathbf{y} = [x(n_1) \;\; x(n_2) \;\; \ldots \;\; x(n_M)]^T$$

consists of the available graph signal samples. In general, since $M < N$ this system is underdetermined, and cannot be solved uniquely for $\mathbf{X}$ without additional constraints.

The assumption that the spectral representation of a signal contains a linear combination of only $K \leq M$ slowest varying eigenvectors allows us to exclude the GFT coefficients $X(K), X(K+1), \ldots, X(N-1)$ in (4.2) since these are zero-valued and do not contribute to the formation of graph signal samples. With this in mind, the $M \times N$ system of equations in (4.3) is reduced to the following $M \times K$ system

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_0(n_1) & u_1(n_1) & \ldots & u_{K-1}(n_1) \\ u_0(n_2) & u_1(n_2) & \ldots & u_{K-1}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_0(n_M) & u_1(n_M) & \ldots & u_{K-1}(n_M) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(K-1) \end{bmatrix},$$

or, in the matrix form

$$\mathbf{y} = \mathbf{A}_{MK}\mathbf{X}_K, \tag{4.4}$$

where the definitions of the reduced measurement matrix $\mathbf{A}_{MK}$ and the reduced GFT vector $\mathbf{X}_K$ are obvious. For $M = K$ independent measurements, this system can be solved uniquely, while for $M > K$ the system is typically overdetermined and the solution is found in the least squares (LS) sense, as Stanković *et al.* (2018c)

$$\mathbf{X}_K = (\mathbf{A}_{MK}^T\mathbf{A}_{MK})^{-1}\mathbf{A}_{MK}^T\mathbf{y} = \text{pinv}(\mathbf{A}_{MK})\mathbf{y}, \tag{4.5}$$

where $\text{pinv}(\mathbf{A}_{MK}) = (\mathbf{A}_{MK}^T\mathbf{A}_{MK})^{-1}\mathbf{A}_{MK}^T$ is the matrix pseudo-inverse of $\mathbf{A}_{MK}$.

After $\mathbf{X}_K$ is calculated, all GFT values follow directly as $\mathbf{X} = [X(0), X(1), \ldots, X(K-1), 0, 0, \ldots, 0]^T$, where the assumed zero values are added for $X(K)$, $X(K+1)$, $\ldots$, $X(N-1)$. The graph signal is then recovered at all vertices using $\mathbf{x} = \mathbf{U}\,\mathbf{X}$.

Recovery condition. The signal reconstruction in (4.5) is possible if the inverse $(\mathbf{A}_{MK}^T\mathbf{A}_{MK})^{-1}$ exists, which means that

$$\text{rank}(\mathbf{A}_{MK}^T\mathbf{A}_{MK}) = K. \tag{4.6}$$

In terms of the matrix condition number, this requirement is equivalent to

$$\text{cond}(\mathbf{A}_{MK}^T\mathbf{A}_{MK}) < \infty,$$

that is, a nonsingular $\mathbf{A}_{MK}^T\mathbf{A}_{MK}$.

Remark 23: For noisy measurements of graph signals, the noise in the reconstructed GFT coefficients is directly related to the input noise and the matrix condition number. If we are able to choose the available signal sample positions (vertices), then the *sampling strategy* would be to find the set of measurements so that these produce the condition number which is as close to unity as possible (for stability and reduced influence of noise).

Example 9: To demonstrate the principle of reconstruction from a reduced set of graph signal samples, consider the values of a graph signal at $M = 3$ vertices, given by

$$\mathbf{y} = [x(0), x(2), x(6)]^T = [1.140, 0.996, 0.563]^T,$$

as shown in Figure 4.1 (upper panel). Assume that the graph signal is of a bandlimited type, with $K = 2$ lowest nonzero GFT coefficients $X(0)$ and $X(1)$. The GFT coefficients of this graph signal can then be reconstructed from

$$\mathbf{y} = \mathbf{A}_{32}\mathbf{X}_2, \tag{4.7}$$

that follows from the definition in (4.1) for the assumed available signal samples, $x(n)$, at the three vertices $n = 0$, $n = 2$, and $n = 6$, for two nonzero coefficients, $X(0)$ and $X(1)$,

$$\begin{bmatrix} x(0) \\ x(2) \\ x(6) \end{bmatrix} = \begin{bmatrix} u_0(0) & u_1(0) \\ u_0(2) & u_1(2) \\ u_0(6) & u_1(6) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \end{bmatrix}.$$

The rank of the matrix $\mathbf{A}_{32}$ is 2. The corresponding matrix condition number is $\text{cond}(\mathbf{A}_{32}^T\mathbf{A}_{32}) = 4.33$, while the reconstructed nonzero values of the GFT are $X(0) = 2$ and $X(1) = 1$, to yield the reconstructed graph signal $\mathbf{x} = \mathbf{U}\,\mathbf{X}$, with $\mathbf{X} = [2, 1, 0, 0, 0, 0, 0, 0]^T$, as shown in Figure 4.1 (lower panel).

Remark 24: For a directed circular graph, with the eigenvectors $u_k(n) = \exp(j2\pi nk/N)/\sqrt{N}$, the above downsampling and interpolation relations are identical to those in classical signal processing (Stanković, 2015).

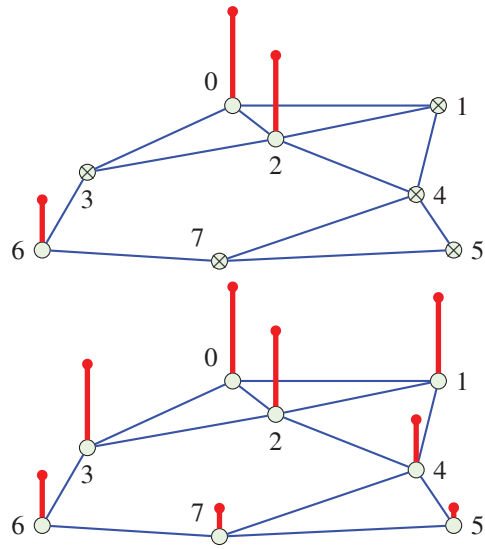


Figure 4.1: Illustration of the subsampling of a lowpass graph signal. Top: A graph signal with missing samples at vertices $1, 3, 4, 5$, and 7. Bottom: The reconstructed graph signal.

4.2 Subsampling of Sparse Graph Signals

The subsampling of graph signals which are sparse in the GFT domain will be next considered for the cases of both known and unknown positions of the nonzero GFT coefficients. This is a generalization of the previous case with bandlimited signals when the positions of nonzero GFT coefficients are assumed to be known and located at the spectral indices from 0 to $K-1$.

Known Coefficient Positions in GFT

The previous analysis in Section 4.1 holds not only for a bandlimited type of the graph signal, $\mathbf{x}$, and its corresponding GFT, $\mathbf{X}$, but also for case of GFT, $\mathbf{X}$, with K nonzero values at arbitrary, but known spectral positions, that is,

$$X(k) = 0 \quad \text{for } k \notin \mathbb{K} = \{k_1, k_2, \ldots, k_K\}.$$

Similar to (4.3), the corresponding system of equations

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_{k_1}(n_1) & u_{k_2}(n_1) & \dots & u_{k_K}(n_1) \\ u_{k_1}(n_2) & u_{k_2}(n_2) & \dots & u_{k_K}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_{k_1}(n_M) & u_{k_2}(n_M) & \dots & u_{k_K}(n_M) \end{bmatrix} \begin{bmatrix} X(k_1) \\ X(k_2) \\ \vdots \\ X(k_K) \end{bmatrix}, \tag{4.8}$$

of which the matrix form is $\mathbf{y} = \mathbf{A}_{MK}\mathbf{X}_K$, is solved for the nonzero spectral values $X(k)$, $k \in \mathbb{K}$, in the same way as in the case of a bandlimited signal presented in Section 4.1.

Support Matrices, Subsampling and Upsampling

In graph signal processing literature, *the subsampling problem is often defined using the so called support matrices* (Chen *et al.*, 2015c; Lorenzo *et al.*, 2018; Tsitsvero *et al.*, 2016). Assume that a graph signal, $\mathbf{x}$, is subsampled in such way that it is available on a subset of vertices $n \in \mathbb{M} = \{n_1, n_2, \dots, n_M\}$, rather than on the full set of vertices. For this subsampled signal, we can define its upsampled version, $\mathbf{x}_s$, by adding zeros at the vertices where the signal is not available. Using a mathematical formalism, the *subsampled and upsampled* version, $\mathbf{x}_s$, of the original signal, $\mathbf{x}$, is then

$$\mathbf{x}_s = \mathbf{B}\mathbf{x}, \tag{4.9}$$

where *the support matrix* $\mathbf{B}$ is an $N \times N$ diagonal matrix with ones at the diagonal positions which correspond to $\mathbb{M} = \{n_1, n_2, \dots, n_M\}$ and zeros elsewhere. The subsampled and upsampled version, $\mathbf{x}_s$, of the signal $\mathbf{x}$ is obtained is such a way that the signal $\mathbf{x}$ is subsampled on a reduced set of vertices, and then upsampled by adding zeros at the original signal positions where the subsampled signal is not defined.

Recall that in general a signal, $\mathbf{x}$, with N independent values cannot be reconstructed from its $M < N$ nonzero values in $\mathbf{x}_s$, without additional constraints. However, for graph signals which are also sparse in the GFT domain, the additional constraint is that the signal, $\mathbf{x}$, has only $K \leq M$ nonzero coefficients in the GFT domain, $\mathbf{X} = \mathbf{U}^T\mathbf{x}$, at $k \in \mathbb{K} = \{k_1, k_2, \dots, k_K\}$, so that the relation

$$\mathbf{X} = \mathbf{C}\mathbf{X}$$

holds, where the support matrix $\mathbf{C}$ is an $N \times N$ diagonal matrix with ones at the diagonal positions which correspond to $\mathbb{K} = \{k_1, k_2, \ldots, k_K\}$ and zeros elsewhere. Note that the presence of the GFT, $\mathbf{X}$, is on both sides of this equation, contrary to $\mathbf{x}_s = \mathbf{B}\mathbf{x}$ in (4.9). The reconstruction formula then follows from

$$\mathbf{x}_s = \mathbf{B}\mathbf{x} = \mathbf{B}\mathbf{U}\mathbf{X} = \mathbf{B}\mathbf{U}\mathbf{C}\mathbf{X}$$

as $\mathbf{X} = \text{pinv}(\mathbf{B}\mathbf{U}\mathbf{C})\mathbf{x}_s$. The inversion

$$\mathbf{X} = \mathbf{C}\mathbf{X} = \text{pinv}(\mathbf{B}\mathbf{U}\mathbf{C})\mathbf{x}_s$$

is possible for K nonzero coefficients of $\mathbf{C}\mathbf{X}$ if the rank of $\mathbf{B}\mathbf{U}\mathbf{C}$ is K (if there are K linearly independent equations), that is

$$\text{rank}(\mathbf{C}) = K = \text{rank}(\mathbf{B}\mathbf{U}\mathbf{C}).$$

This condition is equivalent to (4.6) since the nonzero part of matrix $\mathbf{B}\mathbf{U}\mathbf{C}$ is equal to $\mathbf{A}_{MK}$ in (4.8).

Unknown Coefficient Positions

The reconstruction problem is more complex if the positions of nonzero spectral coefficients $\mathbb{K} = \{k_1, k_2, \ldots, k_K\}$ are not known. This case has been addressed within standard compressive sensing theory and can be formulated as

$$\min \|\mathbf{X}\|_0 \text{ subject to } \mathbf{y} = \mathbf{A}_{MN}\mathbf{X}, \tag{4.10}$$

where $\|\mathbf{X}\|_0$ denotes the number of nonzero elements in $\mathbf{X}$ (ℓ_0 pseudo-norm).

While the ways to solve this minimization problem are manifold, we here adopt a simple, two-step approach:

1. Estimate the positions $\mathbb{K} = \{k_1, k_2, \ldots, k_K\}$ of the nonzero coefficients using $M > K$ signal samples.

2. Reconstruct the nonzero coefficients of $\mathbf{X}$ at the estimated positions $\mathbb{K}$, along with the signal $\mathbf{x}$ at all vertices, using the methods for the reconstruction with the known nonzero coefficient positions, described in Sections 4.1 and 4.2. The nonzero coefficients at positions $\mathbb{K}$ are calculated as $\mathbf{X}_K = \text{pinv}(\mathbf{A}_{MK})\mathbf{y}$.

The nonzero positions of the GFT in Step 1 can be estimated through the projection of measurements (available signal samples), $\mathbf{y}$, on the measurement matrix

$$\mathbf{A}_{MN} = \begin{bmatrix} u_0(n_1) & u_1(n_1) & \dots & u_{N-1}(n_1) \\ u_0(n_2) & u_1(n_2) & \dots & u_{N-1}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_0(n_M) & u_1(n_M) & \dots & u_{N-1}(n_M) \end{bmatrix}$$

to give

$$\mathbf{X}_0 = \mathbf{A}_{MN}^T \mathbf{y}, \tag{4.11}$$

where the positions of K largest values in absolute values of $\mathbf{X}_0$ are used as an estimate of the nonzero positions, $\mathbb{K}$.

This procedure can also be implemented in an iterative way (Stanković *et al.*, 2018c), where:

(i) In the first iteration we assume $K = 1$ and proceed to estimate the largest spectral component (absolute value) in the graph signal. Upon determining its position as $k_1 = \text{argmax}|\mathbf{A}_{MN}^T \mathbf{y}|$, the initially empty set of the nonzero positions becomes $\mathbb{K} = \{k_1\}$. The reconstructed vector $\mathbf{y}_1 = \mathbf{A}_1 \mathbf{X}_1$, where $\mathbf{X}_1 = \text{pinv}(\mathbf{A}_{M1})\mathbf{y}$, is then removed from the measurements, $\mathbf{y}$. In this case, the matrix $\mathbf{A}_{M1}$ is a column of the matrix $\mathbf{A}_{MN}$ defined by the index k_1. The difference $\mathbf{e} = \mathbf{y} - \mathbf{y}_1$ is used as the measurement vector in the next step.

(ii) The position of the second largest spectral component in the graph signal is estimated by solving $k_2 = \text{argmax}|\mathbf{A}_{MN}^T \mathbf{e}|$. The set of nonzero positions now becomes $\mathbb{K} = \{k_1, k_2\}$. The first and the second component of the graph signal are now (re)estimated as $\mathbf{X}_2 = \text{pinv}(\mathbf{A}_{M2})\mathbf{y}$, where the matrix $\mathbf{A}_{M2}$ is a submatrix of the measurement matrix, $\mathbf{A}_{MN}$, which consists of the columns defined by the indices k_1 and k_2. The reconstructed vector $\mathbf{y}_2 = \mathbf{A}_2 \mathbf{X}_2$, is removed from the measurements, $\mathbf{y}$, with the error, $\mathbf{e} = \mathbf{y} - \mathbf{y}_2$, now acting as the new measurement vector.

(iii) The procedure is iteratively repeated K times or until the remaining measurement error values in $\mathbf{e}$ are negligible. In the cases

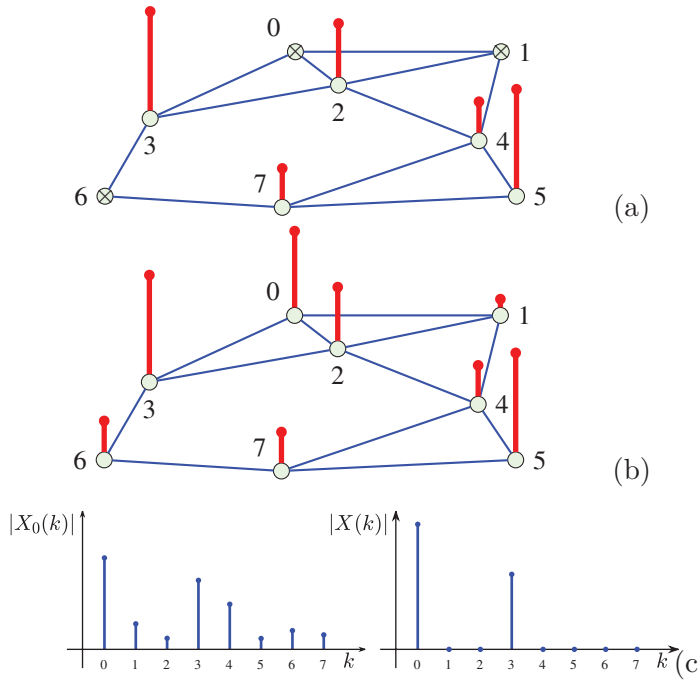


Figure 4.2: Compressive sensing on graphs. (a) Available samples (measurements), $\mathbf{y} = [x(2), x(3), x(4), x(5), x(7)]^T$, with missing samples at $n = 0, 1, 6$. (b) Reconstructed signal, $\mathbf{x}$, over the whole set of vertices. (c) Initial estimate of the GFT, $X_0(k)$, (left), and the reconstructed sparse GFT, $X(k)$, (right).

when the sparsity, K, is unknown, the procedure is iterated until $\|\mathbf{e}\|_2 < \varepsilon$, where ε is a predefined precision.

Example 10: Consider a sparse graph signal, of the sparsity degree $K = 2$, measured at vertices $n = 2, 3, 4, 5$, and 7, which takes the values

$$\mathbf{y} = [0.707, 1.307, 0.407, 1.307, 0.407]^T,$$

as shown in Figure 4.2(a). Our task is to reconstruct the full signal, that is, to find the missing samples $x(0)$, $x(1)$, and $x(6)$.

To estimate positions of the nonzero elements in the GFT, $\mathbf{X}$, the initial estimate, $\mathbf{X}_0$, is calculated for given measurements, $\mathbf{y}$, according to (4.11). The positions of $K = 2$ largest values in $|\mathbf{X}_0|$ are estimated as positions of the nonzero elements in $\mathbf{X}$. In the considered case,

$\mathbb{K} = \{k_1, k_2\} = \{0, 3\}$, as shown in Figure 4.2(c). The GFT coefficients are then reconstructed for the sparsity degree $K = 2$, as $\mathbf{X}_2 = \text{pinv}(\mathbf{A}_{52})\mathbf{y}$, resulting in $X(0) = 2$, $X(3) = 1.2$, as illustrated in Figure 4.2(c–right). Finally, the reconstructed graph signal at all vertices, $\mathbf{x} = \mathbf{U}\mathbf{X}$, is shown in Figure 4.2(b).

Unique Reconstruction Conditions

When the positions of the nonzero coefficients are not known (standard compressive sensing setup), the uniqueness of the solution can be compromised. To this end, it is crucial to establish that, for a given reduced set of the graph signal samples at vertices $\mathbb{M}$, the set of nonzero positions, $\mathbb{K}$, of the sparse vector, $\mathbf{X}$, is unique. In order to define other unique reconstruction conditions, we shall consider again the solution to $\mathbf{y} = \mathbf{A}_{MN}\mathbf{X}$ which assumes a minimum number of nonzero coefficients in $\mathbf{X}$. Assume that the sparsity degree K is known, then a set of K measurements would yield a possible solution, $\mathbf{X}_K$, for any combination of K nonzero coefficients in $\mathbf{X}$. For another set of K measurements, we would obtain another set of possible solutions, $\mathbf{X}_K$. Then, a common solution between these two sets of solutions would be the solution to our problem. For a unique solution, there are no two different K-sparse solutions $\mathbf{X}_K^{(1)}$ and $\mathbf{X}_K^{(2)}$ if all possible matrices, $\mathbf{A}_{M2K}^T\mathbf{A}_{M2K}$, are nonsingular. Namely, both of these two different solutions would satisfy measurement equations,

$$\mathbf{A}_{M2K}\begin{bmatrix}\mathbf{X}_K^{(1)}\\ \mathbf{0}_K\end{bmatrix} = \mathbf{y} \quad \text{and} \quad \mathbf{A}_{M2K}\begin{bmatrix}\mathbf{0}_K\\ \mathbf{X}_K^{(2)}\end{bmatrix} = \mathbf{y},$$

where $\mathbf{A}_{M2K} = [\mathbf{A}_{MK}^{(1)} \quad \mathbf{A}_{MK}^{(2)}]$. Obviously, if we subtract these two matrix equations we get a zero-vector on the right-side and a nonzero solution for the resulting vector,

$$\mathbf{X}_{2K} = \begin{bmatrix}\mathbf{X}_K^{(1)}\\ -\mathbf{X}_K^{(2)}\end{bmatrix},$$

requires the zero-valued determinant of $\mathbf{A}_{M2K}$. The nonzero determinant of $\mathbf{A}_{M2K}$ guarantees that two such, nonzero solutions, $\mathbf{X}_K^{(1)}$ and $\mathbf{X}_K^{(2)}$,

cannot exist. If all possible submatrices $\mathbf{A}_{M2K}$ of the measurement matrix $\mathbf{A}_{MK}$ are nonsingular, then two solutions of sparsity K cannot exist, and the solution is unique. The requirement that all reduced measurement matrices corresponding to a $2K$-sparse $\mathbf{X}$ are nonsingular can be written in several forms, listed below

$$\det\{\mathbf{A}_{M2K}^T\mathbf{A}_{M2K}\} = d_1 d_2 \dots d_{2K} \neq 0$$

$$\text{cond}\{\mathbf{A}_{M2K}^T\mathbf{A}_{M2K}\} = \frac{d_{\max}}{d_{\min}} \leq \frac{1+\delta_{2K}}{1-\delta_{2K}} < \infty.$$

These conditions are satisfied if $d_{\min} > 0$.

Remark 25: In classical compressive sensing, it is commonly assumed that the measurement matrix is normalized in such a way that the energy of each column is equal to one. Therefore, to be able to directly use and/or compare the results from classical compressive sensing to those for graph data, it is convenient to normalize the matrix $\mathbf{A}_{MK}$ so that its columns have unit energy. This is equivalent to the condition that all diagonal elements of $\mathbf{A}_{MK}^T\mathbf{A}_{MK}$ are equal to one.

Upon normalization, the measurement relation (4.8), becomes

$$\mathbf{y} = \mathbf{A}_{MK}\mathbf{X}_K = \mathbf{A}_{MK}\mathbf{N}_K^{-1}\mathbf{N}_K\mathbf{X}_K,$$

where $\mathbf{N}_K$ is a diagonal $K \times K$ matrix, of which the elements are equal to the square root of the energy of the corresponding columns in $\mathbf{A}_{MK}$, that is, $N_K(k) = \sqrt{\sum_{m\in\mathbb{M}} |u_k(m)|^2}$. Upon introducing $\bar{\mathbf{A}}_{MK} = \mathbf{A}_{MK}\mathbf{N}_K^{-1}$ and $\bar{\mathbf{X}}_K = \mathbf{N}_K\mathbf{X}_K$, with the elements $\bar{u}_k(n) = u_k(n)/N_K(k)$ and $\bar{X}(k) = X(k)N_K(k)$, we obtain

$$\mathbf{y} = \bar{\mathbf{A}}_{MK}\bar{\mathbf{X}}_K. \tag{4.12}$$

With that, we may directly use the standard compressive sensing results derived for the normalized measurement matrices. After the normalized vector of sparse elements, $\bar{\mathbf{X}}_K = \mathbf{N}_K\mathbf{X}_K$, is found, the reconstruction of nonzero elements is given by $\mathbf{X}_K = \mathbf{N}_K^{-1}\bar{\mathbf{X}}_K$, and the vertex domain signal now becomes $\mathbf{x} = \mathbf{U}\mathbf{X}$.

The reconstruction of a K-sparse signal is unique if the restricted isometry property (RIP) is satisfied for a $2K$-sparse signal, that is

$$1-\delta_{2K} \leq d_{\min} \leq \frac{\|\bar{\mathbf{A}}_{M2K}\bar{\mathbf{X}}_{2K}\|_2^2}{\|\bar{\mathbf{X}}_{2K}\|_2^2} \leq d_{\max} \leq 1+\delta_{2K}$$

where d_i are the eigenvalues of $\bar{\mathbf{A}}_{M2K}^T \bar{\mathbf{A}}_{M2K}$, $d_{\min}$ is the minimum eigenvalue, $d_{\max}$ is the maximum eigenvalue, and δ_{2K} is the restricted isometry constant. All these conditions are satisfied if $d_{\min} > 0$ or $0 \leq \delta_{2K} < 1$.

Noisy data require robust estimators, and thus more strict bounds on $d_{\min}$ and δ_{2K}. For example, it has been shown that the condition $0 \leq \delta_{2K} < 0.41$ will guarantee stable inversion of $\mathbf{A}_{M2K}^T \mathbf{A}_{M2K}$ and consequently a robust reconstruction for noisy signals; in addition, this bound will allow for convex relaxation of the reconstruction problem (Candes, 2008). Namely, the previous problem, (4.10), can be solved using the convex relaxation from the norm-zero to a norm-one formulation given by

$$\min \|\mathbf{X}\|_1 \text{ subject to } \mathbf{y} = \mathbf{A}_{MN}\mathbf{X}. \tag{4.13}$$

The solutions to these two problem formulations are the same if the measurement matrix satisfies the previous conditions, with $0 \leq \delta_{2K} < 0.41$.

Once the reconstruction of graph signals is formulated within the compressive sensing framework in (4.10) and (4.13), it can be solved using various well-established optimization techniques in this field, such as gradient-based approaches, Bayesian-based reconstruction, and linear programming methods (Candes, 2008; Stanković *et al.*, 2018c).

As is the case with the standard compressive sensing problem, the initial GFT estimate, $\mathbf{X}_0$, will produce correct positions of the nonzero elements, $X(k)$, and the reconstruction will be unique, if

$$K < \frac{1}{2}\left(1 + \frac{1}{\mu}\right),$$

where μ is equal to the maximum value of the inner product among any two columns of the measurement matrix, $\bar{\mathbf{A}}_{MN}$ (μ is referred to as the coherence index) (Stanković *et al.*, 2020).

For illustration of the uniqueness of reconstruction, recall that a K-sparse signal can be written as

$$x(n) = \sum_{i=1}^{K} X(k_i) u_{k_i}(n) = \sum_{i=1}^{K} \bar{X}(k_i) \bar{u}_{k_i}(n),$$

of which the initial estimate in (4.11) is equal to $\bar{\mathbf{X}}_0 = \bar{\mathbf{A}}_{MN}^T \mathbf{y} = \bar{\mathbf{A}}_{MN}^T \bar{\mathbf{A}}_{MN} \bar{\mathbf{X}}$, or element-wise

$$\bar{X}_0(k) = \sum_{i=1}^{K} \bar{X}(k_i) \sum_{n \in \mathbb{M}} \bar{u}_k(n) \bar{u}_{k_i}(n) = \sum_{i=1}^{K} \bar{X}(k_i) \mu(k, k_i),$$

where $\mathbb{M} = \{n_1, n_2, \ldots, n_M\}$ and

$$\mu(k, k_i) = \sum_{n \in \mathbb{M}} \bar{u}_k(n) \bar{u}_{k_i}(n).$$

If the maximum possible absolute value of $\mu(k, k_i)$ is denoted by $\mu = \max |\mu(k, k_i)|$ (coherence index of $\mathbf{A}_{MN}$) then, in the worst case scenario, the amplitude of the largest component, $X(k_i)$, (assumed with the normalized amplitude 1), will be reduced for the maximum possible influence of other equally strong (unity) components $1 - (K-1)\mu$, and should be greater than the maximum possible disturbance at $k \neq k_i$, which is $K\mu$. From $1 - (K-1)\mu > K\mu$, the unique reconstruction condition follows; see also Stanković *et al.* (2018c) and Stanković *et al.* (2020).

4.3 Measurements as Linear Combinations of Samples

It should be mentioned that if some spectrum coefficients of a graph signal are strongly related to only a few of the signal samples, then these signal samples may not be good candidates for the measurements.

Example 11: Consider a graph with one of its eigenvectors of the form close to $u_i(n) = \delta(n - m)$. This case is possible on graphs, in contrast to the classic DFT analysis where the basis functions are spread over all sensing instants (vertices). A similar scenario is also possible in wavelet analysis or short time Fourier transforms, which also allow for some of the transform coefficients to be related to only a few of the signal samples. In the assumed simplified case, if a considered sparse signal contains a nonzero coefficient, $X(i)$, corresponding to $u_i(n) = \delta(n - m)$, then all information about $X(i)$ is contained in the graph signal sample $x(m)$ only. This is prohibitive to the principle of reduced number of samples, since an arbitrary set of available samples may not contain $x(m)$.

In classical and graph data analysis this class of problems is solved by defining a more complex form of the measurements, $y(n)$, through linear combinations of all signal samples rather than the original samples themselves. In this way, each measurement, $y(n)$, will contain information about all signal samples, $x(n)$, $n = 0, 1, \ldots, N-1$.

Such measurements are linear combinations of all signal samples, and are given by

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(M) \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & \ldots & b_{1N} \\ b_{21} & b_{12} & \ldots & b_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ b_{M1} & b_{M2} & \ldots & b_{MN} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix},$$

or in a matrix form

$$\mathbf{y} = \mathbf{B}_{MN}\mathbf{x}.$$

The weighting coefficients for the measurements, b_{mn}, in the matrix, $\mathbf{B}_{MN}$, may be, for example, drawn from a Gaussian random distribution.

For reconstruction, the sparsity of a graph signal, $\mathbf{x}$, should be again assumed in the GFT domain. The relation of the measurement vector, $\mathbf{y}$, with this sparsity domain vector of coefficients, $\mathbf{X}$, is then given by

$$\mathbf{y} = \mathbf{B}_{MN}\mathbf{x} = \mathbf{B}_{MN}\mathbf{U}\mathbf{X} = \mathbf{A}_{MN}\mathbf{X}.$$

The reconstruction is now obtained as a solution to

$$\min \|\mathbf{X}\|_0 \text{ subject to } \mathbf{y} = (\mathbf{B}_{MN}\mathbf{U})\mathbf{X}$$

or as a solution of the corresponding convex minimization problem,

$$\min \|\mathbf{X}\|_1 \text{ subject to } \mathbf{y} = (\mathbf{B}_{MN}\mathbf{U})\mathbf{X},$$

as described in Section 4.2.

4.4 Aggregate Sampling

A specific form of a linear combination of graph signals is referred to as *aggregate sampling*.

For clarity, we shall first establish an interpretation of sampling in classical signal processing through its graph counterpart – sampling on

a directed circular graph (Figure 3.2). Consider a graph signal, $\mathbf{x}$, at a vertex/instant n. If the signal is observed at this vertex/instant only, then its value is $y_0(n) = x(n)$. Upon applying the graph shift operator, we have $\mathbf{y}_1 = \mathbf{A}\mathbf{x}$, then for the same vertex, n, we have $y_1(n) = x(n-1)$. If we continue this "shift and observe" operation on the directed circular graph N times at the same vertex/instant, n, we will eventually have all signal values $x(n), x(n-1), \ldots, x(n-N+1)$ observed at the vertex n.

To proceed with signal reconstruction, observe that if the shifts are stopped after $M < N$ steps, the available signal samples will be $x(n), x(n-1), \ldots, x(n-M+1)$. From this reduced set of measurements/samples we can still recover the full graph signal, $\mathbf{x}$, using compressive sensing based reconstruction methods, if the appropriate reconstruction conditions are met.

Principle of aggregate sampling on an arbitrary graph. The same procedure can be applied to a signal observed in the same way on an arbitrary graph. Assume that we observe the graph signal at only one vertex, n, and obtain one graph signal sample

$$y_0(n) = x(n),$$

which will be considered as the measurement $y(0) = y_0(n)$.

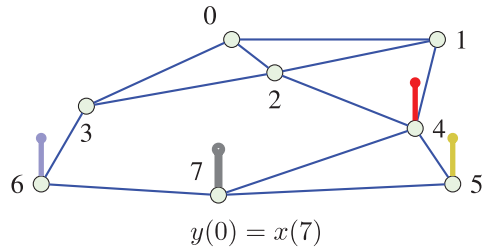
This graph signal may now be "graph shifted" to produce $\mathbf{y}_1 = \mathbf{A}\mathbf{x}$. Recall that in a one-step signal shift on a graph, all signal samples will move by one step along the graph edges, as described in detail in Section 3.1 and illustrated in Figure 4.3. The sample of a graph signal at vertex n will now be a sum of all signal samples that have shifted to this vertex. Its value is obtained as an inner product of the mth row of the adjacency matrix, $\mathbf{A}$, and the original signal vector, $\mathbf{x}$. The value of graph shifted signal at the vertex n, is therefore given by

$$y_1(n) = \sum_m A_{nm} x(m),$$

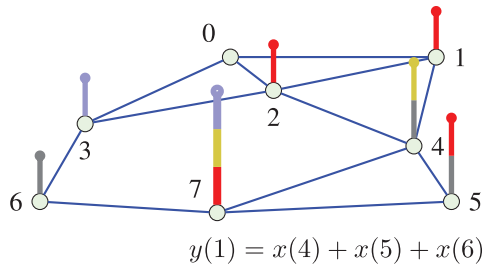
and represents a linear combination of some of the signal samples, which is now considered as the measurement $y(1) = y_1(n)$.

One more signal shift on the graph yields

$$y_2(n) = \sum_m A^{(2)}_{nm} x(m),$$



(a) signal $\mathbf{x}$



(b) shifted signal $\mathbf{Ax}$

Figure 4.3: Principle of aggregate sampling. (a) A graph signal $\mathbf{x}$. (b) Its graph shifted version $\mathbf{Ax}$. For example, for a graph signal value observed at the vertex $n = 7$ in the graph in (a) the measurement is $y(0) = x(7)$, and the aggregate measurement at the same vertex, $n = 7$, after the graph signal is shifted, is equal to $y(1) = x(4) + x(5) + x(6)$ in (b). These two observations, $y(0)$ and $y(1)$, would be sufficient to reconstruct a signal whose sparsity degree is $K = 2$ with nonzero values at the known spectral index positions, k_1 and k_2, if the reconstruction condition (4.6) is satisfied for the matrix $\mathbf{A}_{MN} = \mathbf{B}_{MN}\mathbf{U}$ at the specified spectral index positions.

where $A^{(2)}_{nm}$ are the elements of matrix $\mathbf{A}^2 = \mathbf{AA}$ (see Property M_2 in Part I, Section 2.3). Such an observed value, after two one-step shifts, $y_2(n)$ at a vertex n, represents a new linear combination of some signal samples and will be considered as the measurement $y(2) = y_2(n)$.

If we proceed with shifts $M = N$ times, a system of N linear equations, $\mathbf{y} = \mathbf{B}_{MN}\mathbf{x}$, is obtained from which all signal values, $x(n)$, can be calculated. If we stop at $M < N$, the signal can still be recovered using compressive sensing based reconstruction methods if the signal is sparse and the reconstruction conditions are met.

Instead of M signal samples (instants) at one vertex, we may use, for example, P samples at vertex n and $(M-P)$ samples from a vertex m. Other combinations of vertices and samples may be also used to obtain M measurements and to fully reconstruct a signal.

4.5 Random Sampling with Optimal Strategy

Consider a realistic case of bandlimited signals on a graph. For convenience, assume that they admit a representation through linear combinations of K eigenvectors with the smallest eigenvalues, that is

$$x(n) = \sum_{k=0}^{K-1} X(k) u_k(n).$$

Recall that, in graphs, the basis functions may be highly concentrated at specific vertices; this means that for adequate graph sampling some vertices are more important and are almost "must keep", while some vertices can be omitted (with a higher probability). For example, if one of the eigenvectors, for $k = 0, 1, \ldots, K-1$, is fully concentrated at a certain vertex, then this vertex must be included in the sampling scheme.

To this end, a sampling scheme with an adaptive strategy, proposed by Puy *et al.* (2018), introduces the probability, p_n, of a vertex n being selected in the reduced set of signal samples (measurements) and finds its optimal value using the *graph weighted coherence*.

To clarify this method, consider a signal which is equal to the delta pulse at a vertex $n = m$, that is $x(n) = \delta(n-m)$. In the time domain, the energy of this signal is completely concentrated at the vertex $n = m$, with the GFT of this signal

$$\Delta_m(k) = \sum_{n=0}^{N-1} x(n) u_k(n) = u_k(m).$$

The *local graph weighted coherence* then represents the energy of GFT within the first K eigenvectors, and is given by

$$\|\mathbf{\Delta}_m\|_2^2 = \sum_{k=0}^{K-1} |u_k(m)|^2 = \|\mathbf{U}_K^T \mathbf{x}\|_2^2 \leq 1.$$

The value of energy equal to 1 indicates that there exist a bandlimited signal whose energy is completely concentrated at the vertex $n = m$, and this vertex must be used in any successful sampling scheme. The lower this value, the larger the spread of signal energy over vertices, so that we can randomly pick any of these vertices.

With the probability of picking a vertex, p_m, the graph weighted coherence can now be defined as

$$\nu_K = \max_m \{p_m^{-1/2} \|\mathbf{\Delta}_m\|_2^2\}.$$

The optimal sampling distribution, p_m^*, that minimizes the graph weighted coherence is then equal to

$$p_m^* = \frac{1}{K} \|\mathbf{\Delta}_m\|_2.$$

This can be seen by recognizing that the energy of K normalized eigenvectors equals to K, that is

$$K = \sum_{m=0}^{N-1} \sum_{k=0}^{K-1} |u_k(m)|^2$$

since $\sum_{m=0}^{N-1} |u_k(m)|^2 = 1$, by definition. The above expression for p_m^* follows from

$$\begin{aligned} K = \sum_{m=0}^{N-1} \sum_{k=0}^{K-1} |u_k(m)|^2 &= \sum_{m=0}^{N-1} \sum_{k=0}^{K-1} p_m \frac{|u_k(m)|^2}{p_m} \\ &\leq \max_m \left\{ \sum_{m=0}^{N-1} \sum_{k=0}^{K-1} \frac{|u_k(m)|^2}{p_m} \right\} \sum_{m=1}^{N-1} p_m = \nu_K^2. \end{aligned}$$

The sampling distribution, p_n^*, is optimal in the sense that the number of measurements needed to embed the set of K-bandlimited signals is effectively reduced to its minimum value (Puy *et al.*, 2018).

Example 12: Consider a very simple case of a graph with three disconnected components (sub-graphs). The number of vertices in these graph components is $N_1 = 1$ vertex, $N_2 = 2$ vertices, and $N_3 = 4$ vertices, denoted by $n = 0, 1, 2, 3, 4, 5, 6$, respectively. Assume that the signal is constant over the three graph components, and only the three ($K = 3$) lowest eigenvalues with $\lambda_0 = \lambda_1 = \lambda_2 = 0$ are considered.

The corresponding nonzero eigenvector elements are $u_0(n) = 1$, for $n = 0$, $u_1(n) = 1/\sqrt{2}$, for $n = 1, 2$, and $u_2(n) = 1/\sqrt{4}$, for $n = 3, 4, 5, 6$.

Intuitively, in order to recover this graph signal we must have at least $M = 3$ samples; also each graph component should contain a sample.

We will now compare the two strategies: (1) Fully random selection of samples; and (2) Selection of samples with probabilities defined as

$$p_m^* = \frac{1}{K}\|\mathbf{\Delta}_m\|_2^2 = \frac{1}{K}\sum_{k=0}^{K-1}|u_k(m)|^2 = \frac{1}{3}\sum_{k=0}^{2}|u_k(m)|^2,$$

which results in $p_m^* = (1/3, 1/6, 1/6, 1/12, 1/12, 1/12, 1/12)$ for $m = (0, 1, 2, 3, 4, 5, 6)$. Obviously, when using the optimal sampling strategy the probability of selecting $M = 3$ samples from different sets is much higher than when the samples are chosen randomly with $p_m = (1/7, 1/7, 1/7, 1/7, 1/7, 1/7, 1/7)$.

This kind of variable sampling density is also used in classical compressive sensing to improve the density of samples where this is needed due to signal variations. Two variants of this approach are used: (1) with vertex replacement, when every selected vertex can be chosen again; and (2) without vertex replacement, when a vertex can be selected only once. Within the compressive sensing framework, taking some samples several times means that these vertices are considered with higher importance. Since the selection process accounts for the importance of the vertices, we will assume the latter approach.

For large graphs, it is also very important to try to avoid the eigenvector calculation and even to estimate the factor $\|\mathbf{\Delta}_m\|_2^2$ without the eigendecomposition. One such method was introduced by Puy *et al.* (2018) using random vectors and the property that an average of the power of several random vectors, filtered by bandlimited filters to the K lowest eigenvectors, can estimate the value of $\|\mathbf{\Delta}_m\|_2^2$. These filters can be implemented using the described graph shifts.

Having in mind that the set of vertices, $\mathcal{M}$, needs to be selected only once to sample all K-bandlimited signals on a graph $\mathcal{G}$, the problem of the *uniqueness of solution* does not exist in this sampling setup, in contrast to classical compressive sensing (meaning that in this case there

is no need for sophisticated tools like the restricted isometry property, since there is no possibility that two different sets of K nonzero elements, with different indices k, satisfy the same measurements). Namely, here we assume that the signal is bandlimited and with known indices of nonzero spectral elements, $X(k)$, that is, $X(k)$ may assume nonzero values only for $k = 0, 1, 2, \ldots, K-1$. Once optimal sampling vertices are selected, the problem reduces to the solution of

$$\mathbf{y} = \mathbf{A}_{MK}\mathbf{X}_K, \tag{4.14}$$

where the reduced measurement matrix, $\mathbf{A}_{MK}$, is well defined, without any random combination of indices, while the reduced GFT vector, $\mathbf{X}_K$, is already explained, with $\mathbf{y}$ containing the signal values at the selected vertices. Since $M \geq K$, the *stability of this system solution* is well studied within linear algebra, with the condition number being the best parameter for its description.

The reconstruction is performed using

$$\mathbf{X}_K = \text{pinv}(\mathbf{A}_{MK})\mathbf{y}$$

and

$$\mathbf{x} = \mathbf{U}_K\mathbf{X}_K.$$

If the measurements, $\mathbf{y}$, are noisy, then

$$\mathbf{y} + \boldsymbol{\varepsilon} = \mathbf{A}_{MK}(\mathbf{X}_K + \mathbf{X}_N),$$

where $\boldsymbol{\varepsilon}$ denotes additive noise, and $\mathbf{X}_K + \mathbf{X}_N$ is the reconstructed vector which contains the true elements, $\mathbf{X}_K$, and the noise in the reconstruction, $\mathbf{X}_N$. Then, the output signal-to-noise ratio is defined by the condition number (ratio of the maximum and the minimum eigenvalue, $d_{\max}$ and $d_{\min}$, of the matrix $\mathbf{A}_{MK}^T\mathbf{A}_{MK}$), to yield

$$-10\log\left(\frac{d_{\max}}{d_{\min}}\right) \leq \text{SNR}_i - \text{SNR} \leq 10\log\left(\frac{d_{\max}}{d_{\min}}\right),$$

where the input and output signal-to-noise ratios are defined as

$$\text{SNR}_i = 10\log(\|\mathbf{y}\|_2^2 \,/\, \|\boldsymbol{\varepsilon}\|_2^2)$$

and

$$\text{SNR} = 10\log(\|\mathbf{X}_K\|_2^2 \,/\, \|\mathbf{X}_N\|_2^2).$$

Here, we have assumed that the columns of the measurement matrix are energy normalized, for notation simplicity.

Other strategies for random sampling can be found, for example, in Chen *et al.* (2015b, 2016) and Tanaka and Eldar (2020).

5

Filter Bank on a Graph

Subsampling and upsampling are the two standard operators used to alter the scale at which the signal is processed. Subsampling of a signal by a factor of 2, followed by the corresponding upsampling, can be described in classical signal processing by

$$f(n) = \frac{1}{2}(x(n) + (-1)^n x(n)) = \frac{1}{2}((1 + (-1)^n)x(n)),$$

as illustrated in Figure 5.1.

This is the basic operation used in multiresolution approaches based on filter banks and can be extended to signals on graphs in the following way. Consider a graph with the set of vertices $\mathcal{V}$. Any set of vertices can be considered as a union of two disjoint subsets $\mathcal{E}$ and $\mathcal{H}$, such that $\mathcal{V} = \mathcal{E} \cup \mathcal{H}$ and $\mathcal{E} \cap \mathcal{H} = \emptyset$. The subsampling-upsampling procedure can then be performed in the following two steps:

1. Subsample the signal on a graph by keeping only signal values on the vertices $n \in \mathcal{E}$, while not altering the original graph topology.

2. Upsample the graph signal by setting the signal values for the vertices $n \notin \mathcal{E}$ to zero.

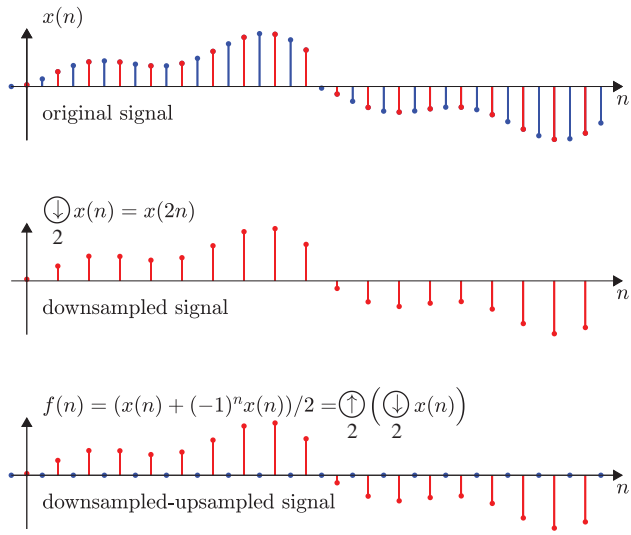


Figure 5.1: Principle of a signal, $x(n)$, downsampling and upsampling in the classical time domain.

This combined subsampling-upsampling operation produces a graph signal

$$f(n) = \frac{1}{2}(1 + (-1)^{\beta_{\mathcal{E}}(n)})x(n),$$

where

$$\beta_{\mathcal{E}}(n) = \begin{cases} 0, & \text{if } n \in \mathcal{E} \\ 1, & \text{if } n \in \mathcal{H}. \end{cases}$$

The values of the resulting graph signal, $f(n)$, are therefore $f(n) = x(n)$ if $n \in \mathcal{E}$ and $f(n) = 0$ elsewhere.

The vector form of the subsampled-upsampled graph signal, $f(n)$, which comprises all $n \in \mathcal{V}$, is given by

$$\mathbf{f} = \frac{1}{2}(\mathbf{x} + \mathbf{J}_{\mathcal{E}}\mathbf{x}) = \frac{1}{2}(\mathbf{I} + \mathbf{J}_{\mathcal{E}})\mathbf{x}, \tag{5.1}$$

where $\mathbf{J}_{\mathcal{E}} = \text{diag}((-1)^{\beta_{\mathcal{E}}(n)}),\ n \in \mathcal{V}$.

The focus of our analysis will be on the two-channel wavelet filter bank on a graph, shown in Figure 5.2. As in the classical wavelet analysis framework for the time domain signals, such a filter bank

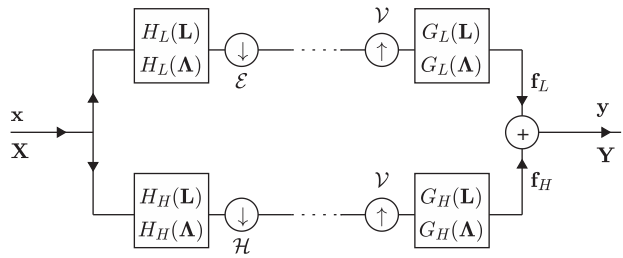


Figure 5.2: Principle of a filter bank for a graph signal.

provides decomposition of a graph signal into the corresponding low-pass (smooth) and high-pass (fast-varying) constituents. The analysis side (left part of the system in Figure 5.2) consists of two channels with filters characterized by the vertex domain operators $H_L(\mathbf{L})$ and $H_H(\mathbf{L})$, with the corresponding spectral domain operators $H_L(\mathbf{\Lambda})$ and $H_H(\mathbf{\Lambda})$. The operator $H_L(\mathbf{L})$ acts as a low-pass filter, transferring the low-pass components of the graph signal, while the operator $H_H(\mathbf{L})$ does the opposite, acting as a high-pass filter. The low-pass filter, $H_L(\mathbf{H})$, is followed by a downsampling operator which keeps only the graph signal values, $\mathbf{x}$, at the vertices $n \in \mathcal{E}$. Similarly, the high-pass filtering with the operator $H_H(\mathbf{L})$, is subsequently followed by a downsampling to the vertices $n \in \mathcal{H}$. These operations are crucial to alter the scale at which the graph signal is processed.

The synthesis side (right part in Figure 5.2), comprises the complementary upsampling and filtering operations, aiming to perform the graph signal reconstruction based on the upsampled versions, $\frac{1}{2}(\mathbf{I} + \mathbf{J}_{\mathcal{E}})H_L(\mathbf{L})\mathbf{x}$ and $\frac{1}{2}(\mathbf{I} + \mathbf{J}_{\mathcal{H}})H_H(\mathbf{L})\mathbf{x}$, of signals obtained on the filter bank analysis side. Therefore, upon performing the upsampling of these signals onto the original set of vertices, $\mathcal{V}$, by adding zeros to the complementary sets of vertices, filtering is performed by adequate low-pass, $G_L(\mathbf{L})$, and high-pass, $G_H(\mathbf{L})$, filters, to replace the zeros with meaningful values, as required for a successful reconstruction of the original signal. As in the classical wavelet analysis, to achieve the perfect (distortion-free) reconstruction it is necessary to conveniently design the analysis filters, $H_L(\mathbf{L})$ and $H_H(\mathbf{L})$, and the synthesis filters,

$G_L(\mathbf{L})$ and $G_H(\mathbf{L})$, as well as to determine adequate downsampling and upsampling operators.

It will be shown that the spectral folding phenomenon, described by Equations (3.9)–(3.10) in Part I, characterized by the specific spectral symmetry in the case of bipartite graphs, can be used to form the basis for the two-channel filter bank framework discussed in this section.

Consider a graph signal, $\mathbf{x}$, and the filter-bank as in Figure 5.2. If the graph signal, $\mathbf{x}$, passes through a low-pass analysis filter, $H_L(\mathbf{L})$, the output signal is $H_L(\mathbf{L})\mathbf{x}$. According to (5.1), the downsampled–upsampled form of the output signal, $H_L(\mathbf{L})\mathbf{x}$, is given by $\frac{1}{2}(\mathbf{I}+\mathbf{J}_{\mathcal{E}})H_L(\mathbf{L})\mathbf{x}$. After the syntheses filter, $G_L(\mathbf{L})$, the graph signal output becomes

$$\mathbf{f}_L = \frac{1}{2}G_L(\mathbf{L})(\mathbf{I}+\mathbf{J}_{\mathcal{E}})H_L(\mathbf{L})\mathbf{x}. \tag{5.2}$$

The same holds for the high-pass part

$$\mathbf{f}_H = \frac{1}{2}G_H(\mathbf{L})(\mathbf{I}+\mathbf{J}_{\mathcal{H}})H_H(\mathbf{L})\mathbf{x}, \tag{5.3}$$

where $\mathbf{J}_{\mathcal{H}} = -\mathbf{J}_{\mathcal{E}} = \text{diag}((-1)^{1-\beta_{\mathcal{E}}(n)})$ and

$$\mathbf{J}_{\mathcal{H}} + \mathbf{J}_{\mathcal{E}} = \mathbf{0}. \tag{5.4}$$

The overall output is a sum of these two signals, as illustrated in Figure 5.2, which after rearranging of terms gives

$$\begin{aligned}\mathbf{y} = \mathbf{f}_L + \mathbf{f}_H &= \frac{1}{2}(G_L(\mathbf{L})H_L(\mathbf{L}) + G_H(\mathbf{L})H_H(\mathbf{L}))\mathbf{x} \\ &+ \frac{1}{2}(G_L(\mathbf{L})\mathbf{J}_{\mathcal{E}}H_L(\mathbf{L}) + G_H(\mathbf{L})\mathbf{J}_{\mathcal{H}}H_H(\mathbf{L}))\mathbf{x}.\end{aligned} \tag{5.5}$$

The perfect reconstruction condition, $\mathbf{y} = \mathbf{x}$, is then achieved if

$$G_L(\mathbf{L})H_L(\mathbf{L}) + G_H(\mathbf{L})H_H(\mathbf{L}) = 2\mathbf{I}, \tag{5.6}$$

$$G_L(\mathbf{L})\mathbf{J}_{\mathcal{E}}H_L(\mathbf{L}) - G_H(\mathbf{L})\mathbf{J}_{\mathcal{E}}H_H(\mathbf{L}) = \mathbf{0}. \tag{5.7}$$

Spectral solution. For the spectral representation of the filter-bank signals in the domain of Laplacian basis functions, we will use the decomposition of the graph Laplacian in the form

$$\mathbf{F} = \mathbf{U}^T\mathbf{f} = \frac{1}{2}(\mathbf{U}^T\mathbf{x} + \mathbf{U}^T\mathbf{J}_{\mathcal{E}}\mathbf{x}) = \frac{1}{2}(\mathbf{X} + \mathbf{X}^{(alias)}), \tag{5.8}$$

where $\mathbf{X}^{(alias)} = \mathbf{U}^T\mathbf{J}_{\mathcal{E}}\mathbf{x}$ is the aliasing spectral component.

In the case of *bipartite graphs*, the matrix operator $\mathbf{U}^T\mathbf{J}_{\mathcal{E}}$ produces the transformation matrix $\mathbf{U}^T$ with reversed (left–right flipped) order of eigenvectors. This is obvious from (3.10) in Part I, since

$$\begin{aligned}\mathbf{U}^T\mathbf{J}_{\mathcal{E}} &= [\mathbf{u}_0 \;\; \mathbf{u}_1 \;\; \ldots \;\; \mathbf{u}_{N-1}]^T\mathbf{J}_{\mathcal{E}} \\ &= \begin{bmatrix} \mathbf{u}_{0\mathcal{E}} & \mathbf{u}_{1\mathcal{E}} & & \mathbf{u}_{N-1\mathcal{E}} \\ -\mathbf{u}_{0\mathcal{H}} & -\mathbf{u}_{1\mathcal{H}} & \cdots & -\mathbf{u}_{N-1\mathcal{H}} \end{bmatrix}^T \\ &= [\mathbf{u}_{N-1} \; \mathbf{u}_{N-2} \;\; \ldots \;\; \mathbf{u}_0]^T = \mathbf{U}_{\mathrm{LR}}^T\end{aligned}$$

where

$$\mathbf{u}_k = \begin{bmatrix} \mathbf{u}_{k\mathcal{E}} \\ \mathbf{u}_{k\mathcal{H}} \end{bmatrix}, \quad \mathbf{u}_{N-1-k} = \begin{bmatrix} \mathbf{u}_{k\mathcal{E}} \\ -\mathbf{u}_{k\mathcal{H}} \end{bmatrix}, \quad k = 0, 1, \ldots N-1,$$

and

$$\mathbf{U}_{\mathrm{LR}} = [\mathbf{u}_{N-1} \;\; \mathbf{u}_{N-2} \;\; \ldots \;\; \mathbf{u}_0]$$

is a left–right flipped version of the eigenvector matrix

$$\mathbf{U} = [\mathbf{u}_0 \;\; \mathbf{u}_1 \;\; \ldots \;\; \mathbf{u}_{N-1}].$$

The element-wise form of Equation (5.8) is given by

$$F(k) = \frac{1}{2}(X(k) + X(N-1-k)).$$

For *bipartite graphs and the normalized graph Laplacian*, we can write

$$F(\lambda_k) = \frac{1}{2}(X(\lambda_k) + X(2-\lambda_k)).$$

The second term in $F(\lambda_k)$ represents an aliasing component of the GFT of the original signal.

The spectral representation of (5.6) is obtained with a left-multiplication by $\mathbf{U}^T$ and a right-multiplication by $\mathbf{U}$,

$$\mathbf{U}^T G_L(\mathbf{L})\mathbf{U}\mathbf{U}^T H_L(\mathbf{L})\mathbf{U} + \mathbf{U}^T G_H(\mathbf{L})\mathbf{U}\mathbf{U}^T H_H(\mathbf{L})\mathbf{U} = 2\mathbf{I},$$

having in mind that we can add $\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}$ between $G_L(\mathbf{L})$ and $H_L(\mathbf{L})$, and between $G_H(\mathbf{L})$ and $H_H(\mathbf{L})$. Using the spectral domain definition of the transfer functions, $\mathbf{U}^T H_L(\mathbf{L})\mathbf{U} = H_L(\mathbf{\Lambda})$, we obtain

the spectral domain form of the reconstruction condition (5.6) as

$$G_L(\mathbf{\Lambda})H_L(\mathbf{\Lambda}) + G_H(\mathbf{\Lambda})H_H(\mathbf{\Lambda}) = 2\mathbf{I}. \tag{5.9}$$

For the aliasing part in Equation (5.7), the left-multiplication is performed by $\mathbf{U}^T$, while the right-multiplication is done by $\mathbf{U}_{\text{LR}}^T$. The first term in (5.7) is then of the form

$$\begin{aligned}\mathbf{U}^T G_L(\mathbf{L})\mathbf{U}\mathbf{U}^T\mathbf{J}_{\mathcal{E}}H_L(\mathbf{L})\mathbf{U}_{\text{LR}} &= \mathbf{U}^T G_L(\mathbf{L})\mathbf{U}\mathbf{U}_{\text{LR}}^T H_L(\mathbf{L})\mathbf{U}_{\text{LR}} \\ &= G_L(\mathbf{\Lambda})H_L^{(R)}(\mathbf{\Lambda}),\end{aligned} \tag{5.10}$$

since $\mathbf{U}^T\mathbf{J}_{\mathcal{E}} = \mathbf{U}_{\text{LR}}^T$ and $\mathbf{U}_{\text{LR}}^T\mathbf{U}_{\text{LR}} = \mathbf{I}$. The term

$$H_L^{(R)}(\mathbf{\Lambda}) = \mathbf{U}_{\text{LR}}^T H_L(\mathbf{L})\mathbf{U}_{\text{LR}} = H_L(2\mathbf{I} - \mathbf{\Lambda})$$

is just a reversed order version of the diagonal matrix $H_L(\mathbf{\Lambda})$, with diagonal elements $H_L(\lambda_{N-1-k}) = H_L(2 - \lambda_k)$ instead of $H_L(\lambda_k)$.

The same holds for the second term in (5.7) which is equal to $G_H(\mathbf{L})\mathbf{J}_{\mathcal{H}}H_H(\mathbf{L})$, yielding the final spectral form of the aliasing condition in (5.7) as

$$G_L(\mathbf{\Lambda})H_L(2\mathbf{I} - \mathbf{\Lambda}) - G_H(\mathbf{\Lambda})H_H(2\mathbf{I} - \mathbf{\Lambda}) = \mathbf{0}. \tag{5.11}$$

An element-wise solution to the system in (5.6)–(5.7), for bipartite graphs and the normalized graph Laplacian, according to (5.9) and (5.11), reduces to

$$G_L(\lambda_k)H_L(\lambda_k) + G_H(\lambda_k)H_H(\lambda_k) = 2, \tag{5.12}$$

$$G_L(\lambda_k)H_L(2 - \lambda_k) - G_H(\lambda_k)H_H(2 - \lambda_k) = 0. \tag{5.13}$$

Remark 26: *A quadratic mirror filter solution* would be such that for the designed transfer function of the low-pass analysis filter, $H_L(\lambda)$, the other filters are

$$\begin{aligned}G_L(\lambda) &= H_L(\lambda), \\ H_H(\lambda) &= H_L(2 - \lambda), \\ G_H(\lambda) &= H_H(\lambda) = H_L(2 - \lambda).\end{aligned} \tag{5.14}$$

For this solution, the design equation is given by

$$H_L^2(\lambda) + H_L^2(2 - \lambda) = 2, \tag{5.15}$$

while the aliasing cancellation condition, (5.13), is always satisfied.

An example of such a system would be an ideal low-pass filter, defined by $H_L(\lambda) = \sqrt{2}$ for $\lambda < 1$ and $H_L(\lambda) = 0$ elsewhere. Since $H_H(\lambda) = H_L(2-\lambda)$ holds for systems on bipartite graphs, this satisfies the reconstruction condition. For the vertex domain realization, an approximation of the ideal filter with a finite neighborhood filtering relation would be required.

Example 13: Consider a simple form of the low-pass system

$$H_L^2(\lambda) = 2 - \lambda,$$

which satisfies the design equation, $H_L^2(\lambda) + H_L^2(2-\lambda) = 2$. It also satisfies the condition that its form is of low-pass type for the normalized Laplacian of bipartite graphs, $H_L^2(\lambda_0) = 2 - \lambda_0 = 2$, since $\lambda_0 = 0$, and $H_L^2(\lambda_{\max}) = 2 - \lambda_{\max} = 0$, as $\lambda_{\max} = 2$. The vertex domain system operators which satisfy all four quadratic mirror analysis and synthesis filters in (5.14), are

$$H_L(\mathbf{\Lambda}) = \sqrt{\mathbf{2I} - \mathbf{\Lambda}}, \quad G_L(\mathbf{\Lambda}) = H_L(\mathbf{\Lambda}) = \sqrt{\mathbf{2I} - \mathbf{\Lambda}},$$
$$H_H(\mathbf{\Lambda}) = H_L(\mathbf{2I} - \mathbf{\Lambda}) = \sqrt{\mathbf{\Lambda}}, \quad G_H(\mathbf{\Lambda}) = H_H(\mathbf{\Lambda}) = \sqrt{\mathbf{\Lambda}}.$$

The spectral domain filtering form for the low-pass part of graph signal is then obtained from (5.2), as

$$\begin{aligned}\mathbf{F}_L &= \mathbf{U}^T\mathbf{f}_L = \frac{1}{2}\mathbf{U}^T G_L(\mathbf{L})(\mathbf{I} + \mathbf{J}_{\mathcal{E}})H_L(\mathbf{L})\mathbf{x} \\ &= \frac{1}{2}\mathbf{U}^T G_L(\mathbf{L})\mathbf{U}\mathbf{U}^T(\mathbf{I} + \mathbf{J}_{\mathcal{E}})H_L(\mathbf{L})\mathbf{U}_{\rm LR}\mathbf{U}_{\rm LR}^T\mathbf{U}\mathbf{X} \\ &= \frac{1}{2}G_L(\mathbf{\Lambda})H_L(\mathbf{\Lambda})\mathbf{X} + \frac{1}{2}G_L(\mathbf{\Lambda})H_L(2\mathbf{I} - \mathbf{\Lambda})\mathbf{X}_{\rm UD}\end{aligned}$$

since $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{U}_{\rm LR}^T\mathbf{U}_{\rm LR} = \mathbf{I}$, $\mathbf{U}^T\mathbf{J}_{\mathcal{E}} = \mathbf{U}_{\rm LR}^T$, $\mathbf{U}_{\rm LR}^T\mathbf{U} = \mathbf{I}_{\rm LR}$, and $\mathbf{I}_{\rm LR}\mathbf{X} = \mathbf{X}_{\rm UD}$, where $\mathbf{I}_{\rm LR}$ is an anti-diagonal (backward) identity matrix, and $\mathbf{X}_{\rm UD}$ is the GFT vector, $\mathbf{X}$, with elements flipped upside-down.

The same holds for the high-pass part in (5.3), to yield

$$\begin{aligned}\mathbf{F}_H &= \frac{1}{2}\mathbf{U}^T G_H(\mathbf{L})(\mathbf{I} + \mathbf{J}_{\mathcal{H}})H_H(\mathbf{L})\mathbf{x} \\ &= \frac{1}{2}G_H(\mathbf{\Lambda})H_H(\mathbf{\Lambda})\mathbf{X} - \frac{1}{2}G_H(\mathbf{\Lambda})H_H(2\mathbf{I} - \mathbf{\Lambda})\mathbf{X}_{\rm UD}\end{aligned}$$

$$
\mathbf{L} = \begin{array}{c} \\ 0 \\ 2 \\ 4 \\ 6 \\ 8 \\ 10 \\ 12 \\ 14 \\ 1 \\ 3 \\ 5 \\ 7 \\ 9 \\ 11 \\ 13 \\ 15 \end{array}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\tag{5.16}
$$

and

$$\mathbf{F}_L + \mathbf{F}_H = \mathbf{X}.$$

Therefore, after the one-step filter-bank based decomposition on a bipartite graph, we have a new low-pass signal, $\mathbf{f}_L$, for which the nonzero values are at the vertices in $\mathcal{E}$, and a high-pass signal, $\mathbf{f}_H$, with nonzero values only on $\mathcal{H}$. Note that the high-pass operator on the graph signal is the graph Laplacian, $\mathbf{L}$, while the low-pass operator is $\mathbf{2I} - \mathbf{L}$, which easily reduces to $|\mathbf{L}|$, for the normalized graph Laplacian used here.

Another simple transfer function that satisfies the design equation (5.15) is $H_L(\lambda) = \sqrt{2}\cos(\pi\lambda/4)$. A similar analysis can also be done for this transfer function and other functions defined by (5.14).

The considered transfer functions $H_L(\lambda) = \sqrt{2-\lambda}$ and $H_L(\lambda) = \sqrt{2}\cos(\pi\lambda/4)$ have several disadvantages, the most important being that they are not sufficiently smooth in the spectral domain at the boundary interval points (Stanković, 2015). In addition, although the

graph Laplacian, $\mathbf{L}$, is commonly sparse (with a small number of nonzero elements in large graphs), the transfer function form $H_L(\mathbf{L}) = \sqrt{2\mathbf{I} - \mathbf{L}}$ is not sparse. This is the reason to use other forms which are sufficiently smooth toward the boundary points, along with their polynomial approximations, $H_L(\mathbf{\Lambda}) = c_0\mathbf{\Lambda} + c_1\mathbf{\Lambda}^2 + \cdots + c_{M-1}\mathbf{\Lambda}^{M-1}$, with the coefficients $c_0, c_1, \ldots, c_{M-1}$, that approximate $H_L(\lambda)$ and $H_H(\lambda) = H_L(2-\lambda)$ for each $\lambda = \lambda_k$, $k = 0, 1, \ldots, N-1$. This topic will be addressed in detail on a general form of graphs in Section 8.

The classic time-domain Haar wavelet (and scale) functions are easily obtained for a bipartite graph, such that $\mathcal{E} = 0, 2, 4, \ldots, N-2$ and $\mathcal{H} = 1, 3, 5, \ldots, N-1$, with the adjacency/weighting matrix defined by the elements $A_{mn} = 1$, for $(m, n) \in \{(0,1), (2,3), \ldots, (N-2, N-1)\}$, as shown in Figure 5.3(a). This adjacency matrix has the block form as in Equation (2.19), Part I. The corresponding graph Laplacian is given in (5.16). Its eigenvectors are equal to the wavelet transform functions. The bipartite graph for the four-vertex resolution level in the Haar wavelet transform is shown in Figure 5.3(b).

Synthesis operators, comprised of more general interpolation methods, may be found in Li *et al.* (2019).

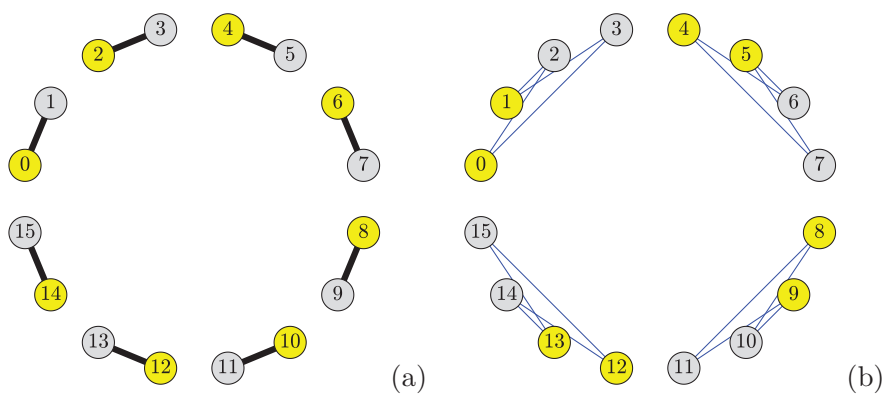


Figure 5.3: Bipartite graph for the Haar wavelet transform with $N = 16$ vertices. (a) Vertices in yellow are used for the low-pass part of the signal and correspond to the set $\mathcal{E}$, while the vertices in gray belong to the set $\mathcal{H}$. This is the highest two-vertex resolution level for the Haar wavelet. (b) Graph for a four-vertex resolution level in the Haar wavelet.

6

Time-Varying Signals on Graphs

We shall denote a time-varying signal by $x_p(n)$, where n designates the vertex index and p the discrete-time index. For uniform sampling in time, the index p corresponds to the time instant $p\Delta t$, where Δt is the sampling interval. In general, this type of data can be considered within the graph Cartesian product framework (given in Property M_{15}, Section 2.3, Part I). The resulting graph $\mathcal{G} = (\mathcal{V}, \mathcal{B})$ follows as a Cartesian product of the given graph $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{B}_1)$ and a simple path (or circular) graph $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{B}_2)$ that corresponds to the classical uniformly sampled time-domain axis.

Example 14: A graph topology for a time varying signal on a graph is shown in Part I, Figure 2.9, where the graph vertices are designated by $1, 2, 3, 4, 5$ and time instants are denoted as the a, b, c vertices on the path graph. The resulting Cartesian product graph, for the analysis of this kind of signals, is shown in Part I, Figure 2.9.

The adjacency matrix of a Cartesian product of two graphs is then given by

$$\mathbf{A} = \mathbf{A}_1 \otimes \mathbf{I}_{N_2} + \mathbf{I}_{N_1} \otimes \mathbf{A}_2 = \mathbf{A}_1 \oplus \mathbf{A}_2,$$

where $\mathbf{A}_1$ is the adjacency matrix of the graph of interest $\mathcal{G}_1$, and $\mathbf{A}_2$ is the adjacency matrix for the path or circular graph, $\mathcal{G}_2$, which designates

the sampling grid, while N_1 and N_2 denote, respectively, the number of vertices in $\mathcal{G}_1$ and $\mathcal{G}_2$.

We will next consider a simple and important example of a time-varying signal defined on graph in an iterative way, which designates the *diffusion process on a graph* in time.

6.1 Diffusion on Graph and Low Pass Filtering

Consider the diffusion equation

$$\partial \mathbf{x}/\partial t = -\alpha \mathbf{L}\mathbf{x}.$$

Its discrete-time form, at a time instant p, may be obtained by using the backward difference approximation of the partial derivative ($\partial \mathbf{x}/\partial t \sim \mathbf{x}_{p+1} - \mathbf{x}_p$), and has the form

$$\mathbf{x}_{p+1} - \mathbf{x}_p = -\alpha \mathbf{L}\mathbf{x}_{p+1}$$

or $\mathbf{x}_{p+1}(\mathbf{I} + \alpha \mathbf{L}) = \mathbf{x}_p$ to produce

$$\mathbf{x}_{p+1} = (\mathbf{I} + \alpha \mathbf{L})^{-1}\mathbf{x}_p.$$

On the other hand, the forward difference approximation ($\partial \mathbf{x}/\partial t \sim \mathbf{x}_p - \mathbf{x}_{p-1}$) to the diffusion equation yields

$$\mathbf{x}_{p+1} - \mathbf{x}_p = -\alpha \mathbf{L}\mathbf{x}_p$$

or

$$\mathbf{x}_{p+1} = (\mathbf{I} - \alpha \mathbf{L})\mathbf{x}_p.$$

It is interesting to note that these iterative forms lead to the *minimization of the quadratic form of a graph signal*, $E_x = \mathbf{x}\mathbf{L}\mathbf{x}^T$, (see Section 4.2, Part I). The minimum of this quadratic form can be found based on the steepest descent method, whereby the signal value at a time instant p is moving in the direction opposite to the gradient, toward the energy minimum position, with a step α. The gradient of the quadratic form, $E_x = \mathbf{x}\mathbf{L}\mathbf{x}^T$, is $\partial E_x/\partial \mathbf{x}^T = 2\mathbf{x}\mathbf{L}$, which results in an iterative procedure

$$\mathbf{x}_{p+1} = \mathbf{x}_p - \alpha \mathbf{L}\mathbf{x}_p = (\mathbf{I} - \alpha \mathbf{L})\mathbf{x}_p. \tag{6.1}$$

This relation can be used for simple and efficient filtering of graph signals (with the aim to minimize E_x as a measure of signal smoothness). The spectral domain relation follows immediately, and has the form

$$\mathbf{X}_{p+1} = (\mathbf{I} - \alpha\mathbf{\Lambda})\mathbf{X}_p$$

or for every individual component

$$X_{p+1}(k) = (1 - \alpha\lambda_k)X_p(k).$$

Recall that the eigenvalues, λ_k, represent the index of smoothness for a spectral vector (eigenvector), $\mathbf{u}_k$, with a small λ_k indicating smooth slow-varying elements of the eigenvectors; therefore, for low-pass filtering we should retain the slow-varying eigenvectors in a spectral representation of the graph signal. Obviously, these slow-varying components will pass through this system since $(1 - \alpha\lambda_k)$ is close to 1 for small λ_k, while the fast-varying components with a larger λ_k, are attenuated. This iterative procedure will converge if $|1 - \alpha\lambda_{\max}| < 1$.

In a stationary state of a diffusion process, the trivial minimal energy solution is obtained when

$$\lim_{p\to\infty} X_{p+1}(k) = (1 - \alpha\lambda_k)^{p+1} X_0(k),$$

that is, all components $X_{p+1}(k)$ tend to 0, except for the constant component, $X_{p+1}(0)$, for which $\lambda_0 = 0$. This component therefore defines the stationary state (maximally smooth solution). In order to avoid this effect in the processing of data on graphs, and to retain several low-pass components (eigenvectors) in the signal, the iteration process in (6.1) can be used in alternation with

$$\mathbf{x}_{p+2} = (\mathbf{I} + \beta\mathbf{L})\mathbf{x}_{p+1}. \tag{6.2}$$

This is the basis for Taubin's $\alpha - \beta$ algorithm, presented next.

6.2 Taubin's $\alpha - \beta$ Algorithm

When the two iterative processes in (6.1) and (6.2) are used in a successive order, the resulting system on a graph is referred to as Taubin's $\alpha - \beta$ algorithm. This algorithm is widely used for low-pass

filtering of data on graphs, since it is very simple, and admits efficient implementation in the vertex domain.

Definition: Taubin's $\alpha - \beta$ algorithm is a two-step iterative algorithm for efficient low-pass data filtering on graphs. Its two steps are defined in a unified way as

$$\mathbf{x}_{p+2} = (\mathbf{I} + \beta\mathbf{L})(\mathbf{I} - \alpha\mathbf{L})\mathbf{x}_p. \tag{6.3}$$

The corresponding element-wise transfer function in the spectral domain of the two iteration steps in (6.3) is given by

$$H(\lambda_k) = (1 + \beta\lambda_k)(1 - \alpha\lambda_k).$$

After K iterations of this algorithm, the spectral domain transfer function can be written as

$$H_K(\lambda_k) = ((1 + \beta\lambda_k)(1 - \alpha\lambda_k))^K. \tag{6.4}$$

For some values of $\alpha < \beta$, this system can be a good and computationally very simple approximation of a graph low-pass filter.

Example 15: Consider the graph from Figure 3.4(a) and its graph Laplacian, $\mathbf{L}$. For the choice of parameters $\alpha = 0.1798$ and $\beta = 0.2193$, the spectral transfer function in (6.4) is shown in Figure 6.1 for the considered graph filter, and for the numbers of iterations in Taubin's algorithm $K = 1, 5, 30$, and 150. Observe how the transfer function, $H(\lambda_k)$, approaches the ideal low-pass form as the number of iterations, K, increases.

The task is next to low-pass filter the noisy signal from Figure 3.12(b). The initial noisy signal is denoted by $\mathbf{x}_0$. Then $\mathbf{x}_1 = (\mathbf{I} - 0.1545\mathbf{L})\mathbf{x}_0$ is calculated using the corresponding graph Laplacian, followed by obtaining $\mathbf{x}_2 = (\mathbf{I} + 0.1875\mathbf{L})\mathbf{x}_1$. In the third and fourth iteration, the signal values $\mathbf{x}_3 = (\mathbf{I} - 0.1545\mathbf{L})\mathbf{x}_2$ and $\mathbf{x}_4 = (\mathbf{I} + 0.1875\mathbf{L})\mathbf{x}_3$ are calculated. This two-step iteration cycle is repeated $K = 20$ times. The resulting signal is the same as the output of an ideal low-pass filter shown in Figure 3.12(c).

Finally, the noisy signal from Figure 2.3 was filtered using Taubin's $\alpha - \beta$ algorithm, with $\alpha = 0.15$ and $\beta = 0.15$, over $K = 100$ iterations, and the result is shown in Figure 6.2. Observe the reduced level of additive noise in the output.

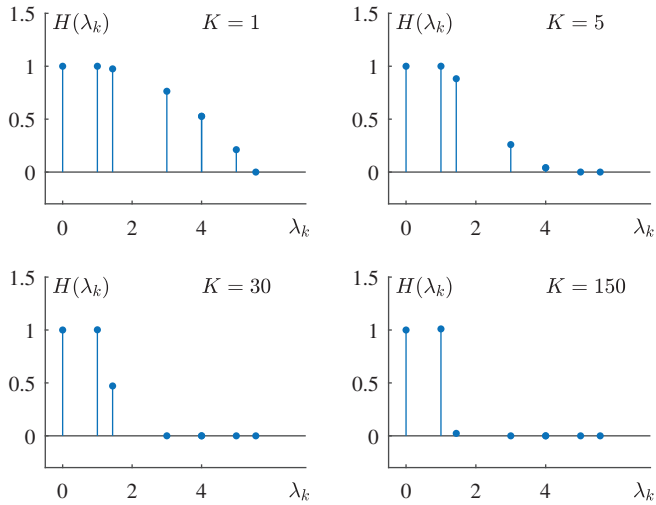


Figure 6.1: Filter approximation in the spectral domain for a varying number of iterations, K, using Taubin's algorithm and the graph Laplacian matrix of the graph in Figure 3.4.

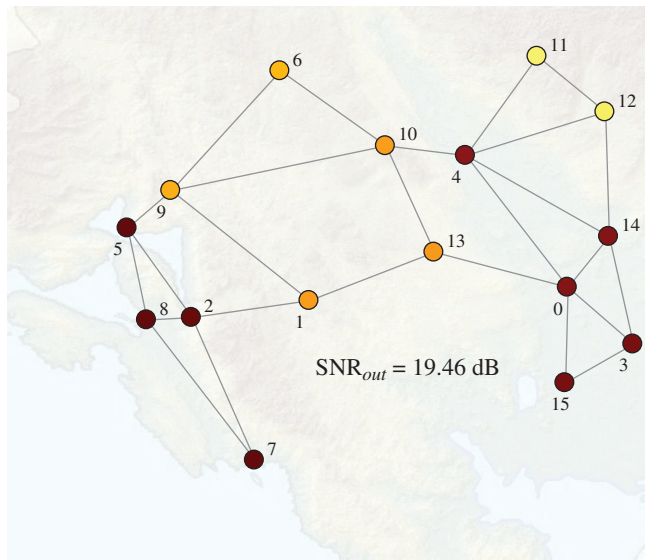


Figure 6.2: The noisy signal from Figure 2.3 was filtered using $K = 100$ iterations of the Taubin two-step algorithm with $\alpha = 0.15$ and $\beta = 0.15$.

Processing of time-varying signals on graphs has been a topic of intensive research; for a deeper insight we refer the reader to Isufi *et al.* (2017), Grassi *et al.* (2017), and Gama *et al.* (2019).

7

Random Graph Signal Processing

This section extends the concepts of data analytics for deterministic signals on graphs addressed so far, to introduce notions of random signals on graphs, their properties, and statistical graph-specific methods for their analysis. The main focus is on wide-sense stationary (WSS) data observed on graphs. In general, the stationarity of a signal is inherently related to the signal shift operator and its properties. We have already presented two approaches to define a shift on a graph (through the adjacency matrix and the graph Laplacian, and their spectral decompositions). These will be used, along with other general properties of WSS signals, to define the conditions for wide sense stationarity of random signals on graphs (Chepuri and Leus, 2016; Loukas and Perraudin, 2016; Marques *et al.*, 2017; Perraudin and Vandergheynst, 2017; Puy *et al.*, 2018; Zhang *et al.*, 2015). However the main obstacle toward extending the classical statistical data analytics to graphs is that the shift on a graph typically does not preserve signal energy (isometry property), that is, $\|\mathbf{Ax}\|_2^2 \neq \|\mathbf{x}\|_2^2$.

For completeness, we first provide a short review of WSS definitions in classical signal processing, together with their properties.

7.1 Review of WSS and Related Properties for Random Signals in Standard Time Domain

Definition: A real-valued random signal, $x(n)$, is WSS in the standard time domain if its mean value is time-invariant, $\mu_x(n) = \mathrm{E}\{x(n)\} = \mu_x$, and its autocorrelation function is shift-invariant, that is, $r_x(n, n-m) = \mathrm{E}\{x(n)x(n-m)\} = r_x(m)$.

Remark 27: A random WSS time-domain signal, $x(n)$, can be considered as an output of a linear shift invariant system with impulse response, $h(n)$, which is driven by a white noise input, $\varepsilon(n)$, with $r_\varepsilon(n, m) = \delta(n-m)$.

Remark 28: In classical time domain, the eigenvectors, $\mathbf{u}_k$, of the shift operator $y(n) = x(n-1)$, or in a matrix form $\mathbf{y} = \mathbf{A}\mathbf{x}$, are the DFT basis functions, with $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H$. This property is discussed in detail and proven in Part I, Section 3.2, Equations (3.4)–(3.5).

Remark 29: For a random signal, its DFT $\mathbf{X} = \mathbf{U}^H\mathbf{x}$ is also a random signal with the power spectrum matrix $\mathbf{P}_x = \mathrm{E}\{\mathbf{X}\mathbf{X}^H\}$, where $\mathbf{U}^H$ is the DFT transformation matrix. For WSS signals, the matrix $\mathbf{P}_x$ is diagonal and has the power spectral density (PSD) as its diagonal values

$$p_x(k) = \mathrm{DFT}\{r_x(n)\} = \mathrm{E}\{|X(k)|^2\}.$$

Remark 30: For WSS random signals, their correlation matrix, $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\}$, is diagonalizable with the same transform matrix, $\mathbf{U}$, which defines the DFT, $\mathbf{X} \overset{def}{=} \mathbf{U}^H\mathbf{x}$, with $\mathbf{x} \overset{def}{=} \mathbf{U}\mathbf{X}$. The proof follows from

$$\begin{aligned} \mathbf{R}_x &= \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} = \mathrm{E}\{\mathbf{U}\mathbf{X}(\mathbf{U}\mathbf{X})^H\} \\ &= \mathbf{U}\mathrm{E}\{\mathbf{X}\mathbf{X}^H\}\mathbf{U}^H = \mathbf{U}\mathbf{P}_x\mathbf{U}^H, \end{aligned} \tag{7.1}$$

and the fact that $\mathbf{P}_x$ is a diagonal matrix for WSS signals.

The properties of the WSS signals in classical analyses, presented in this subsection, will be used next to define the corresponding properties of *random signals on undirected graphs.*

7.2 Adjacency Matrix Based Definition of GWSS

Consider a real-valued white noise signal on a graph, $\boldsymbol{\varepsilon} = \{\varepsilon(n)\}$. Following Remark 27, a signal $\mathbf{x}$ on the graph is graph wide-sense stationary (GWSS) if it can be considered as an output of a linear shift invariant system on a graph, $H(\mathbf{A}) = \sum_{m=0}^{M-1} h_m \mathbf{A}^m$, which is driven by a white noise input, $\boldsymbol{\varepsilon}$, that is

$$\mathbf{x} = H(\mathbf{A})\boldsymbol{\varepsilon}.$$

Remark 31: The autocorrelation matrix, $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\}$, of a GWSS signal is diagonalizable with the eigenmatrix of the adjacency matrix, $\mathbf{A}$, since (*cf.* Remark 30)

$$\begin{aligned} \mathbf{A} &= \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T \\ \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} &= \mathbf{U}\mathbf{P}_x\mathbf{U}^T, \end{aligned} \tag{7.2}$$

where $\mathbf{P}_x$ is a diagonal matrix. The values on the diagonal of matrix $\mathbf{P}_x$ can be comprised into the vector $\mathbf{p}_x$, which represents the PSD of a graph signal, $\mathbf{x}$, $p_x(k) = \mathrm{E}\{|X(k)|^2\}$.

To prove this property for a signal $\mathbf{x} = H(\mathbf{A})\boldsymbol{\varepsilon}$, consider

$$\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} = \mathrm{E}\{H(\mathbf{A})\boldsymbol{\varepsilon}(H(\mathbf{A})\boldsymbol{\varepsilon})^T\} = H(\mathbf{A})H^T(\mathbf{A}),$$

since $\mathrm{E}\{\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T\} = \mathbf{I}$. Using $H(\mathbf{A}) = \mathbf{U}^T H(\boldsymbol{\Lambda})\mathbf{U}$, we obtain

$$\mathbf{R}_x = \mathbf{U}^T|H(\boldsymbol{\Lambda})|^2\mathbf{U},$$

which concludes the proof that the matrix $\mathbf{P}_x$ is diagonal

$$\mathbf{P}_x = |H(\boldsymbol{\Lambda})|^2,$$

with the diagonal elements equal to the PSD of signal $\mathbf{x}$,

$$p_x(k) = |H(\lambda_k)|^2.$$

The periodogram of a graph signal can be estimated using K realizations of the random signal, denoted by $\mathbf{x}_i$, and is equal to the diagonal elements of the matrix

$$\hat{\mathbf{P}}_x = \frac{1}{K}\sum_{i=1}^{K}\mathbf{X}_i\mathbf{X}_i^T = \mathbf{U}^T\frac{1}{K}\sum_{i=1}^{K}(\mathbf{x}_i\mathbf{x}_i^T)\mathbf{U}.$$

Consider a system on a graph, with a spectral domain transfer function $H(\mathbf{\Lambda})$. Assume that the input signal to this system is GWSS, with PSD $p_x(k)$. The PSD of the output graph signal, $y(n)$, is then given by

$$p_y(k) = |H(\lambda_k)|^2 p_x(k).$$

This expression is conformal with the output power of a standard linear system.

7.3 Wiener Filter on a Graph

Consider a real-valued graph signal, $\mathbf{s}$, which serves as an input to a linear shift-invariant system on an undirected graph, to yield a noisy output

$$\mathbf{x} = \sum_{m=0}^{M-1} h_m \mathbf{A}^m \mathbf{s} + \boldsymbol{\varepsilon}.$$

In the spectral domain, this system is described by

$$\mathbf{X} = H(\mathbf{\Lambda})\mathbf{S} + \mathbf{E},$$

where $\mathbf{E}$ is the GFT of the noise, $\boldsymbol{\varepsilon}$.

Assume that the signal and noise are statistically independent, and that the noise is a zero-mean GWSS random signal. The aim is to find the system function of the optimal filter, $G(\mathbf{\Lambda})$, such that its output $\mathbf{Y} = G(\mathbf{\Lambda})\mathbf{X}$, estimates the GFT of the input, $\mathbf{S}$, in the least squares sense. This condition can be expressed as

$$e^2 = \mathrm{E}\{\|\mathbf{S} - \mathbf{Y}\|_2^2\} = \mathrm{E}\{\|\mathbf{S} - G(\mathbf{\Lambda})\mathbf{X}\|_2^2\}.$$

Upon setting the derivative of e^2 with respect to the elements of $G(\mathbf{\Lambda})$ to zero, we arrive at

$$2\mathrm{E}\{(\mathbf{S} - G(\mathbf{\Lambda})\mathbf{X})\mathbf{X}^T\} = \mathbf{0},$$

which results in the system function of the graph Wiener filter in the form (using matrix division in a symbolic way)

$$\begin{aligned} G(\mathbf{\Lambda}) &= \frac{\mathrm{E}\{\mathbf{S}\mathbf{X}^T\}}{\mathrm{E}\{\mathbf{X}\mathbf{X}^T\}} = \frac{\mathrm{E}\{\mathbf{S}(H(\mathbf{\Lambda})\mathbf{S} + \mathbf{E})^T\}}{\mathrm{E}\{(H(\mathbf{\Lambda})\mathbf{S} + \mathbf{E})(H(\mathbf{\Lambda})\mathbf{S} + \mathbf{E})^T\}} \\ &= \frac{H(\mathbf{\Lambda})\mathbf{P}_s}{H^2(\mathbf{\Lambda})\mathbf{P}_s + \mathbf{P}_\varepsilon} \end{aligned}$$

or element-wise

$$G(\lambda_k) = \frac{H(\lambda_k)p_s(k)}{H^2(\lambda_k)p_s(k) + E(k)}.$$

When the noise is not present, the elements of the vector $\mathbf{E}$ are zero-valued, $E(k) = 0$ for all k, and the graph inverse filter (introduced in Section 3.5.1) directly follows.

Remark 32: The above expressions for the graph Wiener filter are conformal with the standard frequency domain Wiener filter, given by

$$G(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_\varepsilon(\omega)},$$

which again demonstrates the generic nature of Graph Data Analytics.

7.4 Spectral Domain Shift Based Definition of GWSS

Consider an m-step shift on a graph defined using the graph filter response

$$\mathcal{T}_m\{h(n)\} = h_m(n) = \sum_{k=0}^{N-1} H(\lambda_k)u_k(m)u_k(n). \tag{7.3}$$

The matrix form of this relation is given by

$$\mathcal{T}_h = H(\mathbf{L}) = \sum_{m=0}^{M-1} h_m\mathbf{L}^m = \mathbf{U}H(\mathbf{\Lambda})\mathbf{U}^T, \tag{7.4}$$

where $\mathcal{T}_m\{h(n)\}$ are the elements of $\mathcal{T}_h$.

Note that the graph filter response function is well localized on a graph. Namely, if we use, for example, the $(M-1)$-neighborhood of a vertex n, within a filtering function of order M defined by $H(\mathbf{\Lambda})$, then only the vertices within this neighborhood are used in the calculation of graph filter response. From (7.4), we see that the localization operator acts in the spectral domain and associates the corresponding shift to the vertex domain.

Definition: A random graph signal, $x(n)$, is GWSS if its autocorrelation function is invariant with respect to the shift, $\mathcal{T}_m\{r_x(n)\}$.

Similar to (7.2), the autocorrelation matrix, $\mathbf{R}_x$, of a GWSS signal is diagonalizable based on the matrix of eigenvectors of the graph Laplacian $\mathbf{L}$, whereby

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T. \tag{7.5}$$

For the basic autocorrelation we use

$$\mathbf{R}_x = \mathbf{U}P_x(\mathbf{\Lambda})\mathbf{U}^T$$

so that

$$\mathcal{T}_m\{r_x(n)\} = \sum_{k=0}^{N-1} p_x(\lambda_k)u_k(m)u_k(n)$$

where

$$P_x(\mathbf{\Lambda}) = \mathbf{U}\mathbf{R}_x\mathbf{U}^T$$

is a diagonal matrix.

7.5 Isometric Shift Operator

Another possible approach may be based on the shift operator defined as $\mathcal{T}_h = \exp(j\pi\sqrt{\mathbf{L}/\rho})$, where ρ is the upper bound on the eigenvalues, $\rho = \max_k\{\lambda_k\}$ (Girault, 2015; Girault *et al.*, 2015). Physically, this operator casts the eigenvalues of the Laplacian, $\mathbf{L}$, onto a unit circle, thus preserving in this way the isometry property, since

$$\mathcal{T}_h = \exp(j\pi\sqrt{\mathbf{L}/\rho}) = \mathbf{U}\exp(j\pi\sqrt{\mathbf{\Lambda}/\rho})\mathbf{U}^T. \tag{7.6}$$

The property $f(\mathbf{L}) = \mathbf{U}f(\mathbf{\Lambda})\mathbf{U}^H$ was used above. Observe that for real-valued eigenvalues, λ_k, all eigenvalues of the matrix $\exp(j\pi\sqrt{\mathbf{\Lambda}/\rho})$ reside on the unit circle, with the frequency $0 \leq \omega_k = \pi\sqrt{\lambda_k/\rho} \leq \pi$ being associated to the eigenvector $\mathbf{u}_k$.

8

Vertex-Frequency Representations

Oftentimes in practical applications concerned with large graphs, we may not be interested in the analysis of the entire graph signal, but rather in its local behavior. Indeed, the Big Data paradigm has revealed the possibility of using smaller and localized subsets of the available information to enable reliable mathematical analysis and local characterization of subsets of data of interest (Sandryhaila and Moura, 2014a). Our aim in this section is to characterize the localized graph signal behavior simultaneously in the vertex-frequency domain, in a natural analogy with classical time-frequency analysis (Boashash, 2015; Cohen, 1995; Stanković *et al.*, 2014). Indeed, the concept of vertex-frequency analysis was introduced in Shuman *et al.* (2012), by extending the principle of signal localization by applying localization window functions to signals defined on graphs. This concept was further developed in Shuman *et al.* (2016), with the extensions of this approach including the multi-window form Zheng *et al.* (2016), a short-graph Fourier transform combined with page-rank vectors (Tepper and Sapiro, 2016), or vertex domain localization windows (Stanković *et al.*, 2017). Window forms have also been adapted to define the frequency-varying localized graph Fourier transform (Cioacă *et al.*, 2019) and spectral domain wavelet

transform-based vertex-frequency kernels, including the signal adaptive kernels with polynomial approximations and recursive realizations (Behjat and Van De Ville, 2019; Hammond *et al.*, 2011, 2019).

It is important to note that, while the concept of window functions for signal localization has been extended to signals defined on graphs (Shuman *et al.*, 2012, 2016; Stanković *et al.*, 2017; Tepper and Sapiro, 2016; Zheng *et al.*, 2016), such extensions are not straightforward, since, owing to inherent properties of graphs as irregular but interconnected domains, even an operation which is very simple in classical time-domain analysis, like the time shift, cannot be straightforwardly generalized to graph signal domain. This has resulted in several approaches to the definition of the graph shift operator, and much ongoing research in this domain (Shuman *et al.*, 2012, 2016; Stanković *et al.*, 2017; Tepper and Sapiro, 2016; Zheng *et al.*, 2016).

A common approach to signal windowing in the graph domain is to utilize the eigenspectrum of a graph to obtain window function for each graph vertex (Shuman *et al.*, 2013). Another possibility is to define the window support as a local neighborhood for each vertex (Stanković *et al.*, 2017). In either case, the localization window is defined based on a set of vertices that contain the current vertex, n, and all vertices that are close in some sense to the vertex n, that is, a neighborhood of vertex n. In this monograph, special attention is devoted to the class of local graph Fourier transform approaches which can be implemented in the vertex domain, since this domain often offers a basis for numerically efficient analysis in the case of very large graphs.

Notice that, as in classical signal analysis, a localization window should be narrow enough so as to provide good localization of signal properties, but at the same time wide enough to produce high resolution in the spectral domain.

With vertex-frequency analysis serving as a key to graph signal estimation, filtering, and efficient representation, two forms of the local graph Fourier transform inversion are considered here, while the inversion condition is defined within the framework of frames, that is, based on the analysis of energy of the graph spectrogram. A relation between the graph wavelet transform and the local graph Fourier transform implementation and its inversion is also established.

Remark 33: The energy versions of the vertex-frequency representations are also considered, as these representations can be implemented without a localization window, and they can serve as estimators of the local smoothness index.

The reduced interference vertex-frequency distributions, which satisfy the marginal property and localize graph signal energy in the vertex-frequency domain are also defined, and are subsequently related to classical time-frequency analysis, as a special case.

Consider a graph with N vertices, $n \in \mathcal{V} = \{0, 1, \ldots, N-1\}$, which are connected with edges whose weights are W_{mn}. Spectral analysis of graphs is most commonly based on the eigendecomposition of the graph Laplacian, $\mathbf{L}$, or the adjacency matrix, $\mathbf{A}$. By default, we shall assume the decomposition of the graph Laplacian, $\mathbf{L}$, if not stated otherwise.

8.1 Localized Graph Fourier Transform (LGFT)

The localized graph Fourier transform (LGFT), denoted by $S(m,k)$, can be considered as an extension of the standard time-localized (short-time) Fourier transform (STFT), and can be calculated as the GFT of a signal, $x(n)$, multiplied by an appropriate vertex localization window function, $h_m(n)$, to yield

$$S(m,k) = \sum_{n=0}^{N-1} x(n) h_m(n)\, u_k(n). \tag{8.1}$$

In general, it is desired that a graph window function, $h_m(n)$, should be such that it localizes the signal content around the vertex m. To this end, its values should be close to 1 at vertex m and vertices in its close neighborhood, while it should approach to 0 for vertices that are far from vertex m. For an illustration of the concept of localization window on a graph see Figure 8.2, panels (a) and (c).

The localized GFT in (8.1) admits a matrix notation, $\mathbf{S}$, and contains all elements, $S(m,k)$, $m = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, N-1$. The columns of $\mathbf{S}$ which correspond to a vertex m are given by

$$\mathbf{s}_m = \text{GFT}\{x(n)h_m(n)\} = \mathbf{U}^T \mathbf{x}_m,$$

where $\mathbf{x}_m$ is the vector of which the elements, $x(n)h_m(n)$, are equal to the graph signal samples, $x(n)$, multiplied by the window function, $h_m(n)$, centered at the vertex m, while matrix $\mathbf{U}$ is composed of the eigenvectors $\mathbf{u}_k$, with elements $u_k(n)$, $k = 0, 1, \ldots, N-1$, of the graph Laplacian as its columns.

Special cases:

- For $h_m(n) = 1$, the localized vertex spectrum is equal to the standard spectrum, $S(m,k) = X(k)$, in (8.1) for each m; this means that no vertex localization is performed.
- If $h_m(m) = 1$ and $h_m(n) = 0$ for $n \neq m$, the localized vertex spectrum is equal to the graph signal, $S(m,0) = x(m)/\sqrt{N}$, for $k = 0$.

In the following, we outline ways to create vertex domain windows with desirable localization characteristics, and address two methods for defining graph localization window functions, $h_m(n)$:

- Spectral domain definition of windows, $h_m(n)$, which are defined using their spectral basic function. The spectral domain definition of the window is shown to be related to the wavelet transform.
- Vertex domain window definitions, with one method bearing a direct relation to the spectral analysis of the graph window, while the other method represents a purely vertex domain formulation.

Windows Defined in the GFT Domain

The basic function of a window, $h(n)$, can be conveniently defined in the spectral domain, for example, in the form

$$H(k) = C \exp(-\lambda_k \tau), \tag{8.2}$$

where C denotes the "window amplitude" and $\tau > 0$ is a constant which determines the window width in the spectral domain. Notice that the graph shifted and "modulated" versions of this window are straightforwardly obtained using the generalized convolution of graph signals, defined in Section 3.9. The graph-shifted window in the vertex

domain is then defined by the IGFT of $H(k)u_k(m)$, to give the window localized at the vertex m, denoted by $h_m(n)$, as in (3.46), in the form

$$h_m(n) = h(n) * \delta_m(n) = \sum_{k=0}^{N-1} H(k)u_k(m)u_k(n). \tag{8.3}$$

An example of two windows obtained in this way is given in Figures 8.2(a), (b).

Observe that the exponential function in (8.2) corresponds to a Gaussian window in classical analysis (thus offering the best time-frequency concentration (Boashash, 2015; Cohen, 1995; Stanković *et al.*, 2014)), since graph signal processing on a path graph reduces to classical signal analysis. In this case, the eigenvalues of the graph Laplacian, λ, may be related to the frequency, ω, in classical signal analysis as $\lambda \sim \omega^2$.

Properties of graph window functions. The graph window which is localized at the vertex m, and defined by (8.3), satisfies the following properties:

W_1: Symmetry, $h_m(n) = h_n(m)$, which follows from the definition in (8.3).

W_2: A sum of all coefficients of a localized window, $h_m(n)$, is equal to $H(0)$, since

$$\begin{aligned}\sum_{n=0}^{N-1} h_m(n) &= \sum_{k=0}^{N-1} H(k)u_k(m) \sum_{n=0}^{N-1} u_k(n) \\ &= \sum_{k=0}^{N-1} H(k)u_k(m)\delta(k)\sqrt{N} = H(0),\end{aligned}$$

with $\sum_{n=0}^{N-1} u_k(n) = \delta(k)\sqrt{N}$, following from the definition of the eigenvectors, $u_k(n)$.

W_3: The Parseval theorem for $h_m(n)$ has the form

$$\sum_{n=0}^{N-1} |h_m(n)|^2 = \sum_{k=0}^{N-1} |H(k)u_k(m)|^2. \tag{8.4}$$

These properties will be used in the sequel in the inversion analysis of the LGFT.

Based on the above properties, the LGFT can now be written as

$$S(m,k) = \sum_{n=0}^{N-1} x(n) h_m(n)\, u_k(n) \tag{8.5}$$

$$= \sum_{n=0}^{N-1} \sum_{p=0}^{N-1} x(n) H(p) u_p(m) u_p(n)\, u_k(n). \tag{8.6}$$

The modulated (frequency shifted) version of the window centered at a vertex m and for a spectral index k will be referred to as the *vertex-frequency kernel*, $\mathcal{H}_{m,k}(n)$, which is defined as

$$\mathcal{H}_{m,k}(n) = h_m(n) u_k(n) = \left(\sum_{p=0}^{N-1} H(p) u_p(m) u_p(n) \right) u_k(n). \tag{8.7}$$

Using the kernel notation, it becomes obvious that the LGFT in (8.6), for a given vertex m and a spectral index k, physically represents a projection of a graph signal, $x(n)$, onto the graph kernel, $\mathcal{H}_{m,k}(n)$, that is,

$$S(m,k) = \langle \mathcal{H}_{m,k}(n), x(n) \rangle = \sum_{n=0}^{N-1} \mathcal{H}_{m,k}(n) x(n). \tag{8.8}$$

Remark 34: The classical STFT, a basic tool in time-frequency analysis, can be obtained as a special case of the GFT when the graph is directed and circular. For this type of graph, the eigendecomposition of the adjacency matrix produces complex-valued eigenvectors of the form $u_k(n)\sqrt{N} = \exp(j2\pi nk/N)$. Then, having in mind the *complex nature of these eigenvectors*,

$$S(m,k) = \sum_{n=0}^{N-1} \sum_{p=0}^{N-1} x(n) H(p) u_p^*(m) u_p(n) u_k^*(n), \tag{8.9}$$

the value of $S(m,k)$ in (8.5) becomes the standard STFT, that is

$$S(m,k) = \frac{1}{N^{3/2}} \sum_{n=0}^{N-1} \sum_{p=0}^{N-1} x(n) H(p) e^{-j\frac{2\pi}{N}mp} e^{j\frac{2\pi}{N}np} e^{-j\frac{2\pi}{N}nk},$$

$$= \frac{1}{N} \sum_{n=0}^{N-1} x(n) h(n-m) e^{-j2\pi nk/N}, \tag{8.10}$$

where $h(n)$ is the inverse DFT of $H(k)$.

Example 16: To illustrate the principle of local vertex-frequency representations, consider the graph and the graph signal from Figure 8.1. A graph with $N = 100$ vertices, randomly placed on the so called Swiss roll surface, is shown in Figure 8.1(a). The vertices are connected with edges whose weights are defined as $W_{mn} = \exp(-r_{mn}^2/\alpha)$, where r_{mn} is the distance between the vertices m and n, measured along the Swiss roll manifold, and α is a constant. Small weight values were hard-thresholded

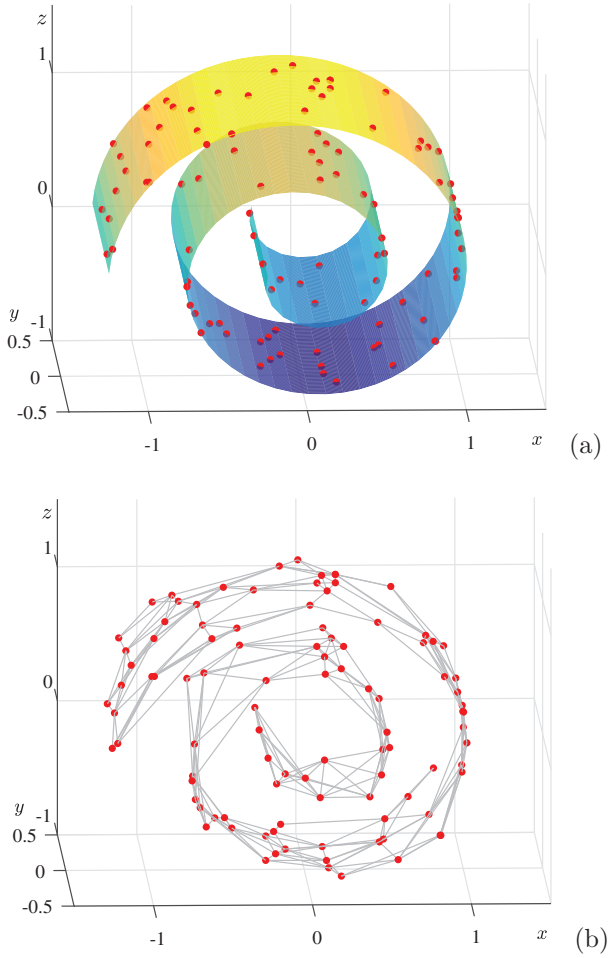


Figure 8.1: Continued.

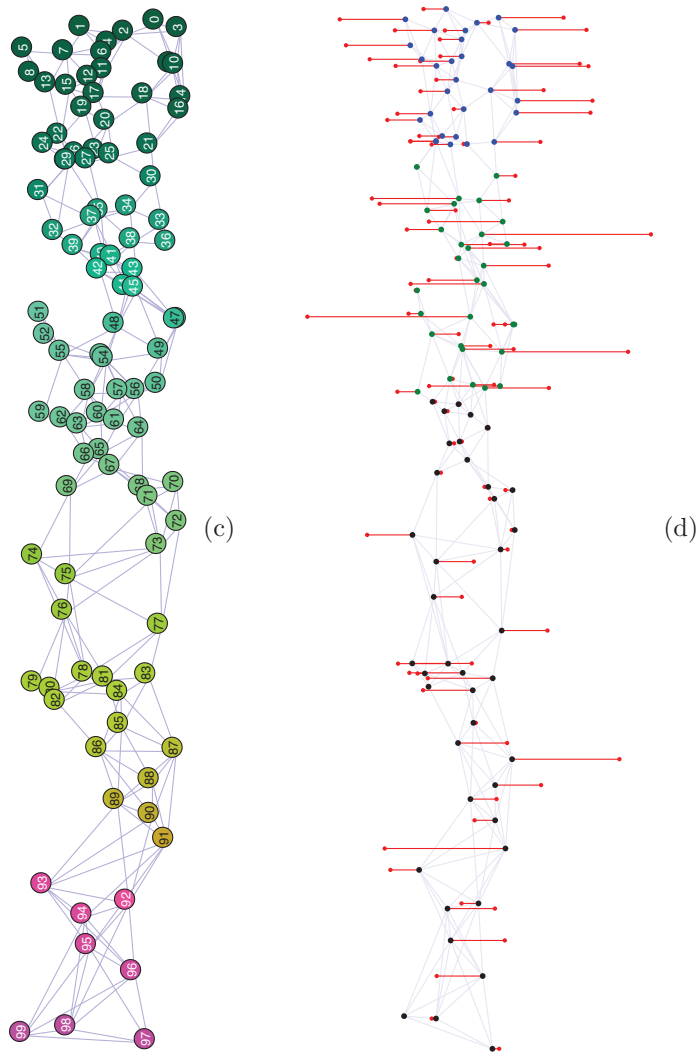


Figure 8.1: Concept of a signal on a graph. (a) Vertices on a three-dimensional manifold Swiss roll surface. (b) A graph representation on the Swiss roll manifold. (c) Two-dimensional presentation of the three-dimensional graph from (b), with vertex colors defined by the three smoothest graph Laplacian eigenvectors $u_1(n)$, $u_2(n)$, and $u_3(n)$. (d) A signal observed on the graph in (c), which is composed of three Laplacian eigenvectors (signal components). The supports of these three components are designated by different vertex colors. The vertex-frequency representations are then assessed based on their ability to accurately resolve and localize these three graph signal components.

to zero, in order to reduce the number of edges associated with each vertex to only a few strongest ones. The so produced graph is shown in Figure 8.1(b), and its two-dimensional presentation in Figure 8.1(c). Vertices are ordered so that the values of the Fiedler eigenvector, $u_1(n)$, are nondecreasing. More detail about the Swiss role graph are given in Part III.

A signal on this graph was created so as to be composed of parts of three Laplacian eigenvectors. For the subset, $\mathcal{V}_1$, of all vertices, $\mathcal{V}$, which comprises the vertices with indices from $m = 0$ to $m = 29$, the eigenvector with the spectral index $k = 8$ was used. For the subset, $\mathcal{V}_2$, with the vertex indices from $m = 30$ to $m = 59$, the signal was equal to the eigenvector $u_{66}(n)$, that is, with $k = 66$. The remaining vertices form the vertex subset $\mathcal{V}_3$, and the signal on this subset was equal to the eigenvector with the spectral index $k = 27$. The amplitudes of these eigenvectors were scaled too.

Consider now the vertex-frequency localization kernels,

$$\mathcal{H}_{m,k}(n) = h_m(n)u_k(n),$$

shown in Figure 8.2. The constant eigenvector, $u_0(n) = 1/\sqrt{N}$, was used in the panel shown in Figure 8.2(a) at $m = 34$. In this case, the localization window, $h_{34}(n)$, is shown since $\mathcal{H}_{34,0}(n) = h_{34}(n)/\sqrt{N}$. The illustration is repeated in the panel in Figure 8.2(c) for the vertex $m = 78$. The frequency shifted version of these two vertex-domain kernels, shown in Figures 8.2(a) and (c), are given respectively in Figures 8.2(b) and (d), where $\mathcal{H}_{m,20}(n) = h_m(n)u_{20}(n)$ is shown for $m = 34$ and $m = 78$, respectively.

Next, the vertex-frequency representation, $S(n, k)$, using the LGFT and the localization window defined in the spectral domain is shown in Figure 8.3. From this representation, we can clearly identify the three constituent signal components, within their intervals of support. The marginal properties, such as the projections of $S(n, k)$ onto the vertex index axis and the spectral index axis, are also clearly distinguishable. From the marginal properties, we can conclude that the considered graph signal in hand is spread over all vertex indices, while its spectral localization is dominated by the three spectral indices which correspond to the three components of the original graph signal. In an ideal case of

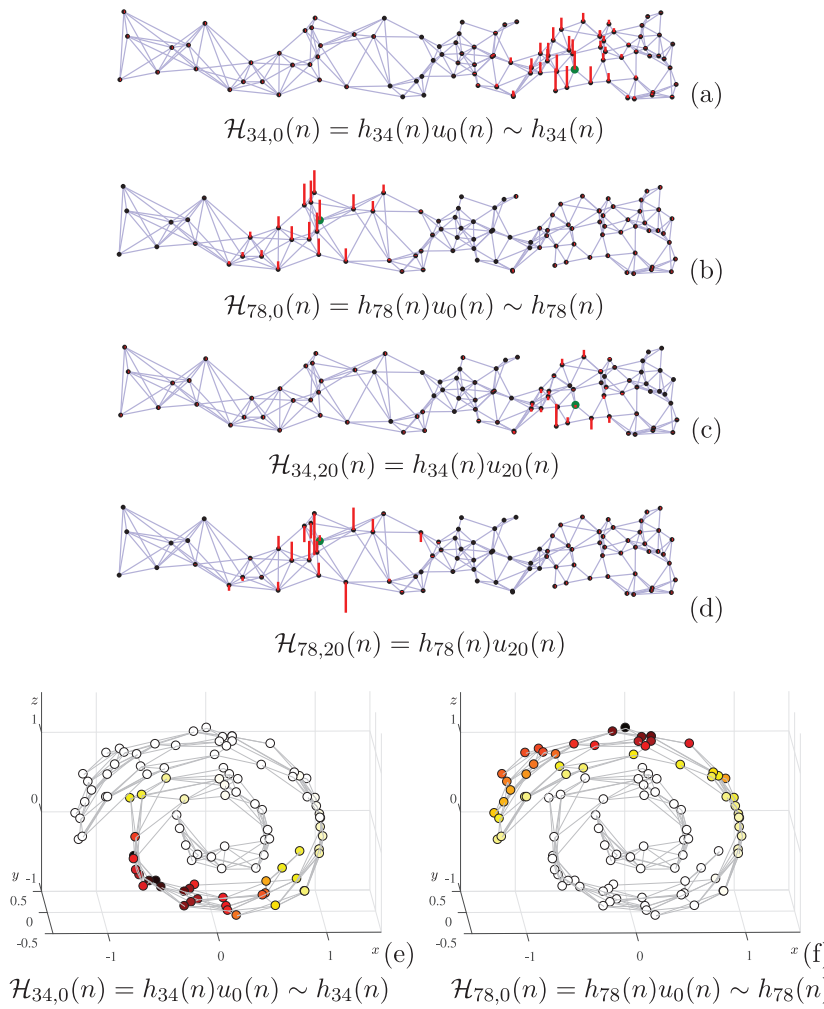


Figure 8.2: Illustration of localization kernels, $\mathcal{H}_{m,k}(n) = h_m(n)u_k(n)$, for vertex-frequency analysis based on *spectral domain defined windows* within the local graph Fourier transform, $S(m,k) = \sum_{n=0}^{N-1} x(n)\mathcal{H}_{m,k}(n)$. (a) Localization kernel $\mathcal{H}_{34,0}(n) = h_{34}(n)u_0(n) \sim h_{34}(n)$, for a constant eigenvector, $u_0(n) = 1/\sqrt{N}$, centered at the vertex $m = 34$. (b) The same localization kernel as in (a) but centered at the vertex $m = 78$. (c) Localization kernel, $\mathcal{H}_{34,20}(n) = h_{34}(n)u_{20}(n)$, centered at the vertex $m = 34$ and frequency shifted by $u_{20}(n)$. Notice that the variations in kernel amplitude indicate the effects of modulation of the localization window, $h_m(n)$. (d) The same localization kernel as in (c), but centered at the vertex $m = 78$. (e) Three-dimensional representation of the kernel $\mathcal{H}_{34,0}(n) = h_{34}(n)u_0(n)$. (f) Three-dimensional representation of the kernel $\mathcal{H}_{78,0}(n) = h_{78}(n)u_0(n)$.

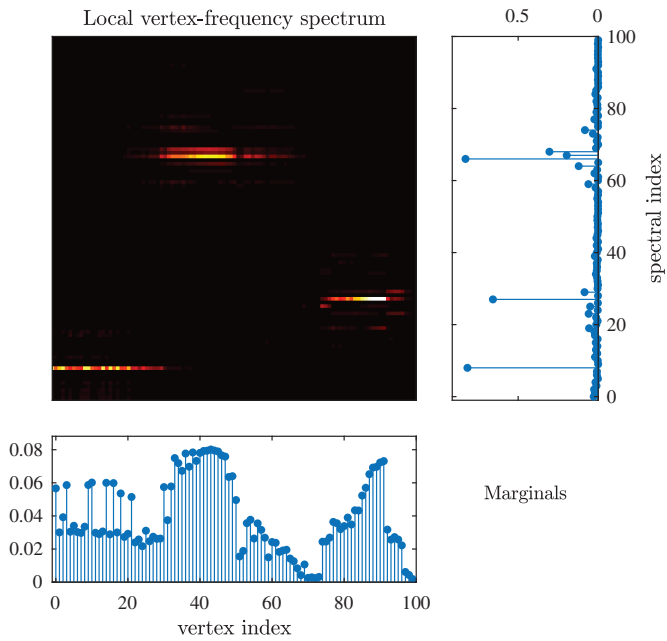


Figure 8.3: Local vertex-frequency spectrum calculated using the LGFT and the vertex-frequency localized kernels defined in the spectral domain, as in (8.7). From this representation, observe that the graph signal consists of three distinct components located at spectral indices $k = 8$, $k = 66$, and $k = 27$, with the corresponding vertex index subsets $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$, where $\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{V}_3 = \mathcal{V}$. The marginal (vertex and spectrum-wise) properties are shown in the panels to the right and below the vertex-frequency representation. Observe that, while the graph signal is spread across all vertices, its spectral content is localized at the three spectral indices which correspond to the constituent signal components. In an ideal case of vertex-frequency analysis, these marginals should be respectively equal to $|x(n)|^2$ and $|X(k)|^2$.

vertex-frequency analysis, these marginals should respectively be equal to $|x(n)|^2$ and $|X(k)|^2$, which is not the case here.

The calculation of (8.8) is computationally demanding, as in addition to the double summation (where it can be reduced having in mind the low-pass nature of the function $H(p)$ and its possible truncation), the bulk of computational load comes from the eigendecomposition of the graph Laplacian. Although this decomposition is performed only once for a given graph and is signal independent, the full eigendecomposition of a graph with N vertices requires an oder of N^3 numerical operations.

For a large graph, this can limit the application of this approach. The issue of computational complexity was a motivation to introduce vertex-frequency analysis without the need for eigendecomposition, which will be presented later.

Spectral Domain Localization of the LGFT

Recall that the classical STFT admits frequency localization in the spectral domain; this is achieved based on the DFT of the original signal and a spectral domain window. For graph signals, we may also adapt this approach to perform signal localization in the spectral domain, whereby the LGFT is obtained as an inverse GFT of $X(p)$ that is localized by a spectral domain window, $H(k-p)$, which is centered around spectral index k, that is

$$S(m,k) = \sum_{p=0}^{N-1} X(p)H(k-p)\, u_p(m). \tag{8.11}$$

Note that this form of the LGFT can be entirely implemented in the graph spectral domain, without a graph shift operator in the vertex domain.

Remark 35: Recall that the classical time-frequency analysis counterpart of (8.11) is Stanković *et al.* (2014)

$$S(m,k) = \frac{1}{\sqrt{N}} \sum_{p=0}^{N-1} X(p)H(k-p)e^{j\frac{2\pi}{N}mp}.$$

The spectral domain LGFT form in (8.11) can be implemented using band-pass transfer functions, $H_k(\lambda_p) = H(k-p)$, as

$$S(m,k) = \sum_{p=0}^{N-1} X(p)H_k(\lambda_p)\, u_p(m). \tag{8.12}$$

The elements $S(m,k)$, $m = 0, 1, \ldots, N-1$ of the LGFT matrix $\mathbf{S}$ can also be written in a matrix form, where the k-th column is defined as

$$\mathbf{s}_k = \text{IGFT}_p\{X(p)H_k(\lambda_p)\} = \mathbf{U}\ H_k(\mathbf{\Lambda})\mathbf{X}, \tag{8.13}$$

where $H_k(\mathbf{\Lambda})$ is a diagonal matrix with elements $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$.

Remark 36: The kernel in (8.7) is defined based on a low-pass transfer function $H(k)$, which is appropriately shifted in the spectral domain using the modulation term, $u_k(n)$. The transfer function in (8.12), $H_k(\lambda_p)$, is centered (shifted) at a spectral index, k, by definition. Hence, in this case, the modulation term, $u_k(n)$, is not needed and the kernel is now of the form

$$\mathcal{H}_{m,k}(n) = \sum_{p=0}^{N-1} H_k(\lambda_p) u_p(m) u_p(n). \tag{8.14}$$

LGFT Realization with Band-Pass Functions

Assume that the GFT of the localization window, $h_m(n)$, corresponds to the transfer function of a band-pass system on a graph, centered at an eigenvalue, λ_k, and around it, and that it is defined in the form of a polynomial given by

$$H_k(\lambda_p) = h_{0,k} + h_{1,k}\lambda_p + \cdots + h_{M-1,k}\lambda_p^{M-1}, \tag{8.15}$$

with $(M-1)$ as the polynomial order and $k = 0, 1, \ldots, K$, where K is the number of spectral bands.

The vertex shifted version of the window, $h_m(n)$, has the GFT of the form, $\text{GFT}\{h(n) * \delta_m(n)\} = H(p)u_p(m)$. Therefore, the inverse GFT of $H_k(\lambda_p)u_p(m)$ represents a vertex domain kernel, where $H_k(\lambda_p)$ is centered at the spectral index k by definition, while $u_p(m)$ corresponds to the shift in the vertex domain which centers the window at the vertex m. In other words, this kernel, centered around the spectral index k and vertex m, is defined as

$$\mathcal{H}_{m,k}(n) = \sum_{p=0}^{N-1} H_k(\lambda_p) u_p(m) u_p(n). \tag{8.16}$$

Remark 37: It is important to emphasize crucial difference between the vertex-frequency kernels in (8.7) and (8.16). The kernel in (8.7) is defined based on the low-pass transfer function $H(k)$, such as in (8.2),

appropriately shifted in the vertex domain and the spectral domain, to be centered at a vertex m and at a spectral index k. This is achieved involving adequate modulation terms $u_k(n)$ and $u_p(m)$. The transfer function in the kernel given by (8.16), $H_k(\lambda_p)$, is centered at k by definition (8.15). Hence, it is needed to perform the spectral modulation only, by $u_p(m)$, in order to center the kernel, $\mathcal{H}_{m,k}(n)$, at a vertex m. Therefore, the main difference between the kernels in (8.7) and (8.16) is that the spectral shift in (8.7) is achieved by a modulation in the vertex domain using $u_k(n)$, while in (8.16) the kernel is directly shifted (defined as a pass-band function) in the spectral domain.

Classical time-frequency domain kernel. To additionally clarify the previous two forms of kernels, we will observe their special cases for a circular directed graph and write the kernels in the classical time-frequency domain.

The kernel defined by (8.7) uses low-pass function $H(k)$ and assumes the following form

$$\begin{aligned}\mathcal{H}_{m,k}(n) &= \frac{1}{N^{3/2}}\sum_{p=0}^{N-1} H(p)e^{-j\frac{2\pi}{N}mp}e^{j\frac{2\pi}{N}np}e^{-j\frac{2\pi}{N}kn} \\ &= \frac{1}{N}h(n-m)e^{-j\frac{2\pi}{N}kn} = \frac{1}{\sqrt{N}}h_k(n-m),\end{aligned}$$

which is shifted for m in time, and modulated by the kth eigenvector elements $u_k^*(n) = e^{-j\frac{2\pi}{N}kn}/\sqrt{N}$, to achieve centering around the spectral index k.

The classical time-frequency domain form of the kernel in (8.16) is given by

$$\begin{aligned}\mathcal{H}_{m,k}(n) &= \frac{1}{N}\sum_{p=0}^{N-1} H_k(\lambda_p)e^{-j\frac{2\pi}{N}mp}e^{j\frac{2\pi}{N}np} \\ &= \frac{1}{N}\sum_{p=0}^{N-1} H(p-k)e^{-j\frac{2\pi}{N}mp}e^{j\frac{2\pi}{N}np} = \frac{1}{\sqrt{N}}h_k(n-m),\end{aligned}$$

where $h_k(n-m)$ is the temporary shifted version of $h_k(n) = \text{IGFT}\{H_k(\lambda_p)\} = \text{IDFT}\{H(k-p)\}$, which corresponds to the already frequency shifted (band-pass) transfer function $H_k(\lambda_p) = H(p-k)$.

In the case of kernel (8.16), the local vertex-frequency transform for a vertex, m, and a spectral index, k, becomes

$$\begin{aligned} S(m,k) &= \sum_{n=0}^{N-1} \mathcal{H}_{m,k}(n)x(n) \\ &= \sum_{n=0}^{N-1}\sum_{p=0}^{N-1} x(n)H_k(\lambda_p)u_p(m)u_p(n) = \sum_{p=0}^{N-1} X(p)H_k(\lambda_p)u_p(m). \end{aligned} \tag{8.17}$$

The relation (8.17) can be written in a vector form as

$$\mathbf{s}_k = \mathbf{U}H_k(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x} = H_k(\mathbf{L})\mathbf{x} = \sum_{p=0}^{M-1} h_{p,k}\mathbf{L}^p\,\mathbf{x}, \tag{8.18}$$

where $\mathbf{s}_k$ is the column vector with elements $S(m,k)$, $m = 0, 1, \ldots, N-1$, and the property of the eigendecomposition of a matrix polynomial is used in derivation. *The number of bands (shifted transfer functions, $H_k(\lambda_p)$, $k = 0, 1, \ldots, K$) is equal to $K+1$ and is not related to the total number of indices, N.*

Example 17: Consider the simplest decomposition into a low-pass and high-pass part of a graph signal, with $K = 1$. In this case, the two values, $k = 0$ and $k = 1$, represent respectively the low-pass part and high-pass part of the graph signal. Such a decomposition can be achieved using the graph Laplacian with $h_{0,0} = 1$, $h_{0,1} = -1/\lambda_{\max}$, and $h_{1,0} = 0$, $h_{1,1} = 1/\lambda_{\max}$, where the coefficients are chosen so as to form a simple linearly decreasing function of λ_p for the low-pass, and a linearly increasing function of λ_p for the high-pass, in the corresponding transfer functions. These low-pass and high-pass transfer functions are respectively given by

$$H_0(\lambda_p) = \left(1 - \frac{\lambda_p}{\lambda_{\max}}\right), \quad H_1(\lambda_p) = \frac{\lambda_p}{\lambda_{\max}},$$

which leads to the vertex domain implementation of the LGFT in the form

$$\mathbf{s}_0 = \left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{L}\right)\mathbf{x}, \quad \mathbf{s}_1 = \frac{1}{\lambda_{\max}}\mathbf{L}\,\mathbf{x}.$$

To improve the spectral resolution, we can employ the same transfer function, but divide the low-pass part into its low-pass and high-pass part. The same can be performed for the high-pass part, to obtain

$$\mathbf{s}_{00} = \left(\mathbf{I} - \frac{\mathbf{L}}{\lambda_{\max}}\right)^2 \mathbf{x}, \quad \mathbf{s}_{01} = 2\left(\mathbf{I} - \frac{\mathbf{L}}{\lambda_{\max}}\right)\frac{\mathbf{L}}{\lambda_{\max}}\mathbf{x}, \quad \mathbf{s}_{11} = \frac{\mathbf{L}^2}{\lambda_{\max}^2}\mathbf{x}.$$

The factor 2 appears in the new middle pass-band, $\mathbf{s}_{01}$, since the low-high-pass and the high-low-pass components are the same.

A division into $(K+1)$ bands would correspond to the terms of a binomial form

$$((\mathbf{I} - \mathbf{L}/\lambda_{\max}) + \mathbf{L}/\lambda_{\max})^K \mathbf{x},$$

with the corresponding transfer functions in the vertex domain given by

$$H_k(\mathbf{L}) = \binom{K}{k}\left(\mathbf{I} - \frac{1}{\lambda_{\max}}\mathbf{L}\right)^{K-k}\left(\frac{1}{\lambda_{\max}}\mathbf{L}\right)^k.$$

Example 18: Consider the transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$ in the spectral domain, corresponding to the binomial form terms for $K = 25$, which are shown in Figure 8.4(a). These functions are used for the LGFT calculation at vertex indices $m = 0, 1, \ldots, N-1$ in the $k = 0, 1, \ldots, K$ bands for the graph and signal from Figure 8.1. Since the bands are quite spread out, the resulting LGFT is also spread along the frequency axis. The frequency concentration can be improved by reassigning the values of $S(m, k)$ to the position of their maximum value along the frequency band index, k, for each vertex index, m. The so reassigned LGFT values are given in Figure 8.5.

Of course, any band-pass function, $H_k(\mathbf{\Lambda})$, can be used in (8.12) or (8.17) to produce the LGFT in the form

$$\mathbf{s}_k = \mathbf{U}H_k(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x} = H_k(\mathbf{L})\mathbf{x}. \tag{8.19}$$

Commonly used examples of such band-pass functions are the spline or raised cosine (Hann window) functions. We will next use the general form of the shifted raised cosine functions as the transfer functions,

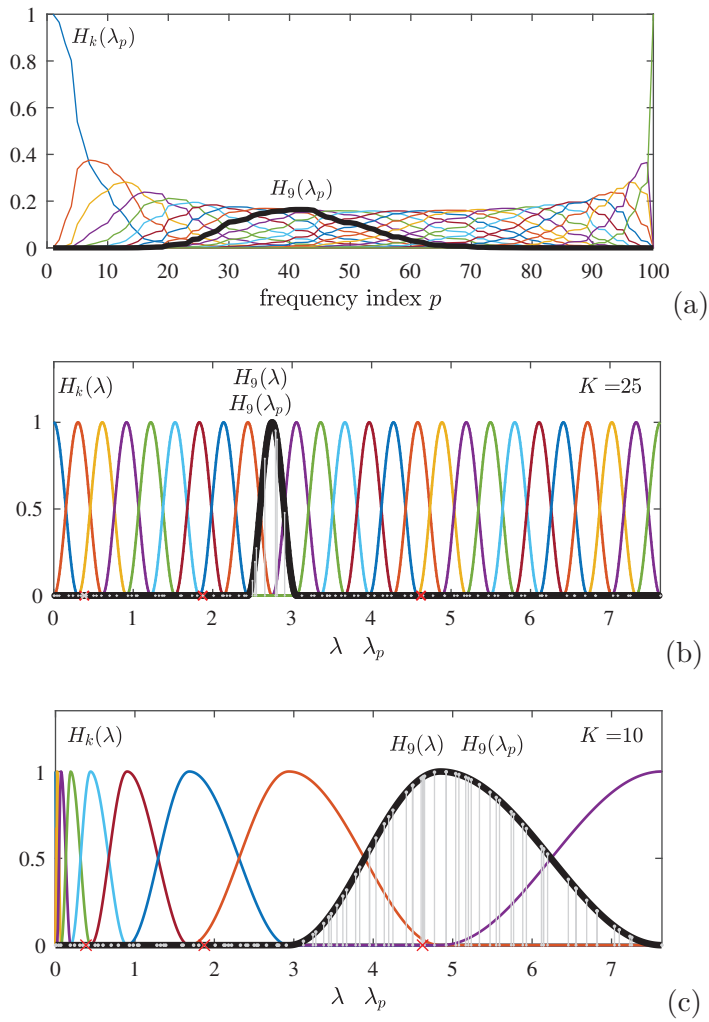


Figure 8.4: Exemplar of transfer functions in the spectral domain. (a) The spectral domain transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$ which correspond to the binomial form terms for $K = 25$. (b) The transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$ which correspond to the raised cosine (Hann) window form for $K = 25$. (c) The spectral index-varying (wavelet-like) transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$ which correspond to the raised cosine (Hann) window form for $K = 10$. The transfer function $H_9(\lambda)$ is designated by the thick black line for each considered domain, while its discrete values at λ_p, $H_9(\lambda_p)$, are shown in gray, in panels (b) and (c).

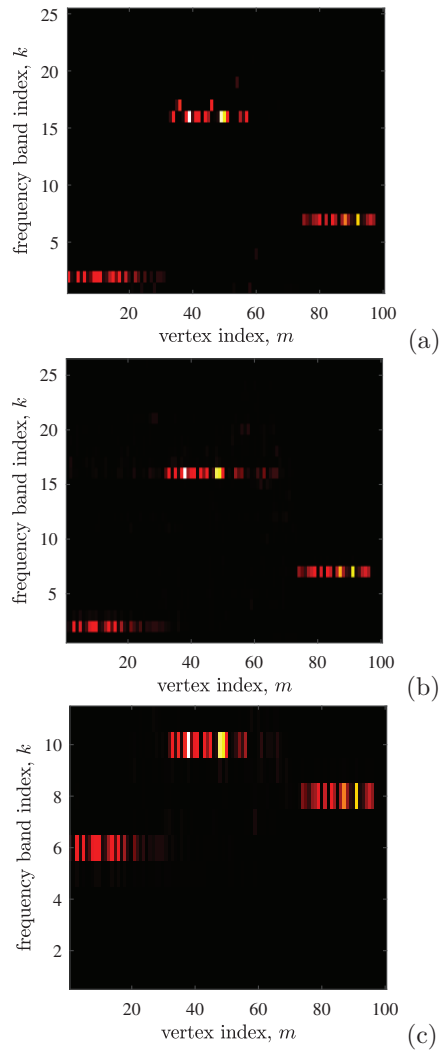


Figure 8.5: Vertex-frequency representation of a three-component signal in Figure 8.1(d). (a) The LGFT of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.4(a). The LGFT values, $S(m,k)$, were reassigned to the position of its maximum value along the frequency band index, k, for each vertex index, m. (b) The LGFT of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.4(b). The LGFT values, $S(m,k)$, were reassigned to the positions of their maximum values along the frequency band index, k, for each vertex index, m. (c) The LGFT of the signal from Figure 8.1(d), calculated using the wavelet-like transfer functions for frequency selection given in Figure 8.4(c).

defined by

$$H_k(\lambda) = \begin{cases} \sin^2\left(\frac{\pi}{2}\frac{a_k}{b_k - a_k}\left(\frac{\lambda}{a_k} - 1\right)\right), & \text{for } a_k < \lambda \leq b_k \\ \cos^2\left(\frac{\pi}{2}\frac{b_k}{c_k - bk}\left(\frac{\lambda}{b_k} - 1\right)\right), & \text{for } b_k < \lambda \leq c_k \\ 0, & \text{elsewhere,} \end{cases} \tag{8.20}$$

where $(a_k, b_k]$ and $(b_k, c_k]$, $k = 1, 2, \ldots, K$, define the spectral bands for $H_k(\mathbf{\Lambda})$. For uniform bands within $0 \leq \lambda \leq \lambda_{\max}$, the intervals can be defined by

$$\begin{aligned} a_k &= a_{k-1} + \frac{\lambda_{\max}}{K} \\ b_k &= a_k + \frac{\lambda_{\max}}{K} \\ c_k &= a_k + 2\frac{\lambda_{\max}}{K} \end{aligned} \tag{8.21}$$

with $a_1 = 0$. The initial transfer function, $H_0(\lambda)$, is defined using only $0 = b_0 \leq \lambda \leq c_0 = \lambda_{\max}/K$, while the last transfer function, $H_K(\lambda)$, is defined using the interval $a_K < \lambda \leq b_K = \lambda_{\max}$ in (8.20).

The raised cosine transfer function satisfy the following condition

$$\sum_{k=0}^{K} H_k(\lambda_p) = 1. \tag{8.22}$$

The conditions for graph signal reconstruction from the LGFT will be discussed in Section 8.2.

Example 19: The shifted raised cosine functions, defined by (8.20) and (8.21), are shown in Figure 8.4(b) for the graph from Figure 8.1, for $K = 25$. These functions are used for the LGFT calculation of the graph signal from Figure 8.1 at the vertex indices $m = 0, 1, \ldots, N-1$, and in $(K+1)$ spectral bands, $k = 0, 1, \ldots, K$. The absolute LGFT values are given in Figure 8.5(b). Spectral resolution depends on the number of bands K, with a larger number of spectral bands resulting in a higher spectral resolution.

Example 20: The experiment from Examples 18 and 19 is repeated with varying bounds of the spectral intervals in the raised cosine transfer

functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$. The spectral index-varying (wavelet-transform like) form of the raised cosine transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$, is defined by the interval bounds $\lambda_{\max}((1.5+p)/11.5)^5$, for $p = 0, 1, 2, \ldots, 10$,

$$
\begin{aligned}
a_k &\in \{0, 0.004, 0.02, 0.07, 0.19, 0.44, 0.9, 1.7, 2.9\}, \\
b_k &\in \{0.004, 0.02, 0.07, 0.19, 0.44, 0.9, 1.7, 2.9, 4.8\}, \\
c_k &\in \{0.02, 0.07, 0.19, 0.44, 0.9, 1.7, 2.9, 4.8, 7.63\}, \\
k &= 1, 2, \ldots, 9,
\end{aligned}
$$

and depicted in Figure 8.4(c). The LGFT values, $S(m, k)$, calculated with the so-obtained transfer functions, $H_k(\lambda_p)$, are shown in Figure 8.5(c). In order to illustrate the change of resolution in this case, the LGFT was reassigned to each eigenvalue λ_p, $p = 0, 1, \ldots, N-1$, and shown in Figure 8.6. As in classical wavelet transform, the spectral resolution is lower for the higher spectral indices.

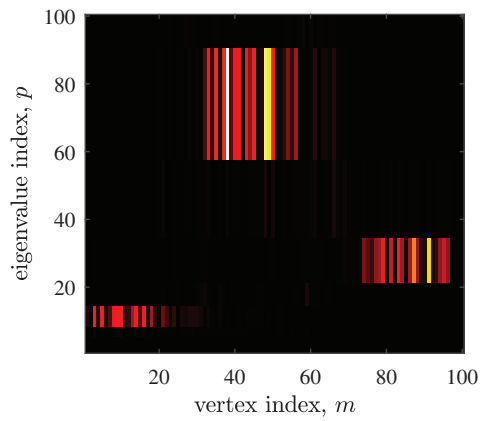


Figure 8.6: Vertex-frequency representation from Figure 8.5(c) with the axis of the eigenvalue index, p, instead of the frequency band index, k. The same value of LGFT, $S(m, k)$, is assigned to each spectral index, p, when $\lambda_p \in (\frac{a_k+b_k}{2}, \frac{b_k+c_k}{2}]$, and without any scaling.

Signal Adaptive LGFT

The spectral graph wavelet-like transform is just an example of varying spectral transfer functions in the LGFT, where the spectral resolution is the highest (spectral wavelet functions narrowest) for small values of the smoothness index, λ_p (Behjat and Van De Ville, 2019). The spectral resolution decreases as the spectral wavelet functions become wider for large smoothness index values, Figure 8.4(c). In general, the change of resolution may be arbitrary and signal adaptive, for example, the resolution may be higher for the spectral intervals of λ which are rich in signal components and lower within the intervals where there are no signal components.

Before introducing an example with a signal adaptive LGFT, we will modify the transfer functions, $H_k(\lambda_p)$, in (8.20) to satisfy the condition

$$\sum_{k=0}^{K} H_k^2(\lambda_p) = 1, \tag{8.23}$$

as this will be important for the frame-based LGFT inversion.

Notice that a simple transformation of the transfer functions, $H_k(\lambda_p) \to H_k^2(\lambda_p)$, would allow for the condition $\sum_{k=0}^{K} H_k^2(\lambda_p) = 1$ to hold instead of $\sum_{k=0}^{K} H_k(\lambda_p) = 1$. This means that a simple removal of squares in the sine and cosine functions in (8.20) would produce a form to satisfy the condition $\sum_{k=0}^{K} H_k^2(\lambda_p) = 1$. Both of these conditions will be used in Section 8.2 in various approaches to the graph signal reconstruction from the LGFT.

By removing the squares in the sine and cosine functions in (8.20), their first derivative loses continuity in λ at the end interval points. In order to preserve continuous derivatives, the arguments in the sine and cosine functions can be mapped by a polynomial,

$$v_x(x) = x^4(35 - 84x + 70x^2 - 20x^3), \quad \text{for } 0 \leq x \leq 1,$$

with $v_x(0) = 0$ and $v_x(1) = 1$. In this way, we arrive at the Meyer wavelet-like transfer functions (Meyer, 1992) for the LGFT calculation,

given by

$$H_k(\lambda) = \begin{cases} \sin\left(\frac{\pi}{2} v_x\left(\frac{a_k}{b_k - a_k}\left(\frac{\lambda}{a_k} - 1\right)\right)\right), & \text{for } a_k < \lambda \le b_k \\ \cos\left(\frac{\pi}{2} v_x\left(\frac{b_k}{c_k - b_k}\left(\frac{\lambda}{b_k} - 1\right)\right)\right), & \text{for } b_k < \lambda \le c_k \\ 0, & \text{elsewhere.} \end{cases} \tag{8.24}$$

The initial transfer function, $k = 0$, and the last transfer function, $k = K$, are calculated using only the half of the interval, as explained after the spectral band definition in relation (8.21).

Example 21: The transfer functions of the form defined in (8.24) are used with signal adaptive intervals. These intervals are defined in such a way that they are small (fine) around λ, where a significant signal spectral content is detected, and are big (rough) around λ where the signal spectral content is low, as in Figures 8.7(a) and (b). The intervals are narrow (with a high resolution) around the three signal components at $\lambda = 0.38$, $\lambda = 1.87$, and $\lambda = 4.62$. Vertex-frequency representation with these transfer functions is shown in Figures 8.7(c) and (d) with the spectral band index, k, and the assigned eigenvalue (spectral) index, p, as a spectral axis. Fine intervals around the spectral signal components allowed for high spectral resolution representation, as in Figure 8.7(c), with a smaller number of transfer functions $K+1 = 17$. A wider interval width for the third component resulted in a lower spectral resolution than in the case of the other two components.

Polynomial LGFT Approximation

Bandpass LGFT functions, $H_k(\lambda)$, $k = 0, 1, \ldots, K$, of the form (8.20) or (8.24) can be implemented using the Chebyshev finite $(M-1)$-order polynomial approximation, $\bar{P}_{k,M-1}(\lambda)$, $k = 0, 1, \ldots, K$, of the form

$$\bar{P}_{k,M-1}(\lambda) = \frac{c_{k,0}}{2} + \sum_{m=1}^{M-1} c_{k,m} \bar{T}_m(\lambda). \tag{8.25}$$

This leads to the vertex domain implementation of the spectral LGFT form, given by

$$\mathbf{s}_k = \bar{P}_{k,M-1}(\mathbf{L})\mathbf{x},$$

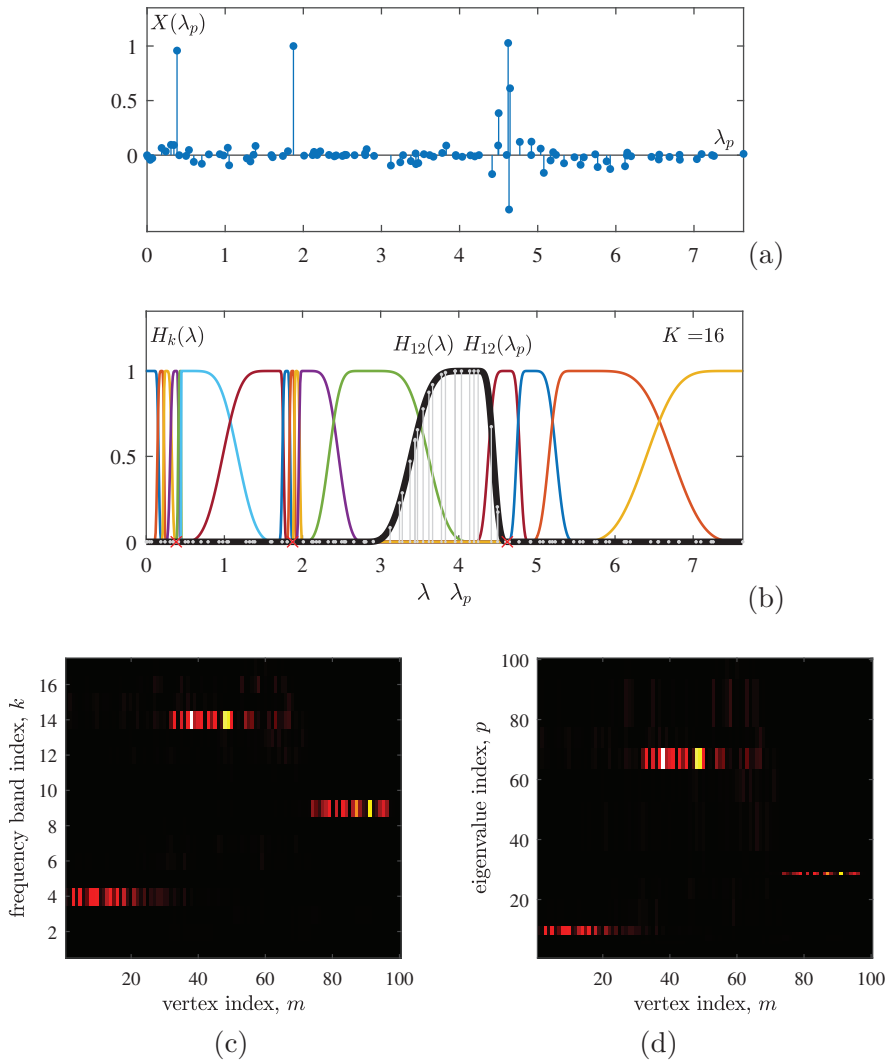


Figure 8.7: A graph signal and transfer functions in the spectral domain for a signal adaptive LGFT. (a) Graph signal in the spectral domain, $X(p)$, as a function of the eigenvalues, λ_p. (b) The spectral domain transfer functions $H_k(\lambda_p)$, $p = 0, 1, \ldots, N-1$, $k = 0, 1, \ldots, K$ which satisfy the condition $\sum_{k=0}^{K} H_k^2(\lambda_p) = 1$, with $K = 16$. (c) The LGFT of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in (b). (d) Vertex-frequency representation from (c) with the eigenvalue (spectral) index, p, axis instead of the frequency band index, k. The same value of LGFT, $S(m, k)$, is assigned to each spectral index, p, when $\lambda_p \in (\frac{a_k+b_k}{2}, \frac{b_k+c_k}{2}]$, without any scaling.

Table 8.1: Coefficients, $h_{i,k}$, $i = 0, 1, \ldots, M-1$, $k = 0, 1, \ldots, K$, for the polynomial calculation of the LGFT, $\mathbf{s}_k$, of a signal, $\mathbf{x}$, in various spectral bands, k, shown in Figure 8.8(b). The obtained LGFT of the three-component signal from Figure 8.1(d) is given in Figure 8.9(a)

$\mathbf{s}_k = (h_{0,k}\mathbf{I} + h_{1,k}\mathbf{L} + h_{2,k}\mathbf{L}^2 + h_{3,k}\mathbf{L}^3 + h_{4,k}\mathbf{L}^4 + h_{5,k}\mathbf{L}^5)\mathbf{x}$						
k	$h_{0,k}$	$h_{1,k}$	$h_{2,k}$	$h_{3,k}$	$h_{4,k}$	$h_{5,k}$
0	1.062	−1.925	1.168	−0.3115	0.03776	−0.001702
1	−0.002	1.773	−1.655	0.5357	−0.07250	0.003508
2	−0.154	1.016	−0.601	0.1295	−0.01155	0.000349
3	0.005	−0.301	0.621	−0.2674	0.04200	−0.002225
4	0.089	−0.748	0.869	−0.3042	0.04217	−0.002040
5	0.060	−0.381	0.319	−0.0704	0.00461	0.000000
6	−0.024	0.277	−0.430	0.2055	−0.03570	0.002040
7	−0.076	0.598	−0.714	0.2814	−0.04292	0.002225
8	−0.027	0.159	−0.122	0.0198	0.00177	−0.000349
9	0.087	−0.699	0.868	−0.3662	0.06140	−0.003508
10	−0.026	0.220	−0.293	0.1333	−0.02435	0.001536

for $k = 0, 1, 2, \ldots, K$, with

$$\begin{aligned} \bar{P}_{k,M-1}(\mathbf{L}) &= \frac{c_{k,0}}{2} + \sum_{m=1}^{M-1} c_{k,m}\bar{T}_m(\mathbf{L}), \\ &= h_{0,k}\mathbf{I} + h_{1,k}\mathbf{L} + h_{2,k}\mathbf{L}^2 + \cdots + h_{(M-1),k}\mathbf{L}^{M-1} \end{aligned} \tag{8.26}$$

as discussed in Section 3.5 and shown in Table 8.1. The polynomial form in (8.26) uses only the $(M-1)$-neighborhood in calculation of the LGFT for each considered vertex, without the need for eigendecomposition analysis, thus significantly reducing the computational cost.

Example 22: Consider the shifted transfer functions, $H_k(\lambda)$, $k = 0, 1, \ldots, K$, defined by (8.20) and (8.21), shown in Figure 8.8(a), for $K = 10$. Functions $H_k(\lambda)$ satisfy $\sum_{k=0}^{K} H_k(\lambda) = 1$, which is numerically confirmed and designated by the horizontal dotted line in 8.8(a). Each individual transfer function, $H_k(\lambda)$, is approximated using the Chebyshev polynomial, $\bar{P}_{k,M-1}, k = 0, 1, \ldots, K$, as detailed in Section 3.5, with three polynomial orders defined by $M = 6$, $M = 20$ and $M = 80$. These polynomial approximations are shown in Figures 8.8(b)–(d). In each considered case, summations $\sum_{k=0}^{K} \bar{P}_{k,M-1}(\lambda)$ are calculated. It can

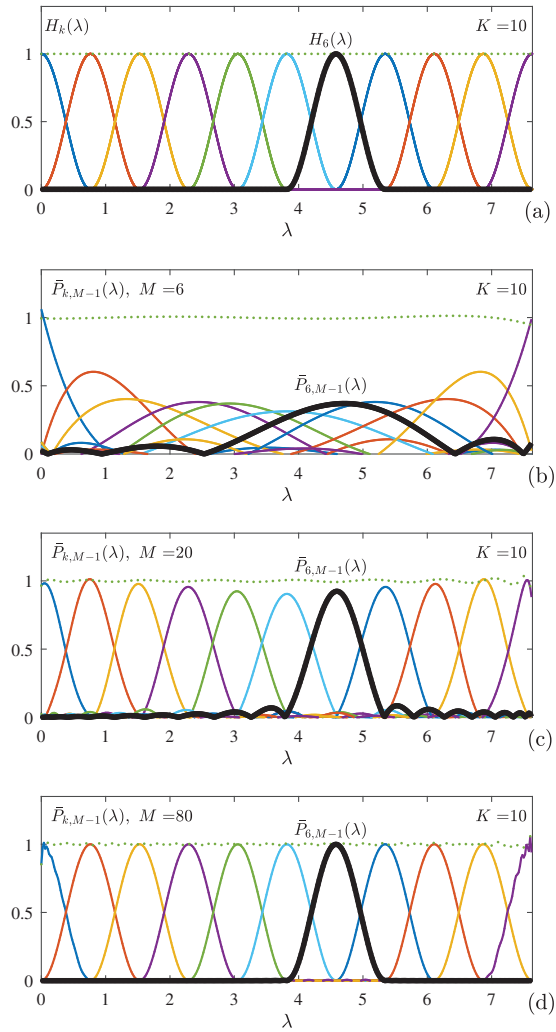


Figure 8.8: Chebyshev approximation of LGFT transfer functions, which correspond to the raised cosine window in the spectral domain. (a) Original transfer functions $H_k(\lambda),\ k = 0, 1, \ldots, K$, for $K = 10$. The dotted horizontal line designates $\sum_{k=0}^{K} H_k(\lambda)$. (b) Polynomial Chebyshev approximations, $\bar{P}_{k,M-1}(\lambda), k = 0, 1, \ldots, K$, with $M = 6$. (c) Polynomial Chebyshev approximations, $\bar{P}_{k,M-1}(\lambda), k = 0, 1, \ldots, K$, with $M = 20$. (d) Polynomial Chebyshev approximations, $\bar{P}_{k,M-1}(\lambda), k = 0, 1, \ldots, K$, with $M = 80$. The dotted horizontal line designates $\sum_{k=0}^{K} \bar{P}_{k,M-1}(\lambda)$, which is close to 1 in all considered approximations, thus guaranteeing stable transform invertibility. Transfer function $H_6(\lambda)$ and approximations, $\bar{P}_{6,M-1}(\lambda)$, are designated by the thick black line.

be observed that for different values of M, the summations in all considered cases are very close to 1, thus guaranteeing numerically stable invertibility of the LGFT, as discussed later.

The so obtained approximations of transfer functions, $H_k(\lambda)$, are used for the LGFT based vertex-frequency analysis. Absolute LGFT values, calculated for the three-component graph signal from Figure 8.1(d), are shown in Figures 8.9(a)–(c), for $M = 6$, $M = 20$ and $M = 80$. Low resolution in Figure 8.9(a) is directly related to the imprecise and very wide (with a low spectral resolution) approximation of the spectral transfer functions for $M = 6$, in Figure 8.8(b). Notice that high values of the polynomial order, $(M - 1)$, increase calculation complexity and require wide vertex neighborhood in the calculation of the LGFT.

Based on the analysis of calculation complexity in Section 3.5 we may conclude that an order of $KMN_{\mathbf{L}}$ of arithmetic operations is needed to calculate the LGFT in the vertex domain, with $(K + 1)$ spectral bands, using a polynomial whose order is $(M - 1)$. The number of nonzero elements in the graph Laplacian is denoted by $N_{\mathbf{L}}$.

The Spectral Graph Wavelet Transform

Several attempts have been made to extend the classical wavelet analysis to general graph signals, some of which were performed on specific tree graphs (Lee *et al.*, 2008; Murtagh, 2007). The most significant attempts to define the wavelet transform on general graphs have been: (i) a lifting-based approach for multi-scale representation of graph signals (Jansen *et al.*, 2009; Narang and Ortega, 2009; Rustamov and Guibas, 2013), (ii) diffusion-based wavelets and diffusion based polynomial frames (Coifman and Lafon, 2006; Maggioni and Mhaskar, 2008), and (iii) separable filter-bank wavelets (Narang and Ortega, 2012). The wavelet definition that can be directly related to the presented spectral domain local graph Fourier transform, and has been commonly used in the graph signal analysis, is based on the extension of the spectral domain form of the classical wavelet transform and its polynomial approximations, and was introduced in Hammond *et al.* (2011).

In classical signal processing theory, time-frequency analysis has many common goals with the wavelet transform (and its generalization

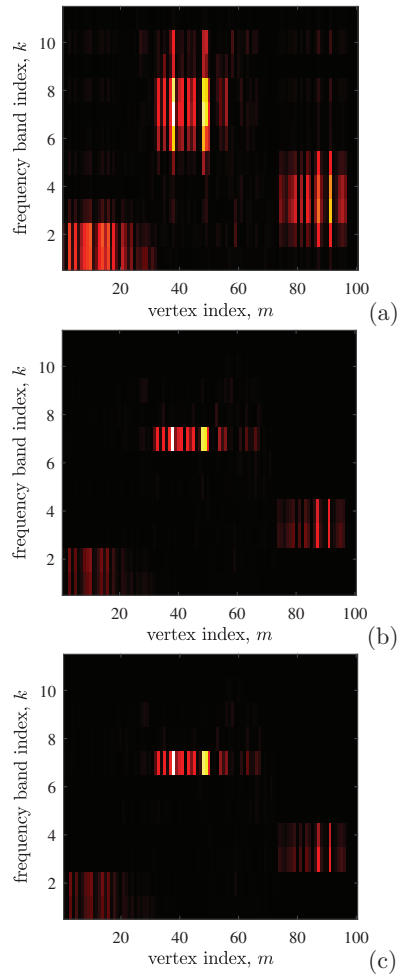


Figure 8.9: Vertex-frequency representation of a three-component signal in Figure 8.1(d). The LGFT is based on raised cosine (Hann window) like band-pass transfer functions for frequency selection, with $K = 10$, approximated using the Chebyshev polynomials of various order, as shown in Figures 8.8(b)–(d). (a) The LGFT of the signal from Figure 8.1(d), calculated using the Chebyshev polynomial approximation of transfer functions given in Figure 8.8(b), with $M = 6$. (b) The LGFT of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.4(c), with $M = 20$. (c) The LGFT of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.4(d), with $M = 80$. Low resolution in (a) can be directly related with low $M = 6$ used in approximation in Figure 8.8(b). The resolution is considerably improved for $M = 20$.

in the form of time-scale analysis). However, these two areas are usually considered separately. The main goals of the wavelet analysis are to perform multi-resolution signal analysis, compression, and signal processing, including the wavelet domain sparsity-driven signal denoising. The main goals in classical time-frequency analysis are in spectral and signal parameter estimation (like, for example, the instantaneous frequency), joint time-frequency domain processing, detection, and denoising of nonstationary signals.

Since the same relation between these areas can be assumed for graph signal processing, we shall consider only the spectral wavelet transform, which is directly related to the frequency-varying LGFT and can be considered as a special case of the frequency-varying vertex-frequency analysis, rather than a transform aimed at graph signal compression and its wavelet-like multi-resolution analysis.

The classic wavelet analysis is based on defining the "mother wavelet" and using its dilatated and translated versions to create signal decomposition kernels. A direct extension of this concept is not possible on graphs as irregular signal domains, since the operations of dilatation and translation are not possible in the same way as in the case of simple regularly sampled line as the signal domain. As in classical signal processing, wavelet coefficients can be defined as a *projection of a graph signal onto the wavelet kernel functions*. Assume that the basic form for the wavelet definition in the spectral domain is $H(\lambda_p)$. The wavelet in spectral domain then represents a scaled version of $H(\lambda_p)$ in the scale s_i, $i = 1, 2, \ldots, K-1$, and is denoted by Hammond *et al.* (2019), Behjat and Van De Ville (2019), Behjat *et al.* (2015), Rustamov and Guibas (2013), Jestrović *et al.* (2017), Masoumi *et al.* (2019), and Cioacă *et al.* (2019)

$$H_i(\lambda_p) = H(s_i\lambda_p).$$

Additionally, a low-pass scale (father wavelet) function $G(\lambda_p)$, plays the role of low-pass function, $H_0(\lambda_p)$, in the LGFT. Therefore, a set of discrete scales for the wavelet calculation, denoted by $s \in \{s_1, s_2, \ldots, s_{K-1}\}$, is assumed with the corresponding spectral transfer functions, $H_{s_i}(\lambda_p)$ and $G(\lambda_p)$. Now, in the same way as in the case of the kernel form of the LGFT in (8.8), the graph wavelet transform is

defined using the band-pass scaled wavelet kernel, $\psi_{m,s_i}(n)$, instead of the LGFT kernel, $\mathcal{H}_{m,k}(n)$, in (8.14). This yields

$$\psi_{m,s_i}(n) = \sum_{p=0}^{N-1} H(s_i\lambda_p)u_p(m)u_p(n), \tag{8.27}$$

which corresponds to the LGFT kernel, $\mathcal{H}_{m,k}(n)$, defined in (8.16). This yields the wavelet coefficients given by

$$\begin{aligned} W(m,s_i) &= \sum_{n=0}^{N-1} \psi_{m,s_i}(n)x(n) \\ &= \sum_{n=0}^{N-1}\sum_{p=0}^{N-1} H(s_i\lambda_p)x(n)u_p(m)u_p(n) = \sum_{p=0}^{N-1} H(s_i\lambda_p)X(p)u_p(m). \end{aligned}$$

The wavelet coefficients may be interpreted as the IGFT of $H(s_i\lambda_p)X(p)$, that is

$$W(m,s_i) = \text{IGFT}\{H(s_i\lambda_p)X(p)\}. \tag{8.28}$$

Remark 38: We will use the notation $H(s_i\lambda) = H_i(\lambda)$ with the corresponding matrix function form $H_i(\mathbf{\Lambda})$. Notice that this scale-based indexing is opposite to the classical frequency band indexing. The largest scale for $H(s_1\lambda)$, $1 < s_1\lambda \leq M$, is obtained for the smallest s_1, $1/s_1 < \lambda \leq M/s_1$, where $M > 1$ is the coefficient of the scale changes, which will be explained later. The associated spectral wavelet transfer function, $H(s_1\lambda) = H_1(\lambda)$, corresponds to the highest frequency band. The wavelet transfer function in scale s_K, $H(s_K\lambda) = H_K(\lambda)$, is associated with the lowest frequency band. Notation for the spectral scale function (low-pass transfer function complementary to $H(s_K\lambda)$ within the lowest spectral interval) is $G(\lambda)$. The spectral scale function, $G(\lambda)$, plays the role of low-pass transfer function with spectral index 0 in the LGFT. Therefore, K spectral wavelet transfer functions $H(s_i\lambda)$, $i = 1, 2, \ldots, K$, along with the scale function $G(\lambda)$, cover exactly $K + 1$ spectral bands as in the LGFT case.

According to (3.36), we can write

$$\mathbf{w}_i = H_i(\mathbf{L})\mathbf{x}, \tag{8.29}$$

where $\mathbf{w}_i$ is a column vector with elements $W(m, s_i)$, $m = 0, 1, \ldots, N-1$.

If $H_i(\lambda) = H(s_i\lambda)$ can be approximated by a polynomial in λ, $H_i(\lambda) \approx P_i(\lambda)$, then the relation

$$\mathbf{w}_i \approx P_i(\mathbf{L})\mathbf{x}, \tag{8.30}$$

follows, where $P_i(\mathbf{L})$ is a polynomial in the graph Laplacian (see Section 3.5 and Example 22).

Example 23: The wavelet transform (vertex-scale) representation of a three-component signal in Figure 8.1(d), obtained using the Meyer-like graph wavelet in the spectral domain, λ, will be illustrated here. As in classical wavelet transform, the wavelet in the first scale should correspond to the high-pass transfer function with nonzero values in the interval $\lambda_{\max}/M < \lambda \leq \lambda_{\max}$, where $M > 1$ is the coefficient of the scale changes. In classical wavelet transforms the dyadic scheme with $M = 2$ is commonly used. *The scale based indexing is opposite to the classical frequency indexing, where large indices indicate the high frequency content.* The Meyer-like graph wavelet in the first scale is defined by Meyer (1992) and Leonardi and Van De Ville (2013)

$$H(s_1\lambda) = \begin{cases} \sin\left(\dfrac{\pi}{2} v_x(q(s_1\lambda - 1))\right), & \text{for } 1 < s_1\lambda \leq M, \\ 0, & \text{elsewhere.} \end{cases}$$

For $2 \leq i \leq K$ the Meyer-like graph wavelet is given by

$$H(s_i\lambda) = \begin{cases} \sin\left(\dfrac{\pi}{2} v_x(q(s_i\lambda - 1))\right), & \text{for } 1 < s_i\lambda \leq M \\ \cos\left(\dfrac{\pi}{2} v_x\left(q\left(\dfrac{s_i\lambda}{M} - 1\right)\right)\right), & \text{for } M < s_i\lambda \leq M^2 \\ 0, & \text{elsewhere,} \end{cases}$$

where $q = 1/(M-1)$. The initial interval is defined by $s_1 = M/\lambda_{\max}$, so that $1 < s_i\lambda \leq M$ corresponds to $\lambda_{\max}/M < \lambda \leq \lambda_{\max}$, while the other interval bounds are defined using a geometric sequence of scale factors,

$$s_i = s_{i-1}M = s_1 M^{i-1} = \frac{1}{\lambda_{\max}} M^i.$$

Observe that the larger the scale factor s_i (and the scale index i), the narrower the transfer function, $H(s_i\lambda)$, while the progression coefficient is

$$M = (q+1)/q > 1.$$

In classical wavelet transforms the dyadic scheme with $M = 2$ is commonly used. The last value of the scale factor, $s_K = M^K/\lambda_{\max}/M$, is defined by K and indicates how close the last wavelet transfer function is to $\lambda = 0$.

The polynomial function, $v_x(x)$, is defined by

$$\begin{aligned} &v_x(x) = x^4(35 - 84x + 70x^2 - 20x^3), \quad \text{for } 0 \le x \le 1, \text{ with} \\ &v_x(q(0)) = v_x(0) = 0, \quad v_x(q(M-1)) = v_x(1) = 1. \end{aligned} \tag{8.31}$$

The wavelet transfer functions,

$$H_i(\lambda) = H(s_i\lambda),$$

are of a band-pass type. The main property (condition for the reconstruction) is that the wavelet functions in two successive scales satisfy the following property

$$\begin{aligned} &H_i^2(\lambda) + H_{i+1}^2(\lambda) \\ &\quad = \cos^2\left(\frac{\pi}{2}v_x\left(q\left(\frac{s_i\lambda}{M} - 1\right)\right)\right) + \sin^2\left(\frac{\pi}{2}v_x\left(q\left(\frac{s_i\lambda}{M} - 1\right)\right)\right) = 1, \end{aligned}$$

within

$$M < s_i\lambda \le M^2.$$

This property implies $\sum_{i=1}^{K} H^2(s_i\lambda) = 1$ for all λ except in the last interval, $s_K\lambda \in [0, M^2]$. To handle the low-pass spectral components (the interval for λ closest to $\lambda = 0$), the low-pass type *scale function*, $G(\lambda)$), is added in the form

$$G(\lambda) = \begin{cases} 1, & \text{for } 0 \le \lambda \le M/s_K = \lambda_{\max}/M^{K-1} \\ \cos\left(\frac{\pi}{2}v_x\left(q\left(\frac{s_K\lambda}{M} - 1\right)\right)\right), & \text{for } M < s_K\lambda \le M^2 \\ 0, & \text{elsewhere.} \end{cases}$$

Remark 39: The number of wavelet transfer functions, K, does not depend on the other wavelet parameters. A large value of K will only increase the number of intervals and the resolution (producing smaller width of the first interval defined by $\lambda_{\max}/M^{K-1}$) toward $\lambda \to 0$, as shown in Figures 8.10(a)–(c).

Remark 40: The wavelet transfer functions, $H(s_i\lambda)$, including the low-pass scale function, $G(\lambda)$, defined in Example 23 satisfy the relation

$$\sum_{i=1}^{K} H^2(s_i\lambda) + G^2(\lambda) = 1.$$

Example 24: For $q = 1$, $M = 2$, and $K = 9$ the Meyer wavelet functions are given in Figure 8.10(a). The Meyer wavelet functions for $q = 3$, $M = 4/3$, $K = 13$ and $q = 9$, $M = 10/9$, $K = 45$ are shown in Figures 8.10(b) and (c). The vertex-frequency representation of the signal from Figure 8.1 using these three sets of wavelet transfer functions are shown in Figures 8.11(a)–(c).

Polynomial SGWT approximation. Chebyshev approximation of the wavelet functions, $H(s_i\lambda) = H_i(\lambda)$, in the form

$$\bar{P}_{i,M-1}(\lambda) = \frac{c_{i,0}}{2} + \sum_{m=1}^{M-1} c_{i,m}\bar{T}_m(\lambda), \tag{8.32}$$

can be used for the vertex domain wavelet transform implementation

$$\bar{P}_{i,M-1}(\mathbf{L}) = \frac{c_{i,0}}{2} + \sum_{m=1}^{M-1} c_{i,m}\bar{T}_m(\mathbf{L}), \quad i = 0, 1, 2, \ldots, K$$

using only the $(M-1)$-neighborhood of each considered vertex, and without any graph Laplacian eigendecomposition analysis. The Chebyshev polynomials can be calculated recursively, as in (3.28), with a change of variables and the recursive implementation as described in detail in Examples 5 and 22.

Windows Defined Using the Vertex Neighborhood

In order to show that the window, $h_m(n)$, which is localized at a vertex m can also be defined using the vertex neighborhood, recall that the

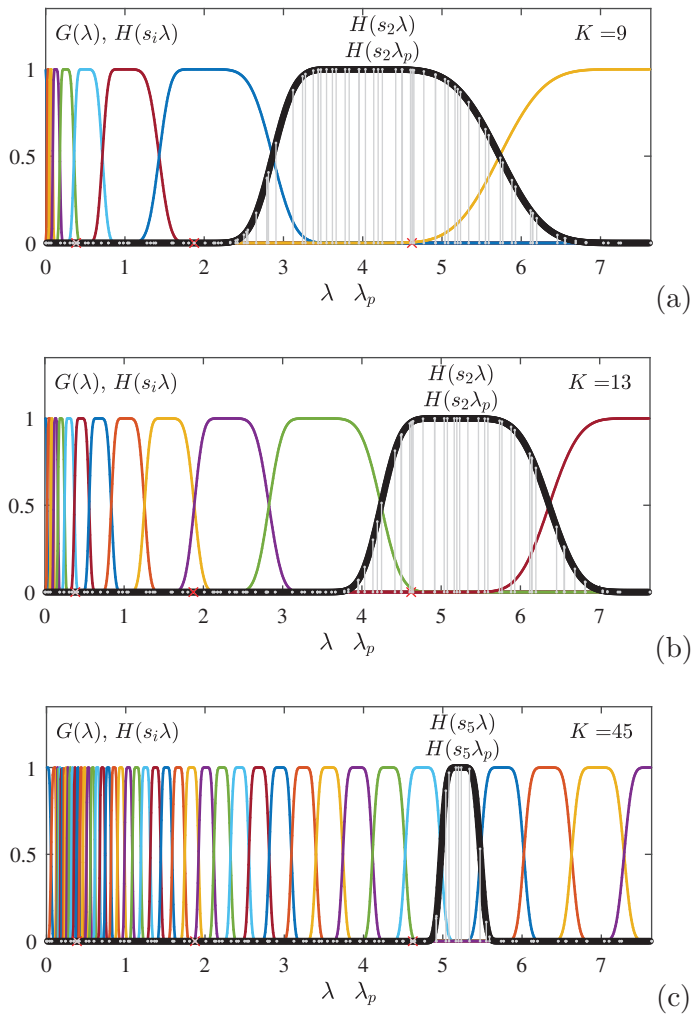


Figure 8.10: Exemplars of Meyer wavelet functions (acting as transfer functions in the wavelet transform), shown in the spectral domain. (a) Band-pass Meyer wavelet functions $H(s_i\lambda)$, $i = 1, 2, \ldots, K$ and the low-pass scale function $G(\lambda)$, for $K = 9$ and $M = 2$. (b) Band-pass Meyer wavelet functions $H(s_i\lambda)$, $i = 1, 2, \ldots, K$ and the low-pass scale function $G(\lambda)$, for $K = 13$ and $M = 3/2$. (c) Band-pass Meyer wavelet functions $H(s_i\lambda)$, $i = 0, 1, \ldots, K$ and the low-pass function $G(\lambda)$, for $K = 45$ and $M = 10/9$. Transfer functions $H(s_2\lambda)$, $H(s_2\lambda)$, $H(s_5\lambda)$ are designated by the thick black line, for each of the considered setups in (a)–(c), respectively; their values at λ_p are shown in gray.

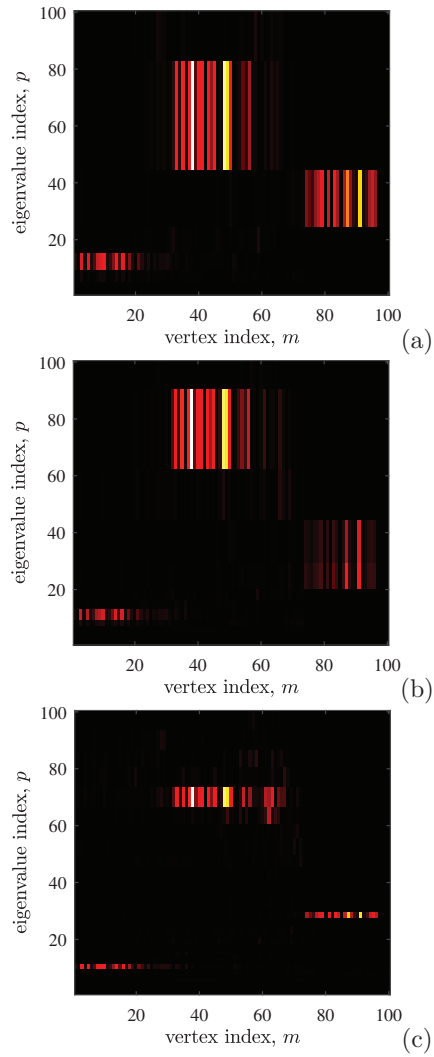


Figure 8.11: Vertex-frequency representation of a three-component signal in Figure 8.1(d). (a) The Meyer wavelet transform of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.10(a). (b) The Meyer wavelet transform of the signal from Figure 8.1(d), calculated using the transfer functions for frequency selection given in Figure 8.10(b). (c) The Meyer wavelet transform of the signal from Figure 8.1(d), calculated using the Meyer wavelet transform transfer functions for frequency selection given in Figure 8.10(c). Wavelet values were reassigned to spectral indices, p, in order to illustrate the change in resolution. The same value of SGWT, $W(m,k)$, is assigned to each spectral index, p, when $\lambda_p \in (\frac{a_k+b_k}{2}, \frac{b_k+c_k}{2}]$, without any scaling.

distance, d_{mn}, between vertices m and n is equal to the length of the shortest walk from vertex m to vertex n, and that d_{mn} takes integer values. Then, the window function can be defined as a function of vertex distance, in the form

$$h_m(n) = g(d_{mn}),$$

where $g(d)$ corresponds to any basic window function in classical signal processing. For example, we can use the Hann window, given by

$$h_m(n) = \frac{1}{2}(1 + \cos(\pi d_{mn}/D)), \quad \text{for } 0 \leq d_{mn} < D,$$

where D is the assumed window width.

For convenience, window functions for every vertex can be calculated in a matrix form as follows:

- For the vertices for which the distance is $d_{mn} = 1$, window functions are defined through an adjacency (neighborhood one) matrix $\mathbf{A}_1 = \mathbf{A}$. In other words, the vertices which belong to the one-neighborhood of a vertex, m, are indicated by unit-value elements in the mth row of the adjacency matrix $\mathbf{A}$ (in unweighted graphs). In weighed graphs, the corresponding adjacency matrix $\mathbf{A}$ can be obtained from the weighting matrix $\mathbf{W}$ as $\mathbf{A} = \text{sign}(\mathbf{W})$.

- Window functions for vertices m and n, for which the distance is $d_{mn} = 2$ are defined by the matrix

  $$\mathbf{A}_2 = (\mathbf{A} \odot \mathbf{A}_1) \circ (\mathbf{1} - \mathbf{A}_1) \circ (\mathbf{1} - \mathbf{I}),$$

  where the symbol $\odot$ denotes the logical (Boolean) matrix product, $\circ$ is the Hadamard (element-by-element) product, and $\mathbf{1}$ is a matrix with all elements equal to 1. The nonzero elements of the mth row of the matrix $\mathbf{A} \odot \mathbf{A}_1$ then designate the vertices that are connected to the vertex m with walks of length $K = 2$ or lower. It should be mentioned that the element-by-element multiplication of $(\mathbf{A} \odot \mathbf{A}_1)$ by matrix $(\mathbf{1} - \mathbf{A}_1)$ removes the vertices connected with walks of length 1, while the multiplication by $(\mathbf{1} - \mathbf{I})$ removes the diagonal elements from $(\mathbf{A} \odot \mathbf{A}_1)$.

- For $d_{mn} = d \geq 2$, we arrive at a recursive relation for the calculation of a matrix which will give the information about the vertices

separated by the distance d. Such a matrix has the form

$$\mathbf{A}_d = (\mathbf{A} \odot \mathbf{A}_{d-1}) \circ (\mathbf{1} - \mathbf{A}_{d-1}) \circ (\mathbf{1} - \mathbf{I}). \tag{8.33}$$

The window matrix for an assumed graph window width, D, can now be defined as

$$\mathbf{P}_D = g(0)\mathbf{I} + g(1)\mathbf{A}_1 + \cdots + g(D-1)\mathbf{A}_{D-1},$$

so that a graph signal which is localized around a vertex m, may be formed based on this matrix, as

$$x_m(n) = h_m(n)x(n) = P_D(n,m)x(n).$$

The LGFT representation of a graph signal, $x(n)$, then becomes

$$S(m,k) = \sum_{n=0}^{N-1} x(n)h_m(n)\ u_k(n) = \sum_{n=0}^{N-1} x(n)P_D(n,m)\ u_k(n), \tag{8.34}$$

with the vertex-frequency kernel given by

$$\mathcal{H}_{m,k}(n) = h_m(n)u_k(n) = P_D(n,m)u_k(n). \tag{8.35}$$

This allows us to arrive at the matrix form of the LGFT, given by

$$\mathbf{S} = \mathbf{U}^T(\mathbf{P}_D \circ [\mathbf{x},\ \mathbf{x},\ \ldots,\ \mathbf{x}]), \tag{8.36}$$

where $[\mathbf{x},\ \mathbf{x},\ \ldots,\ \mathbf{x}]$ is an $N \times N$ matrix, the columns of which are the signal vector, $\mathbf{x}$.

For a rectangular function $g(d) = 1$, for $d < D$, the LGFT can be calculated recursively with respect to the window width, D, as

$$\mathbf{S}_D = \mathbf{S}_{D-1} + \mathbf{U}^T(\mathbf{A}_{D-1} \circ [\mathbf{x},\ \mathbf{x},\ \ldots,\ \mathbf{x}]). \tag{8.37}$$

Example 25: Consider the local vertex-frequency representation of the signal from Figure 8.1, using vertex domain defined windows. The localization kernels, $\mathcal{H}_{m,k}(n) = h_m(n)u_k(n)$, are shown in Figure 8.12 for two vertices and two spectral indices. Observe that for the spectral index $k = 0$, the localization kernel is proportional to the localization function $h_m(n)$, given in Figures 8.12(a) and (c) for the vertices $m = 34$ and $m = 78$. Frequency modulated forms of these localization functions are shown in Figures 8.12(b) and (d), for the same vertices and $k = 20$.

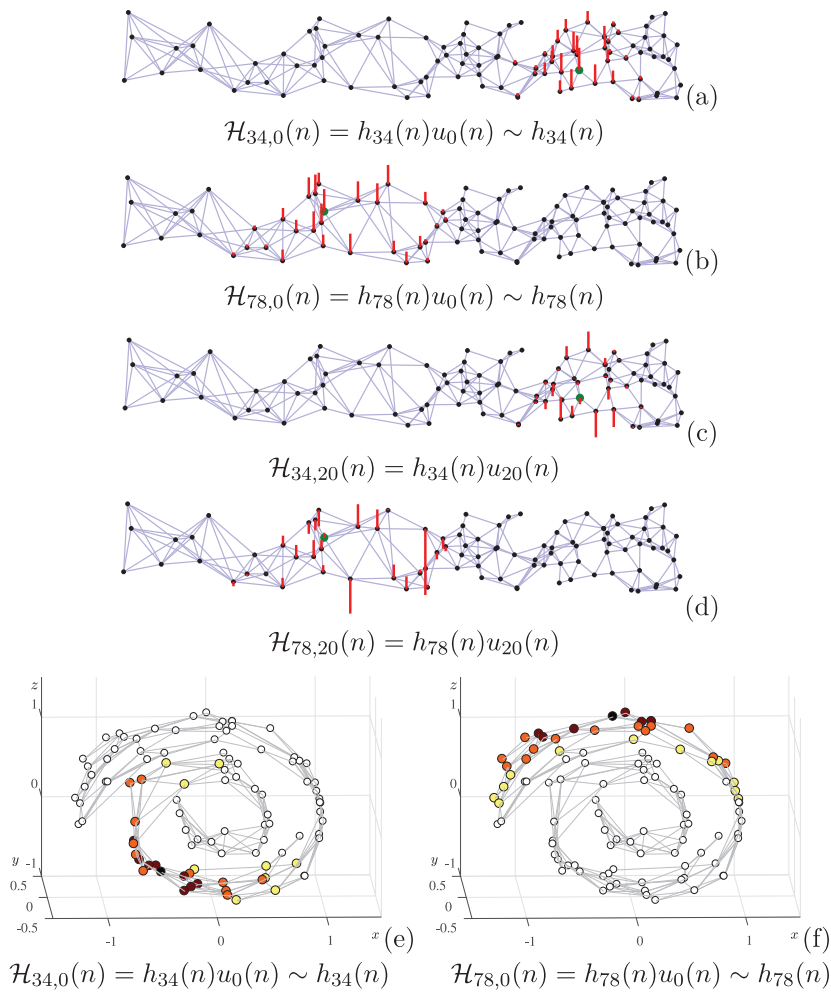


Figure 8.12: Localization kernels for vertex-frequency analysis, $\mathcal{H}_{m,k}(n) = h_m(n)u_k(n)$, for the case of *vertex domain defined windows* in the local graph Fourier transform, $S(m,k) = \sum_{n=0}^{N-1} x(n)\mathcal{H}_{m,k}(n)$. (a) Localization kernel $\mathcal{H}_{34,0}(n) = h_{34}(n)u_0(n) \sim h_{34}(n)$, for a constant eigenvector, $u_0(n) = 1/\sqrt{N}$, centered at the vertex $m = 34$. (b) The same localization kernel as in (a), but centered at the vertex $m = 78$. (c) Localization kernel, $\mathcal{H}_{34,20}(n) = h_{34}(n)u_{20}(n)$, centered at the vertex $m = 35$ and frequency shifted by $u_{20}(n)$. Observe the variations in kernel amplitude, which indicate a modulation of the localization window, $h_m(n)$. (d) The same localization kernel as in (c), but centered at the vertex $m = 78$. (e) Three-dimensional representation of the kernel $\mathcal{H}_{34,0}(n) = h_{34}(n)u_0(n)$. (f) Three-dimensional representation of the kernel $\mathcal{H}_{78,0}(n) = h_{78}(n)u_0(n)$.

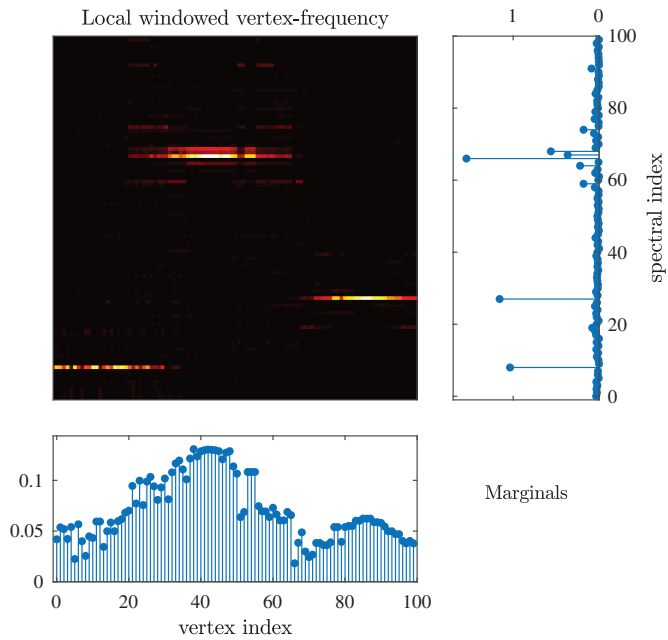


Figure 8.13: Local vertex-frequency spectrum calculated using the LGFT and vertex neighborhood windows, as in (8.35). This representation immediately shows that the graph signal consists of three components located at spectral indices $k = 8$, $k = 66$, and $k = 27$, with the corresponding vertex indices in their respective vertex subsets $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$, where $\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{V}_3 = \mathcal{V}$. The marginal properties are also given in the panels to the right and below the vertex-frequency representation, and they differ from the ideal ones given respectively by $|x(n)|^2$ and $|X(k)|^2$.

A vertex domain window is next used to analyze the graph signal from Figure 8.1. The vertex-frequency representation, $S(n, k)$, obtained with the LGFT and the vertex domain localization window is given in Figure 8.13. Again, we can observe three constituent graph signal components in three distinct vertex regions. The marginals of $S(n, k)$ are also shown in the right and bottom panels.

Remark 41: **Directed graphs.** The vertex neighborhood, as a set of vertices that can be reached from the considered vertex by a walk whose length is at most D, may be also defined on directed graphs. In this case, this approach corresponds to one-sided windows in classical signal analysis.

If we want to define two-sided window, then we should also include all vertices from which we can reach the considered vertex by walk whose length is at most D. This means that for a directed graph we should assume that vertices with distance $d_{mn} = 1$ from the considered vertex m are the vertices from which we can reach vertex m with walk of length 1. In this case $\mathbf{A}_1 = \mathbf{A} + \mathbf{A}^T$ where addition is logical operation (Boolean OR). The matrix $\mathbf{A}_2$ is

$$\mathbf{A}_2 = (\mathbf{A} \odot \mathbf{A} + \mathbf{A}^T \odot \mathbf{A}^T) \circ (\mathbf{1} - \mathbf{I}) \circ (\mathbf{1} - \mathbf{A}_1).$$

This procedure could be continued for walks up to the desired maximal length D.

For a circular directed graph in this way, we will get the classical STFT with symmetric window.

Window Parameter Optimization

The concentration of local vertex spectrum representation can be measured using the normalized one-norm (Stanković, 2001), as

$$\mathcal{M} = \frac{1}{F} \sum_{m=0}^{N-1} \sum_{k=0}^{N-1} |S(m,k)| = \frac{1}{F} \|\mathbf{S}\|_1, \tag{8.38}$$

where

$$F = \|\mathbf{S}\|_F = \sqrt{\sum_{m=0}^{N-1} \sum_{k=0}^{N-1} |S(m,k)|^2}$$

is the Frobenius norm of matrix $\mathbf{S}$. Alternatively, any other norm $\|\mathbf{S}\|_p^p$, with $0 \leq p \leq 1$ can be used instead of $\|\mathbf{S}\|_1$. Recall that norms with p close to 0 are noise sensitive, while the norm with $p = 1$ is the only convex norm, which hence allows for gradient based optimization (Stanković, 2001).

Example 26: The concentration measure, $\mathcal{M}(\tau) = \|\mathbf{S}\|_1 / \|\mathbf{S}\|_F$, for the signal from Figure 8.1, the window given in (8.2), and for various τ is shown in Figure 8.14, along with the optimal vertex frequency representation. This representation is similar to that shown in Figure 8.3, where an empirical value of $\tau = 3$ was used, with the same localization window and kernel form.

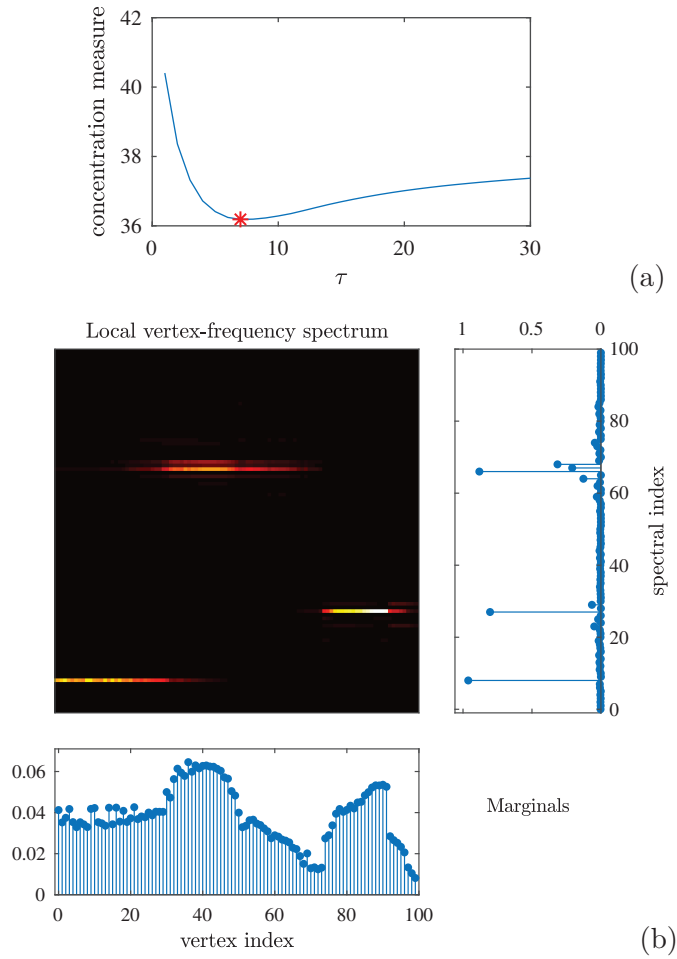


Figure 8.14: Principle of the optimization of localization window. (a) Measure of the concentration of graph spectrogram for a varying spectral domain window parameter τ. (b) The corresponding optimal vertex-frequency representation, calculated with $\tau = 7$, together with its marginals.

The optimal τ can be obtained in only a few steps through the iteration

$$\tau_k = \tau_{k-1} - \alpha(\mathcal{M}(\tau_{k-1}) - \mathcal{M}(\tau_{k-2})),$$

with α a step-size parameter.

The optimization of parameter τ can also be achieved through graph uncertainty principle based techniques (Agaskar and Lu, 2013; Tsitsvero *et al.*, 2016).

8.2 Inversion of the LGFT

The inversion relation of the LGFT, calculated using any of the presented localization (window) forms, will next be considered in a unified way; the two approaches for the LGFT inversion here are: (i) inversion by summation of LGFT and (ii) kernel based inversion.

Inversion by the Summation of the LGFT

The reconstruction of a graph signal, $x(n)$, from its local spectrum, $S(m,k)$, can be performed through an inverse GFT of (8.5), based on the graph windowed signal

$$x(n)h_m(n) = \sum_{k=0}^{N-1} S(m,k)\, u_k(n) \tag{8.39}$$

followed by a summation over all vertices, m, to yield

$$x(n) = \frac{1}{\sum_{m=0}^{N-1} h_m(n)} \sum_{m=0}^{N-1}\sum_{k=0}^{N-1} S(m,k)u_k(n). \tag{8.40}$$

Remark 42: If the windows, $h_m(n)$, for every vertex, n, satisfy the condition

$$\sum_{m=0}^{N-1} h_m(n) = 1,$$

then the reconstruction does not depend on the vertex index, n, or in other words such reconstruction is vertex independent. This becomes clear from

$$x(n) = \sum_{m=0}^{N-1}\sum_{k=0}^{N-1} S(m,k)u_k(n) = \sum_{k=0}^{N-1} X(k)u_k(n), \tag{8.41}$$

where

$$X(k) = \sum_{m=0}^{N-1} S(m,k)$$

is a projection of the LGFT onto the spectral index axis. For windows obtained using the generalized graph shift in (8.33), this conditions is always satisfied since $H(0) = 1$.

The condition $\sum_{m=0}^{N-1} h_m(n) = 1$ can be enforced by normalizing the elements of the matrix $\mathbf{A}_d$, $d = 1, 2, \ldots, D-1$ in (8.33), prior to the calculation of matrix $\mathbf{P}_D$, in such a way that the sum of each of its columns is equal to 1, which allows us to arrive at

$$\sum_{m=0}^{N-1} h_m(n) = \sum_{m=0}^{N-1} P_D(n,m) = \sum_{d=1}^{D-1} g(d) = const.$$

In general, the local vertex spectrum, $S(m,k)$, can also be calculated over a reduced set of vertices, $m \in \mathcal{M} \subset \mathcal{V}$. In this case, the summation over m in the reconstruction formula should be executed over only the vertices $m \in \mathcal{M}$, while a vertex-independent reconstruction is achieved if $\sum_{m\in\mathcal{M}} h_m(n) = 1$.

Inversion of the LGFT with Band-Pass Functions

For the LGFT, defined in (8.18) as $\mathbf{s}_k = \sum_{p=0}^{M-1} h_{p,k}\mathbf{L}^p\mathbf{x}$, the inversion is obtained by a summation over all spectral index shifts, $k = 0, 1, \ldots, K$, that is

$$\sum_{k=0}^{K} \mathbf{s}_k = \sum_{k=0}^{K}\sum_{p=0}^{N-1} h_{p,k}\mathbf{L}^p\mathbf{x} = \sum_{k=0}^{K} H_k(\mathbf{L})\mathbf{x} = \mathbf{x}, \tag{8.42}$$

if $\sum_{k=0}^{K} H_k(\mathbf{L}) = \mathbf{I}$. This condition is equivalent to the following spectral domain form

$$\sum_{k=0}^{K} H_k(\mathbf{\Lambda}) = \mathbf{I} \tag{8.43}$$

since $\mathbf{U}\sum_{k=0}^{K} H_k(\mathbf{\Lambda})\mathbf{U}^T = \mathbf{I}$ and $\mathbf{U}^T\mathbf{U} = \mathbf{I}$. The condition in (8.43) is used to define the transfer functions in Figure 8.4.

Kernel-Based Inversion

Another approach to the inversion of the local vertex spectrum, $S(m,k)$, follows the Gabor expansion framework (Stanković *et al.*, 2014), whereby

the local vertex spectrum, $S(m,k)$, is projected back to the vertex-frequency localized kernels, $\mathcal{H}_{m,k}(n)$. The inversion for two forms of the LGFT, defined as in (8.6) and (8.17), will be analyzed.

(a) For the LGFT defined in (8.6), the sum of all of its projections to the localized kernels, $\mathcal{H}_{m,k}(n)$, is

$$\begin{aligned}
&\sum_{m=0}^{N-1}\sum_{k=0}^{N-1} S(m,k)\mathcal{H}_{m,k}(n) \\
&= \sum_{m=0}^{N-1}\left(\sum_{k=0}^{N-1} S(m,k)h_m(n)u_k(n)\right) \\
&= \sum_{m=0}^{N-1}\left(\sum_{i=0}^{N-1} \underset{k\to i}{\mathrm{IGFT}}\{S(m,k)\}\underset{k\to i}{\mathrm{IGFT}}\{h_m(n)u_k(n)\}\right) \\
&= \sum_{m=0}^{N-1}\sum_{i=0}^{N-1}[x(i)h_m(i)][h_m(n)\delta(n-i)] \\
&= \sum_{m=0}^{N-1} x(n)h_m^2(n) = x(n)\sum_{m=0}^{N-1} h_m^2(n),
\end{aligned} \tag{8.44}$$

where IGFT denotes the inverse GFT transform. Parseval's theorem for graph signals

$$\sum_{n=0}^{N-1} x(n)y(n) = \sum_{k=0}^{N-1} X(k)Y(k)$$

was used in the derivation. In this form of the LGFT all possible spectral shifts, $k = 0, 1, \ldots, N-1$, are used.

The inversion formula for the local vertex spectrum, $S(m,k)$, which yields the original graph signal, $x(n)$, then becomes

$$x(n) = \frac{1}{\sum_{m=0}^{N-1} h_m^2(n)}\sum_{m=0}^{N-1}\sum_{k=0}^{N-1} S(m,k)\mathcal{H}_{m,k}(n). \tag{8.45}$$

Remark 43: This kind of kernel-based inversion is vertex-invariant if the sum over all vertices, m, is invariant with respect to n and is equal to 1, that is

$$\sum_{m=0}^{N-1} h_m^2(n) = 1. \tag{8.46}$$

If the LGFT, $S(m,k)$, is calculated over a reduced set of vertices, $m \in \mathcal{M} \subset \mathcal{V}$, then the vertex independent reconstruction condition becomes $\sum_{m\in\mathcal{M}} h_m^2(n) = 1$.

(b) For the LGFT with spectral shifted spectral windows, defined in (8.17), the kernel based inversion is of the form

$$x(n) = \sum_{m=0}^{N-1} \sum_{k=0}^{K} S(m,k)\mathcal{H}_{m,k}(n) \tag{8.47}$$

if the following condition

$$\sum_{k=0}^{K} H_k^2(\lambda_p) = 1 \tag{8.48}$$

is satisfied for all λ_p, $p = 0, 1, 2, \ldots, N-1$.

The inversion formula in (8.47), with condition (8.48), follows from

$$\begin{aligned}&\sum_{m=0}^{N-1} \sum_{k=0}^{K} S(m,k)\mathcal{H}_{m,k}(n) \\ &\quad = \sum_{m=0}^{N-1} \sum_{k=0}^{K} \sum_{p=0}^{N-1} X(p)H_k(\lambda_p)u_p(m) \sum_{l=0}^{N-1} H_k(\lambda_l)u_l(m)u_l(n).\end{aligned} \tag{8.49}$$

Since $\sum_{m=0}^{N-1} u_p(m)u_l(m) = \delta(p-l)$, the last expression reduces to the graph signal, $x(n)$,

$$\sum_{k=0}^{K} \sum_{p=0}^{N-1} X(p)H_k(\lambda_p)H_k(\lambda_p)u_p(n) = x(n), \tag{8.50}$$

if the transfer functions, $H_k(\lambda_p)$, $k = 0, 1, \ldots, K$, satisfy the condition in (8.48) for all λ_p.

Vertex-Varying Filtering

Filtering in the vertex-frequency domain may be implemented using a vertex-frequency support function, $B(m,k)$. The filtered LGFT is then given by

$$S_f(m,k) = S(m,k)B(m,k),$$

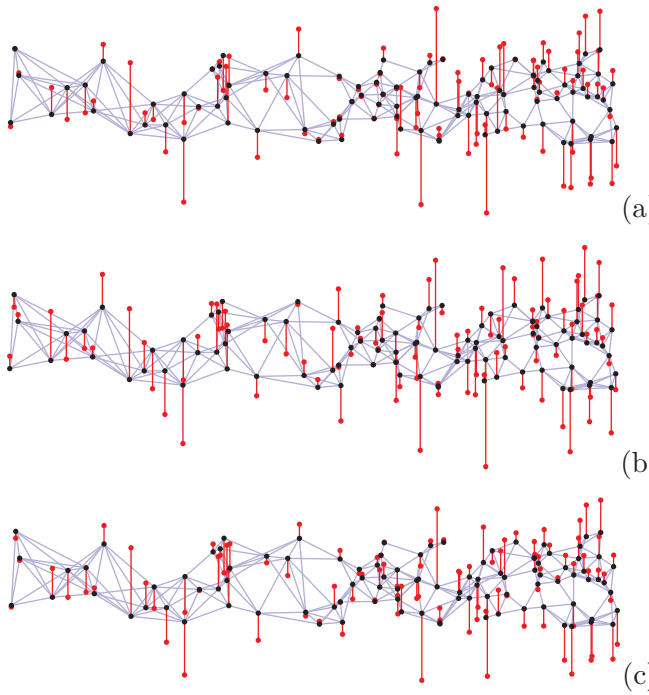


Figure 8.15: Vertex-varying filtering of a graph signal. (a) The original graph signal, $x(n)$, from Figure 8.1 (d). (b) The graph signal, $x(n)$, corrupted by an additive white Gaussian noise, at $\text{SNR}_{in} = 5.3$ dB. (c) The graph signal, $x_f(n)$, after vertex-varying filtering based on thresholding of the LGFT of noisy graph signal, $S(m,k)$, with the final signal-to-noise ratio $\text{SNR}_{out} = 10.36$ dB.

and the filtered signal, $x_f(n)$, is obtained by the inversion of $S_f(m,k)$ using the above mentioned inversion methods. The filtering support function, $B(m,k)$, can be obtained, for example, by thresholding noisy values of the local vertex spectrum, $S(m,k)$.

Example 27: Consider the graph signal, $x(n)$, from Figure 8.1(d), also shown in Figure 8.15(a), and its version corrupted by an additive white Gaussian noise, at the signal-to-noise ratio of $\text{SNR}_{in} = 5.3$ dB, given in Figure 8.15(b). The LGFT, $S(m,k)$ of the noisy graph signal is calculated according to (8.17), using shifted bandpass spectral transfer functions, $H_k(\lambda_p)$, $k = 0, 1, \ldots, K$, $p = 0, 1, \ldots, N-1$, given by (8.20) without squares ($H_k(\lambda_p) \to H_k^2(\lambda_p)$), which allows $\sum_{k=0}^{K} H_k^2(\lambda_p) = 1$ to

hold, instead of $\sum_{k=0}^{K} H_k(\lambda_p) = 1$. In this way, the condition for the inversion (8.48) is satisfied. The transfer functions, $H_k(\lambda_p)$, otherwise correspond to those shown in Figure 8.4(b) with $K = 25$.

The vertex-varying filtering is performed using $S_f(m,k) = S(m,k) \cdot B(m,k)$ for $m = 0,1,\ldots,N-1$, $k = 0,1,\ldots,K$, with a simple thresholding-based filtering support function

$$B(m,k) = \begin{cases} 0, & \text{for } |S(m,k)| < T \\ 1, & \text{otherwise,} \end{cases}$$

$m = 0,1,\ldots,N-1$, $k = 0,1,\ldots,K$, with the threshold $T = 0.09$ set empirically. The output graph signal, $x_f(n)$, is obtained using the inversion relation in (8.47) for the filtered LGFT, $S_f(m,k)$, and shown in Figure 8.15(c). The achieved output SNR was $\text{SNR}_{out} = 10.36$ dB.

If the signal is filtered using the graph Wiener filter, as in Section 7.3, with the estimated noise level $\sigma_\varepsilon = 0.12$, and the available noisy signal, $H(\lambda_k) = |X(k)|^2/(|X(k)|^2 + \sigma_\varepsilon^2)$, with $x_f(n) = \text{IGFT}\{X(k)H(\lambda_k)\}$, then the output SNR is $\text{SNR}_{out} = 7.80$ dB. This value is lower that in the vertex-varying filtering case. If we knew the signal without noise and used it in the definition of the Wiener filter, the output SNR would have been improved to 15.78 dB.

8.3 Uncertainty Principle for Graph Signals

In the classical signal analysis, the purpose of a window function is to enhance signal localization in the joint time-frequency domain. However, the uncertainty principle prevents an ideal localization in both time and frequency. Various forms of the uncertainty principle in the signal analysis have been defined, with surveys in Ricaud and Torrésani (2014) and Perraudin *et al.* (2018). Various forms of the uncertainty principle in graph signal processing are studied in Erb (2019).

These forms are closely related to the concentration measures in time-frequency distributions; for a review see Stanković (2001). While the common uncertainty principle form in time-frequency analysis (whose quantum mechanical form is called the Robertson-Schrödinger inequality) establishes the lower bound for the product of effective signal

widths (variances) in the time and the frequency domain (Cohen, 1995; Stanković, 1997), here we will use a form of the sparsity support measure (Ricaud and Torrésani, 2014; Stanković, 2001) as the one which clearly and in a simple way shows a significant difference in both classical Fourier based analysis and graph signal transforms with respect to the expected concentration in the joint vertex-frequency domain.

In classical signal analysis, the purpose of a window function is to enhance signal localization in the joint time-frequency domain. However, the uncertainty principle prevents the ideal localization in both time and frequency. Indeed, in the classical DFT analysis the uncertainty principle states that

$$\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq N, \tag{8.51}$$

or in other words, that the product of the number of nonzero signal values, $\|\mathbf{x}\|_0$, and the number of its nonzero DFT coefficients, $\|\mathbf{X}\|_0$, is greater or equal than the total number of signal samples N; they cannot simultaneously assume small values.

To arrive at the *uncertainty principle for graph signals*, consider a graph signal, $\mathbf{x}$, and its spectral transform, $\mathbf{X}$, in a domain of orthonormal basis functions, $u_k(n)$. Then, the uncertainty principle states that Tsitsvero *et al.* (2016), Agaskar and Lu (2013), Elad and Bruckstein (2002), and Perraudin *et al.* (2018)

$$\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq \frac{1}{\max_{k,m}\{|u_k(m)|^2\}}. \tag{8.52}$$

This form of the uncertainty principle is generic, and indeed for the basis functions $u_k(n) = \frac{1}{\sqrt{N}}\exp(j2\pi nk/N)$, the standard DFT uncertainty principle form in (8.51) follows. A simple derivation of the support uncertainty principle shall be given in Section 8.5 (Stanković, 2020).

Remark 44: Note, however, that in graph signal processing, the eigenvectors/basis functions can assume quite different forms than in the standard DFT case. For example, when one vertex is loosely connected with other vertices, then $\max\{|u_k(m)|^2\} \to 1$ and even $\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq 1$ is possible for the uncertainty condition in (8.52). *This means that, unlike the classical Fourier transform-based time and frequency domains, a graph signal can be well localized in both the vertex and the spectral domains.*

Example 28: For the graph shown in Figure 8.1, we have

$$\max_{k,m}\{|u_k(m)|^2\} = 0.8713$$

which indicates that even $\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq 1.1478$ is possible. In other words, a graph signal for which the number of nonzero samples, $x(n)$, in the vertex domain is just two, will not violate the uncertainty principle even if it has just one nonzero GFT coefficient, $X(k)$.

8.4 Graph Spectrogram and Frames

Based on (8.5), the *graph spectrogram* can be defined as

$$|S(m,k)|^2 = \left| \sum_{n=0}^{N-1} x(n)h_m(n)u_k(n) \right|^2. \tag{8.53}$$

Then, according to Parseval's theorem, the *vertex marginal property*, which is a projection of $|S(m,k)|^2$ onto the vertex index axis, is given by

$$\begin{aligned}\sum_{k=0}^{N-1} |S(m,k)|^2 &= \sum_{k=0}^{N-1} S(m,k) \sum_{n=0}^{N-1} x(n)h_m(n)\ u_k(n) \\ &= \sum_{n=0}^{N-1} |x(n)h_m(n)|^2,\end{aligned}$$

which would be equal to the signal power, $|x(m)|^2$, at the vertex m, if $h_m(n) = \delta(m-n)$. Since this is not the case, the vertex marginal property of the graph spectrogram is equal to the power of the graph signal in hand, smoothed by the window, $h_m(n)$.

Energy of graph spectrogram. For the total energy of graph spectrogram, we consequently have

$$\sum_{m=0}^{N-1}\sum_{k=0}^{N-1} |S(m,k)|^2 = \sum_{n=0}^{N-1} \left(|x(n)|^2 \sum_{m=0}^{N-1} |h_m(n)|^2 \right). \tag{8.54}$$

If $\sum_{m=0}^{N-1} |h_m(n)|^2 = 1$ for all n, then the spectrogram on the graph is *energy unbiased* (statistically consistent with respect to the energy),

that is

$$\sum_{m=0}^{N-1}\sum_{k=0}^{N-1}|S(m,k)|^2=\sum_{n=0}^{N-1}|x(n)|^2=\|\mathbf{x}\|^2=E_x. \tag{8.55}$$

The LGFT viewed as a frame. A set of functions, $S(m,k)$, is called *a frame* for the expansion of a graph signal, $\mathbf{x}$, if

$$A\|\mathbf{x}\|^2\le\sum_{m=0}^{N-1}|S(m,k)|^2\le B\|\mathbf{x}\|^2,$$

where A and B are positive constants. If $A=B$, the frame is termed *Parseval's tight frame* and the signal can be recovered as

$$x(n)=\frac{1}{A}\sum_{m=0}^{N-1}\sum_{k=0}^{N-1}S(m,k)h_m(n)u_k(n).$$

The constants A and B govern the numerical stability of recovering the original signal $\mathbf{x}$ from the coefficients $S(m,k)$.

The conditions for two forms of the LGFT, defined as in (8.6) and (8.17), to represent frames will be analyzed next.

(a) The LGFT, defined as in (8.6), is a frame, since in this case Parseval's theorem holds (Behjat *et al.*, 2016; Girault, 2015; Hammond *et al.*, 2011; Sakiyama and Tanaka, 2014), that is

$$\sum_{m=0}^{N-1}|h_m(n)|^2=\sum_{k=0}^{N-1}|H(k)|^2|u_k(n)|^2, \tag{8.56}$$

which allows us to write

$$\frac{1}{N}H^2(0)\le\sum_{m=0}^{N-1}|h_m(n)|^2\le\max_{n,k}|u_k(n)|^2\sum_{k=0}^{N-1}|H(k)|^2=\gamma^2E_h, \tag{8.57}$$

where $\gamma=\max_{n,k}|u_k(n)|$ and

$$E_h=\sum_{k=0}^{N-1}|H(k)|^2.$$

By multiplying both sides of the above inequalities by $\|\mathbf{x}\|^2$, we arrive at

$$\frac{1}{N}H^2(0)\|\mathbf{x}\|^2 \leq \sum_{m=0}^{N-1}\sum_{k=0}^{N-1}|S(m,k)|^2 \leq \|\mathbf{x}\|^2\gamma^2 E_h. \tag{8.58}$$

A frame is termed a *tight frame* if the equality in (8.57) holds, that is, if

$$\sum_{m=0}^{N-1}|h_m(n)|^2 = 1,$$

which is the same condition as in (8.46).

(b) The LGFT defined in (8.17) is a tight frame if

$$\sum_{k=0}^{K}\sum_{m=0}^{N-1}|S(m,k)|^2 = \sum_{k=0}^{K}\sum_{p=0}^{N-1}|X(p)H_k(\lambda_p)|^2 = E_x, \tag{8.59}$$

where Parseval's theorem for the $S(m,k)$ as the GFT of $X(p)\cdot H_k(\lambda_p)$ was used to yield

$$\sum_{m=0}^{N-1}|S(m,k)|^2 = \sum_{p=0}^{N-1}|X(p)H_k(\lambda_p)|^2.$$

This means that the LGFT in (8.17) is a tight frame if

$$\sum_{k=0}^{K}|H_k(\lambda_p)|^2 = 1 \quad \text{for } p = 0,1,\ldots,N-1.$$

This condition is used to define transfer functions in Figures 8.4(b) and (c).

From (8.59), it is straightforward to conclude that the graph spectrogram energy is bounded with

$$AE_x \leq \sum_{k=0}^{K}\sum_{m=0}^{N-1}|S(m,k)|^2 \leq BE_x, \tag{8.60}$$

where A and B are respectively the minimum and the maximum of value of

$$g(\lambda_p) = \sum_{k=0}^{K}|H_k(\lambda_p)|^2.$$

Graph Wavelet Transform Inversion

The wavelet inversion formula

$$x(n) = \sum_{n=0}^{N-1} \sum_{i=0}^{K} \psi(n, s_i) W(n, s_i) \tag{8.61}$$

can be derived in the same way and under the same condition as in (8.47)–(8.48), where a set of discrete scales for the wavelet calculation, denoted by $s \in \{s_1, s_1, \ldots, s_K\}$, is assumed, and $\psi(n, s_0)$ is used as a notation for the *scale function*, $\phi(n)$, whose spectral transfer function is $G(\lambda)$, as explained in Remark 38. In the same way as in the LGFT case, it can be shown that the wavelet transform represents a frame with

$$A\|\mathbf{x}\|^2 \leq \sum_{n=0}^{N-1} \sum_{i=0}^{K} |W(n, s_i)|^2 \leq B\|\mathbf{x}\|^2, \tag{8.62}$$

where (Leonardi and Van De Ville (2013), Hammond *et al.* (2019), and Behjat and Van De Ville (2019))

$$A = \min_{0 \leq \lambda \leq \lambda_{\max}} g(\lambda),$$
$$B = \max_{0 \leq \lambda \leq \lambda_{\max}} g(\lambda),$$

and the function $g(\lambda)$ is defined by

$$g(\lambda) = \sum_{i=1}^{K} H^2(s_i \lambda) + G^2(\lambda).$$

The low-pass scale function, $G(\lambda)$, is added in the reconstruction formula, since all $H(s_i\lambda) = 0$ for $\lambda = 0$, as explained in Example 23 and Remark 38. It should be mentioned that the spectral functions of the wavelet transform, $H(s_i\lambda)$, form Parseval's frame if

$$g(\lambda) = 1.$$

Since the number of wavelet transform coefficients, $W(n, s_i)$, for each n and i, is greater than the number of signal samples, N, this representation is redundant, and this redundancy allows us to implement the transform through a fast algorithm, rather than using the explicit

computation of all wavelet coefficients (Behjat and Van De Ville, 2019; Hammond *et al.*, 2019). Indeed, for large graphs, it can be computationally too complex to compute the full eigendecomposition of the graph Laplacian. A common way to avoid this computational burden is to use a polynomial approximation schemes for $H(s_i\lambda)$, $i = 1, 2, \ldots, K$, and $G(\lambda)$. One such approach is the truncated Chebyshev polynomial approximation method which is based on the application of the continuous spectral window functions with Chebyshev polynomials, which admit order-recursive calculation (see Section 3.5 and Example 3.5). If, for a given scale, s_i, the wavelet function is approximated by a polynomial in the Laplacian, $P_i(\mathbf{L})$, then the wavelet transform can be efficiently calculated using

$$\mathbf{w}_i = P_i(\mathbf{L})\mathbf{x}, \tag{8.63}$$

where $\mathbf{w}_i$ a column vector with elements $W(m, s_i)$, $m = 0, 1, \ldots, N-1$. Note that this form corresponds to the LGFT form in (8.18).

8.5 Vertex-Frequency Energy Distributions

Like in time-frequency analysis, the distribution of graph signal energy, as a function of the vertex and spectral indices, is an alternative way to approach vertex-frequency analysis without localization windows. A graph form of the Rihaczek distribution is used as the basic distribution to introduce the concepts of vertex-frequency domain energy parameters, such as the local smoothness and marginal properties. The graph Rihaczek distribution is then used to derive the *support uncertainty principle* and to define a class of reduced interference vertex-frequency energy distributions which satisfy the graph signal marginal properties.

The energy of a general signal is usually defined as

$$E = \sum_{n=0}^{N-1} x^2(n) = \sum_{n=0}^{N-1} x(n) \sum_{k=0}^{N-1} X(k)u_k(n).$$

This expression can be rearranged into

$$E = \sum_{n=0}^{N-1}\sum_{k=0}^{N-1} x(n)X(k)u_k(n) = \sum_{n=0}^{N-1}\sum_{k=0}^{N-1} E(n,k), \tag{8.64}$$

where for each vertex, the vertex-frequency energy distribution, $E(n,k)$, is defined by Stanković *et al.* (2018b, 2019b)

$$E(n,k) = x(n)X(k)u_k(n) = \sum_{m=0}^{N-1} x(n)x(m)u_k(m)u_k(n). \tag{8.65}$$

Remark 45: The definition in (8.65) corresponds to the Rihaczek distribution in classical time-frequency analysis (Boashash, 2015; Cohen, 1995; Stanković *et al.*, 2014). Observe that based on the Rihaczek distribution and the expression in (8.65), we may obtain a vertex-frequency representation even without a localization window. This very important property is also the main advantage (along with the concentration improvement) of classical time-frequency distributions with respect to the spectrogram and STFT based time-frequency representations.

The marginal properties of the vertex-frequency energy distribution, $E(n,k)$, are defined as its projections onto the spectral index axis, k, and the vertex index axis, n, to give

$$\sum_{n=0}^{N-1} E(n,k) = |X(k)|^2 \quad \text{and} \quad \sum_{k=0}^{N-1} E(n,k) = x^2(n),$$

which correspond respectively to the squared spectra, $|X(k)|^2$, and the signal power, $x^2(n)$, of the graph signal, $x(n)$.

Example 29: Figure 8.16 shows the vertex-frequency distribution, $E(n,k)$, of the graph signal from Figure 8.1, together with its marginal properties. The marginal properties are satisfied up to the computer precision. Observe also that the localization of energy is better than in the cases obtained with the localization windows in Figures 8.3, 8.13, and 8.14. Importantly, the distribution, $E(n,k)$, does not employ a localization window.

Smoothness Index and Local Smoothness

The *smoothness index*, l, in graph signal processing plays the role of *frequency*, ω, in classical spectral analysis. For a graph signal, $\mathbf{x}$, the smoothness index is defined as the *Rayleigh quotient* of the matrix $\mathbf{L}$

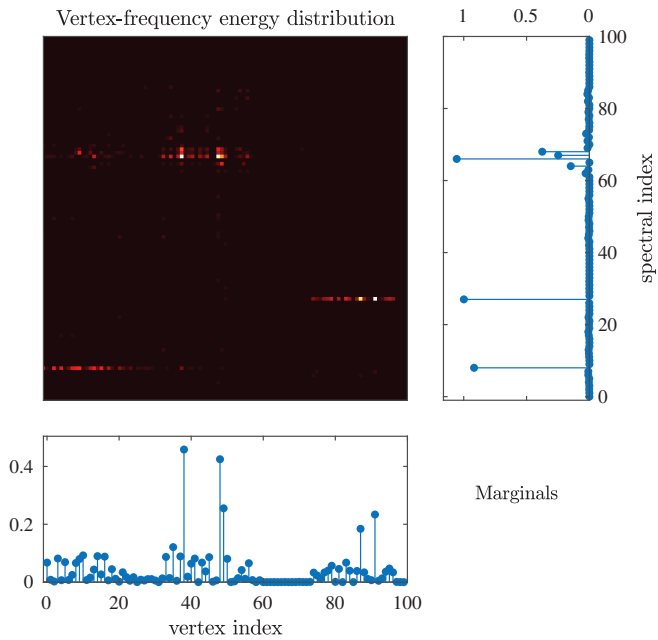


Figure 8.16: Vertex-frequency energy distribution for the graph signal whose vertex-frequency representation is given in Figure 8.3. No localization window was used here.

and vector $\mathbf{x}$, that is (see Section 4.2, Part I)

$$l = \frac{\mathbf{x}^T \mathbf{L}\mathbf{x}}{\mathbf{x}^T \mathbf{x}} \geq 0. \tag{8.66}$$

Remark 46: The expression in (8.66) indicates that the smoothness index can be considered as a measure of the rate of change of a graph signal. *Faster changing signals (corresponding to high-frequency signals) have larger values of the smoothness index.* The maximally smooth graph signal is then a constant signal, $x(n) = c$, for which the smoothness index is $l = 0$.

In the mathematics literature, the inverse of the smoothness index is known as the *curvature* (curvature $\sim 1/l$). While larger values of the smoothness index correspond to graph signals with larger rates of

change (less smooth graph signals), the larger values of curvature would indicate smoother graph signals.

Notice that the smoothness index for an eigenvector, $\mathbf{u}_k$, of the graph Laplacian, $\mathbf{L}$, is equal to its corresponding eigenvalue, λ_k, that is

$$\frac{\mathbf{u}_k^T \mathbf{L}\mathbf{u}_k}{\mathbf{u}_k^T \mathbf{u}_k} = \lambda_k, \tag{8.67}$$

since by definition $\mathbf{L}\mathbf{u}_k = \lambda_k \mathbf{u}_k$.

Remark 47: If the above eigenvectors are the classical Fourier transform basis functions, then the smoothness index corresponds to the squared frequency of the considered basis function, $\lambda_k \sim \omega_k^2$, while the curvature corresponds to the squared period in harmonic signals.

This makes it possible to define the local smoothness index for a vertex n, $\lambda(n)$, in analogy with the standard instantaneous frequency, $\omega(t)$, at an instant t, as Daković *et al.* (2019)

$$\lambda(n) = \frac{\mathcal{L}_x(n)}{x(n)}, \tag{8.68}$$

where it was assumed that $x(n) \neq 0$ and $\mathcal{L}_x(n)$ are the elements of the vector $\mathbf{Lx}$.

The properties of the local smoothness include:

1. The local smoothness index, $\lambda(n)$, for a monocomponent signal

   $$x(n) = \alpha u_k(n),$$

   is vertex independent, and is equal to the global smoothness index, λ_k, since

   $$\mathcal{L}_x(n) = \alpha \mathcal{L}_{u_k}(n) = \alpha \lambda_k u_k(n).$$

   In the standard time-domain signal analysis, this property means that the instantaneous frequency of a sinusoidal signal is equal to its global frequency.

2. Assume a piece-wise monocomponent signal

   $$x(n) = \alpha_i u_{k_i}(n) \quad \text{for } n \in \mathcal{V}_i,\ i = 1, 2, \ldots, M,$$

where $\mathcal{V}_i \subset \mathcal{V}$ are the subsets of the vertices such that $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ for $i \neq j$, $\mathcal{V}_1 \cup \mathcal{V}_2 \cup \cdots \cup \mathcal{V}_M = \mathcal{V}$, that is, every vertex belongs to only one subset, $\mathcal{V}_i$. Given the monocomponent nature of this signal, within each subset, $\mathcal{V}_i$, the considered signal is proportional to the eigenvector, $u_{k_i}(n)$.

Then, for each interior vertex, $n \in \mathcal{V}_i$, i.e., a vertex whose neighborhood lies in the same set, $\mathcal{V}_i$, the local smoothness index is given by

$$\lambda(n) = \frac{\alpha_i \mathcal{L}_{u_{k_i}}(n)}{\alpha_i u_{k_i}(n)} = \lambda_{k_i}. \tag{8.69}$$

3. An ideally concentrated vertex-frequency distribution (ideal distribution) can be defined as

$$I(n,k) \sim |x(n)|^2 \delta(\lambda_k - [\lambda(n)]),$$

whereby it is assumed that the local smoothness index is rounded to the nearest eigenvalue.

This distribution can also be used as a local smoothness estimator, since for each vertex, n, the maximum of $I(n,k)$ is positioned at $\lambda_k = \lambda(n)$. An estimate of the spectral index at a vertex, n, denoted by $\hat{k}(n)$, is then obtained as

$$\hat{k}(n) = \arg\max_k \{I(n,k)\},$$

so that the estimated local smoothness index becomes $\hat{\lambda}(n) = \lambda_{\hat{k}(n)}$. This type of estimator is widely used in classical time-frequency analysis (Boashash, 2015; Cohen, 1995; Stanković *et al.*, 2014).

4. **Local smoothness property.** The vertex-frequency distribution, $E(n,k)$, satisfies the local smoothness property if

$$\frac{\sum_{k=0}^{N-1} \lambda_k E(n,k)}{\sum_{k=0}^{N-1} E(n,k)} = \lambda(n). \tag{8.70}$$

In that case, the centers of masses of the vertex-frequency distribution along the spectral index axis, k, should be exactly at $\lambda = \lambda(n)$, and can be used as an unbiased estimator of this graph signal parameter.

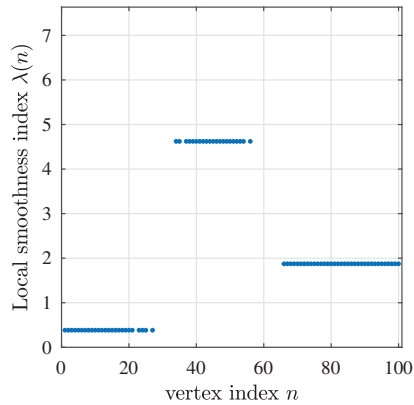


Figure 8.17: Local smoothness index, $\lambda(n)$, of the graph signal from Figure 8.1.

Example 30: The vertex-frequency distribution, defined by $E(n,k) = x(n)X(k)u_k(n)$, satisfies the local smoothness property in (8.70), since

$$\frac{\sum_{k=0}^{N-1}\lambda_k E(n,k)}{\sum_{k=0}^{N-1}E(n,k)} = \frac{\sum_{k=0}^{N-1}\lambda_k x(n)X(k)u_k(n)}{\sum_{k=0}^{N-1}x(n)X(k)u_k(n)} = \frac{\mathcal{L}_x(n)}{x(n)} = \lambda(n).$$

The above relation follows from the fact that $\sum_{k=0}^{N-1}\lambda_k X(k)u_k(n)$ are the elements of the IGFT of $\lambda_k X(k)$. Upon employing the matrix form of the IGFT of $\mathbf{\Lambda X}$, we have $\mathbf{U\Lambda X} = \mathbf{U\Lambda}(\mathbf{U}^T\mathbf{U})\mathbf{X} = (\mathbf{U\Lambda U}^T)(\mathbf{UX}) = \mathbf{Lx}$. With the notation, $\mathcal{L}_x(n)$, for the elements of $\mathbf{Lx}$, we next obtain

$$\sum_{k=0}^{N-1}\lambda_k X(k)u_k(n) = \mathcal{L}_x(n).$$

The local smoothness index for the graph signal from Figure 8.1 is shown in Figure 8.17.

Support Uncertainty Principle Derivation

From the energy condition for the Rihaczek distribution in (8.65) and (8.64), for the case of unit energy, we have

$$1 \leq \sum_{n=0}^{N-1}\sum_{k=0}^{N-1}|E(n,k)|. \tag{8.71}$$

Assume, as in Elad and Bruckstein (2002), that the support, $\mathbb{M}$, of the signal, $x(n)$, is $\mathbb{M} = \{n_1, n_2, \ldots, n_M\}$, meaning that $x(n) \neq 0$ for $n \in \mathbb{M}$ and $x(n) = 0$ for $n \notin \mathbb{M}$, while the support of the graph Fourier transform, $X(k)$, is $\mathbb{K} = \{k_1, k_2, \ldots, k_K\}$, where $X(k) \neq 0$ for $k \in \mathbb{K}$ and $X(k) = 0$ for $k \notin \mathbb{K}$. By definition, we can write

$$\|\mathbf{x}\|_0 = \text{card}\{\mathbb{M}\} = M \quad \text{and} \quad \|\mathbf{X}\|_0 = \text{card}\{\mathbb{K}\} = K. \tag{8.72}$$

Upon applying the Schwartz inequality to the square of (8.71), we have

$$\begin{aligned} 1 &= \Big(\sum_{n\in\mathbb{M}}\sum_{k\in\mathbb{K}} E(n,k)\Big)^2 \leq \Big(\sum_{n\in\mathbb{M}}\sum_{k\in\mathbb{K}} |x(n)|\ |X(k)|\ |u_k(n)|\Big)^2 \\ &= \Big(\sum_{n\in\mathbb{M}}\sum_{k\in\mathbb{K}} (\sqrt{|u_k(n)||x(n)|})\ (\sqrt{|u_k(n)|X(k)|})\Big)^2 \qquad (8.73) \\ &\leq \sum_{n\in\mathbb{M}}\sum_{k\in\mathbb{K}} |u_k(n)||x(n)|^2 \sum_{n\in\mathbb{M}}\sum_{k\in\mathbb{K}} |u_k(n)||X(k)|^2 \qquad (8.74) \\ &\leq \max_{n,k}\{|u_k(n)|^2\} KM = \max_{n,k}\{|u_k(n)|^2\}\|\mathbf{x}\|_0\|\mathbf{X}\|_0, \qquad (8.75) \end{aligned}$$

from the unit energy of the graph signal, $\sum_{n\in\mathbb{M}} |x(n)|^2 = \sum_{k\in\mathbb{K}} |X(k)|^2 = 1$.

The inequality in (8.75) results in the following support uncertainty principle (Elad and Bruckstein, 2002)

$$\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq \frac{1}{\max_{n,k}\{|u_k(n)|^2\}}. \tag{8.76}$$

An improved bound of the support uncertainty principle was recently derived in Stanković (2020), using the same relations.

Reduced Interference Distributions (RID) on Graphs

In order to emphasize the close relations with classical time-frequency analysis, in this subsection we will use the complex-sensitive notation for eigenvectors and spectral vectors. The frequency domain definition of the energy distribution in (8.65) is given by

$$E(n,k) = x(n)X^*(k)u_k^*(n) = \sum_{p=0}^{N-1} X(p)X^*(k)u_p(n)u_k^*(n).$$

Then, the general form of a graph distribution can be defined with the help of a kernel $\phi(p,k,q)$, as Stanković *et al.* (2018a)

$$G(n,k)=\sum_{p=0}^{N-1}\sum_{q=0}^{N-1}X(p)X^*(q)u_p(n)u_q^*(n)\phi(p,k,q). \tag{8.77}$$

Observe that for $\phi(p,k,q)=\delta(q-k)$, the graph Rihaczek distribution in (8.65) follows, while the unbiased energy condition $\sum_{k=0}^{N-1}\sum_{n=0}^{N-1}G(n,k)=E_x$ is satisfied if

$$\sum_{k=0}^{N-1}\phi(p,k,p)=1.$$

The so obtained distribution, $G(n,k)$, may also satisfy the vertex and frequency marginal properties, as elaborated below.

- The *vertex marginal property* is satisfied if

$$\sum_{k=0}^{N-1}\phi(p,k,q)=1.$$

  This is obvious from

$$\sum_{k=0}^{N-1}G(n,k)=\sum_{p=0}^{N-1}\sum_{q=0}^{N-1}X(p)X^*(q)u_p(n)u_q^*(n)=|x(n)|^2.$$

- The *frequency marginal property* is satisfied if

$$\phi(p,k,p)=\delta(p-k).$$

  Then, the sum over all vertex indices produces

$$\sum_{n=0}^{N-1}G(n,k)=\sum_{p=0}^{N-1}|X(p)|^2\phi(p,k,p)=|X(k)|^2,$$

  since $\sum_{n=0}^{N-1}u_p(n)u_q^*(n)=\delta(p-q)$, that is, the eigenvectors are orthonormal.

Reduced Interference Distribution Kernels

A straightforward extension of classical time-frequency kernels to graph signal processing would be naturally based upon exploiting the relation $\lambda\sim\omega^2$, together with an appropriate exponential kernel normalization.

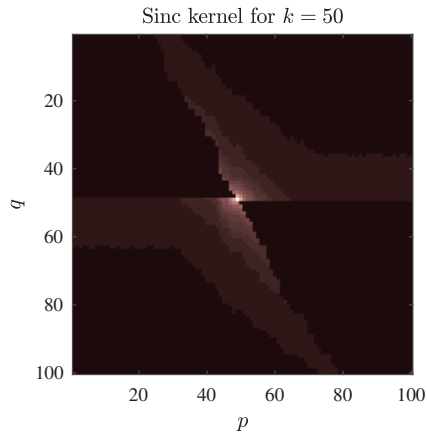


Figure 8.18: The sinc kernel of the reduced interference vertex-frequency distribution in the frequency domain.

The simplest reduced interference kernel in the frequency–frequency shift domain, which would satisfy the marginal properties, is the *sinc kernel*, given by

$$\phi(p,k,q) = \begin{cases} \dfrac{1}{1+2|p-q|}, & \text{for } |k-p| \leq |p-q|, \\ 0, & \text{otherwise,} \end{cases}$$

which is shown in Figure 8.18 at the frequency shift corresponding to $k = 50$.

Example 31: The sinc kernel was used for a vertex-frequency representation of the signal from Figure 8.1(d), with the results shown in Figure 8.19. This representation is a smoothed version of the energy vertex-frequency distribution in Figure 8.16, whereby both (vertex and frequency) marginals are preserved.

Remark 48: **Marginal properties of graph spectrogram.** A general vertex-frequency distribution can be written for the vertex–vertex shift domain as a dual form of (8.77), to yield

$$G(n,k) = \sum_{m=0}^{N-1} \sum_{l=0}^{N-1} x(m)x^*(l)u_k(m)u_k^*(l)\varphi(m,n,l), \tag{8.78}$$

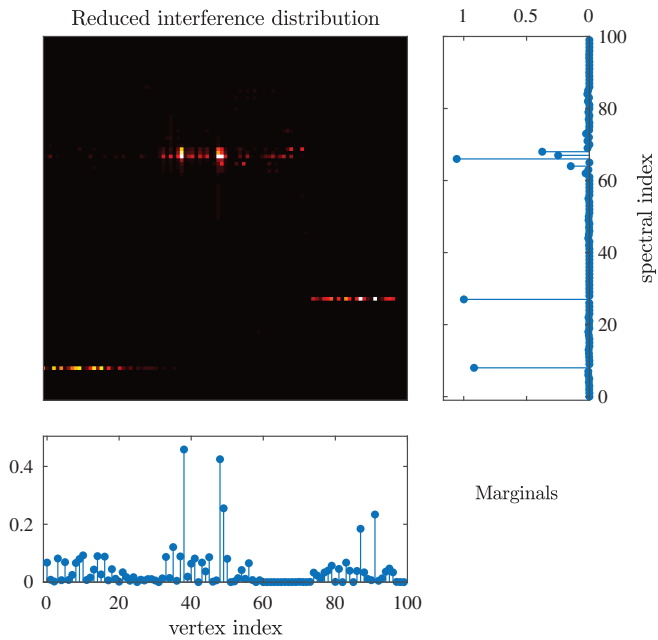


Figure 8.19: Reduced interference vertex-frequency distribution of a signal whose vertex-frequency representation is given in Figure 8.3. The marginal properties are given in the panels to the right and below the vertex-frequency representation, and are equal to their corresponding ideal forms given by $|x(n)|^2$ and $|X(k)|^2$.

where $\varphi(m,n,l)$ is the kernel in this domain (the same mathematical form as for the frequency–frequency shift domain kernel). The frequency marginal is then satisfied if $\sum_{n=0}^{N-1}\varphi(m,n,l)=1$ holds, while the vertex marginal is met if $\varphi(m,n,m)=\delta(m-n)$. The relation of this distribution with the vertex domain spectrogram (8.1) is simple, and is given by

$$\varphi(m,n,l)=h_n(m)h_n^*(l).$$

However, this kernel cannot satisfy both the frequency and vertex marginal properties, while the unbiased energy condition $\sum_{n=0}^{N-1}\varphi(m,n,m)=1$ reduces to (8.46).

Remark 49: **Classical time-frequency analysis** follows as a special case from the general form of graph distributions in (8.77), if the

considered graph is a directed circular graph. This becomes obvious upon recalling that the adjacency matrix eigendecomposition produces complex-valued eigenvectors of the form $u_k(n) = \exp(j2\pi nk/N)/\sqrt{N}$. With the kernel choice

$$\phi(p,k,q) = \phi(p-q,k-p) = \sum_{n=0}^{N-1} c(p-q,n) e^{-j\frac{2\pi nk}{N}} e^{j\frac{2\pi np}{N}}$$

in (8.77), the classical (Rihaczek based) Cohen class of distributions directly follows, where $c(k,n)$ is the distribution kernel in the ambiguity domain (Boashash, 2015; Cohen, 1995; Stanković *et al.*, 2014).

A comparison of various vertex-frequency method may be found in Stanković *et al.* (2020).

A interesting combination of the time and vertex signal variations into time-vertex signal processing is done in Grassi *et al.* (2017) and Bohannon *et al.* (2019).

9

Conclusion

Fundamental ideas of graph signals and their analysis have been introduced starting from an intuitive multisensor estimation example, frequently considered in traditional data analytics. The concept of systems on graphs has been defined using graph signal shift operators, which generalize the signal shift concepts in traditional signal processing. In Part II of our monograph, the Graph Discrete Fourier Transform (GFT) has been at the core of the spectral domain representation of graph signals and systems on graphs, and has been defined based on both the adjacency matrix and graph Laplacian. These spectral domain representations have been used as the basis to introduce graph signal filtering concepts. Methods for the design of graph filters have been presented next, including those based on the polynomial approximation. Various ideas related to the sampling of graph signals, and particularly, the challenging topic of the subsampling, have also been addressed in this part of the monograph. This is followed by conditions for the recovery of signals on graphs, from a reduced number of samples. The concepts of time-varying signals on graphs and basic definitions and methods related to processing random graph signals have also been introduced.

While traditional approaches for graph signal analysis, clustering and segmentation consider only graph topology and spectral properties of graphs, when dealing with signals on graphs, localized analyzes should be employed in order to consider both data on graphs and the graph topology. Such a unified approach to define and implement graph signal localization methods, which takes into account both the data on graph and the corresponding graph topology, is at the core of the presented vertex-frequency analysis. Like in classical time-frequency analysis, main research efforts have been devoted to linear representations of the graph signals which include a localization window for enhanced signal discrimination. Several methods for the definition of localization widows in the spectral and vertex domain have been addressed in Part II of this monograph. Optimization of the window parameters, uncertainty principle, and inversion methods have also been discussed. Following classical time-frequency analysis, energy forms of vertex-frequency energy and reduced interference distributions, which do not use localization windows, have also been considered, together with the elaboration of their role as an estimator of the local smoothness index.

References

Agaskar, A. and Y. M. Lu (2013). "A spectral graph uncertainty principle". *IEEE Transactions on Information Theory*. 59(7): 4338–4356.

Anis, A., A. Gadde, and A. Ortega (2016). "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies". *IEEE Transactions on Signal Processing*. 64(14): 3775–3789.

Behjat, H., N. Leonardi, L. Sörnmo, and D. Van De Ville (2015). "Anatomically-adapted graph wavelets for improved group-level fMRI activation mapping". *NeuroImage*. 123: 185–199.

Behjat, H., U. Richter, D. Van De Ville, and L. Sörnmo (2016). "Signal-adapted tight frames on graphs". *IEEE Transactions on Signal Processing*. 64(22): 6017–6029.

Behjat, H. and D. Van De Ville (2019). "Spectral design of signal-adapted tight frames on graphs". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 177–206.

Boashash, B. (2015). *Time-Frequency Signal Analysis and Processing: A Comprehensive Reference*. Academic Press.

Bohannon, A. W., B. M. Sadler, and R. V. Balan (2019). "A filtering framework for time-varying graph signals". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 341–376.

Candes, E. J. (2008). "The restricted isometry property and its implications for compressed sensing". *Comptes Rendus Mathematique*. 346(9–10): 589–592.

Chen, S., A. Sandryhaila, and J. Kovačević (2015a). "Sampling theory for graph signals". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 3392–3396.

Chen, S., A. Sandryhaila, J. M. Moura, and J. Kovačević (2014). "Signal denoising on graphs via graph filtering". In: *Proc. 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. 872–876.

Chen, S., A. Sandryhaila, J. M. Moura, and J. Kovačević (2015b). "Signal recovery on graphs: Variation minimization". *IEEE Transactions on Signal Processing*. 63(17): 4609–4624.

Chen, S., R. Varma, A. Sandryhaila, and J. Kovačević (2015c). "Discrete signal processing on graphs: Sampling theory". *IEEE Transactions on Signal Processing*. 63(24): 6510–6523.

Chen, S., R. Varma, A. Singh, and J. Kovačević (2016). "Signal recovery on graphs: Fundamental limits of sampling strategies". *IEEE Transactions on Signal and Information Processing over Networks*. 2(4): 539–554.

Chepuri, S. P. and G. Leus (2016). "Subsampling for graph power spectrum estimation". In: *Proc. IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*. 1–5.

Cioacă, T., B. Dumitrescu, and M.-S. Stupariu (2019). "Graph-based wavelet multiresolution modeling of multivariate terrain data". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 479–507.

Cohen, L. (1995). *Time-Frequency Analysis. Electrical Engineering Signal Processing*. Prentice Hall PTR.

Coifman, R. R. and S. Lafon (2006). "Diffusion maps". *Applied and Computational Harmonic Analysis*. 21(1): 5–30.

Daković, M., L. Stanković, and E. Sejdić (2019). "Local smoothness of graph signals". *Mathematical Problems in Engineering*. 2019: Article ID 3208569.

Ekambaram, V. N. (2014). *Graph-Structured Data Viewed Through a Fourier Lens*. Berkeley: University of California.

Elad, M. and A. M. Bruckstein (2002). "Generalized uncertainty principle and sparse representation in pairs of bases". *IEEE Transactions on Information Theory*. 48(9): 2558–2567.

Erb, W. (2019). "Shapes of uncertainty in spectral graph theory". *arXiv preprint arXiv:1909.10865*.

Gama, F., E. Isufi, A. Ribeiro, and G. Leus (2019). "Controllability of bandlimited graph processes over random time varying graphs". *IEEE Transactions on Signal Processing*. 67(24): 6440–6454.

Gavili, A. and X.-P. Zhang (2017). "On the shift operator, graph frequency, and optimal filtering in graph signal processing". *IEEE Transactions on Signal Processing*. 65(23): 6303–6318.

Girault, B. (2015). "Stationary graph signals using an isometric graph translation". In: *Proc. 23rd European Signal Processing Conference (EUSIPCO)*. 1516–1520.

Girault, B., P. Gonçalves, and É. Fleury (2015). "Translation on graphs: An isometric shift operator". *IEEE Signal Processing Letters*. 22(12): 2416–2420.

Grassi, F., A. Loukas, N. Perraudin, and B. Ricaud (2017). "A time-vertex signal processing framework: Scalable processing and meaningful representations for time-series on graphs". *IEEE Transactions on Signal Processing*. 66(3): 817–829.

Hammond, D., P. Vandergheynst, and R. Gribonval (2011). "Wavelets on graphs via spectral graph theory". *Applied and Computational Harmonic Analysis*. 30(2): 129–150.

Hammond, D. K., P. Vandergheynst, and R. Gribonval (2019). "The spectral graph wavelet transform: Fundamental theory and fast computation". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 141–175.

Hamon, R., P. Borgnat, P. Flandrin, and C. Robardet (2016). "Extraction of temporal network structures from graph-based signals". *IEEE Transactions on Signal and Information Processing over Networks*. 2(2): 215–226.

Heimowitz, A. and Y. C. Eldar (2017). "A unified view of diffusion maps and signal processing on graphs". *In Proceedings of the International Conference on Sampling Theory and Applications (SampTA)*: 308–312.

Isufi, E., A. Loukas, A. Simonetto, and G. Leus (2017). "Filtering random graph processes over random time-varying graphs". *IEEE Transactions on Signal Processing*. 65(16): 4406–4421.

Jansen, M., G. P. Nason, and B. W. Silverman (2009). "Multiscale methods for data on graphs and irregular multidimensional situations". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*. 71(1): 97–125.

Jestrović, I., J. L. Coyle, and E. Sejdić (2017). "A fast algorithm for vertex-frequency representations of signals on graphs". *Signal Processing*. 131: 483–491.

Kim, S.-J., K. Koh, S. Boyd, and D. Gorinevsky (2009). "L1 trend filtering". *SIAM Review*. 51(2): 339–360.

Lee, A. B., B. Nadler, and L. Wasserman (2008). "Treelets: An adaptive multiscale basis for sparse unordered data". *The Annals of Applied Statistics*. 2(2): 435–471.

Leonardi, N. and D. Van De Ville (2013). "Tight wavelet frames on multislice graphs". *IEEE Transactions on Signal Processing*. 61(13): 3357–3367.

Leskovec, J. and C. Faloutsos (2006). "Sampling from large graphs". In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 631–636.

Li, S., Y. Jin, and D. I. Shuman (2019). "Scalable M-channel critically sampled filter banks for graph signals". *IEEE Transactions on Signal Processing*. 67(15): 3954–3969.

Lorenzo, P., S. Barbarossa, and P. Banelli (2018). "Sampling and recovery of graph signals". In: *Cooperative and Graph Signal Processing*. Ed. by C. Richard and P. Djuric. Elsevier. 261–282.

Loukas, A. and N. Perraudin (2016). "Stationary time-vertex signal processing". *arXiv preprint arXiv:1611.00255*.

Maggioni, M. and H. Mhaskar (2008). "Diffusion polynomial frames on metric measure spaces". *Applied and Computational Harmonic Analysis*. 24(3): 329–353.

Marques, A. G., S. Segarra, G. Leus, and A. Ribeiro (2016). "Sampling of graph signals with successive local aggregations." *IEEE Transactions Signal Processing*. 64(7): 1832–1843.

Marques, A. G., S. Segarra, G. Leus, and A. Ribeiro (2017). "Stationary graph processes and spectral estimation". *IEEE Transactions on Signal Processing*. 65(22): 5911–5926.

Masoumi, M., M. Rezaei, and A. B. Hamza (2019). "Shape analysis of carpal bones using spectral graph wavelets". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 419–436.

Meyer, Y. (1992). *Wavelets and Operators*. Cambridge University Press.

Misiakos, P., C. Wendler, and M. Püschel (2020). "Diagonalizable shift and filters for directed graphs based on the Jordan–Chevalley decomposition". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 5635–5639.

Moura, J. M. (2018). "Graph signal processing". In: *Cooperative and Graph Signal Processing*. Ed. by P. Djuric and C. Richard. Elsevier. 239–259.

Murtagh, F. (2007). "The Haar wavelet transform of a dendrogram". *Journal of Classification*. 24(1): 3–32.

Narang, S. K. and A. Ortega (2009). "Lifting based wavelet transforms on graphs". In: *Proc. Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*. 441–444.

Narang, S. K. and A. Ortega (2011). "Downsampling graphs using spectral theory". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4208–4211.

Narang, S. K. and A. Ortega (2012). "Perfect reconstruction two-channel wavelet filter banks for graph structured data". *IEEE Transactions on Signal Processing*. 60(6): 2786–2799.

Nguyen, H. Q. and M. N. Do (2015). "Downsampling of signals on graphs via maximum spanning trees." *IEEE Transactions on Signal Processing*. 63(1): 182–191.

Perraudin, N., B. Ricaud, D. I. Shuman, and P. Vandergheynst (2018). "Global and local uncertainty principles for signals on graphs". *APSIPA Transactions on Signal and Information Processing*. 7(e3): 1–26.

Perraudin, N. and P. Vandergheynst (2017). "Stationary signal processing on graphs". *IEEE Transactions on Signal Processing*. 65(13): 3462–3477.

Puy, G., N. Tremblay, R. Gribonval, and P. Vandergheynst (2018). "Random sampling of bandlimited signals on graphs". *Applied and Computational Harmonic Analysis*. 44(2): 446–475.

Ricaud, B. and B. Torrésani (2014). "A survey of uncertainty principles and some signal processing applications". *Advances in Computational Mathematics*. 40(3): 629–650.

Rustamov, R. and L. J. Guibas (2013). "Wavelets on graphs via deep learning". In: *Advances in Neural Information Processing Systems*. 998–1006.

Sakiyama, A. and Y. Tanaka (2014). "Oversampled graph Laplacian matrix for graph filter banks". *IEEE Transactions on Signal Processing*. 62(24): 6425–6437.

Sandryhaila, A. and J. M. Moura (2013). "Discrete signal processing on graphs". *IEEE Transactions on Signal Processing*. 61(7): 1644–1656.

Sandryhaila, A. and J. M. Moura (2014a). "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure". *IEEE Signal Processing Magazine*. 31(5): 80–90.

Sandryhaila, A. and J. M. Moura (2014b). "Discrete signal processing on graphs: Frequency analysis". *IEEE Transactions on Signal Processing*. 62(12): 3042–3054.

Segarra, S., A. G. Marques, G. Leus, and A. Ribeiro (2015). "Interpolation of graph signals using shift-invariant graph filters". In: *Proc. 23rd European Signal Processing Conference (EUSIPCO)*. 210–214.

Segarra, S. and A. Ribeiro (2016). "Stability and continuity of centrality measures in weighted graphs". *IEEE Transactions on Signal Processing*. 64(3): 543–555.

Shuman, D. I., S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst (2013). "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains". *IEEE Signal Processing Magazine*. 30(3): 83–98.

Shuman, D. I., B. Ricaud, and P. Vandergheynst (2012). "A windowed graph Fourier transform". In: *Proc. IEEE Statistical Signal Processing Workshop (SSP)*. 133–136.

Shuman, D. I., B. Ricaud, and P. Vandergheynst (2016). "Vertex-frequency analysis on graphs". *Applied and Computational Harmonic Analysis*. 40(2): 260–291.

Stanković, L. (1997). "Highly concentrated time-frequency distributions: Pseudo quantum signal representation". *IEEE Transactions on Signal Processing*. 45(3): 543–551.

Stanković, L. (2001). "A measure of some time–frequency distributions concentration". *Signal Processing*. 81(3): 621–631.

Stanković, L. (2015). *Digital Signal Processing with Selected Topics*. CreateSpace Independent Publishing Platform, An Amazon.com Company.

Stanković, L. (2020). "The support uncertainty principle and the graph Rihaczek distribution: Revisited and improved". *IEEE Signal Processing Letters*. 27: 1030–1034.

Stanković, L., M. Daković, and E. Sejdić (2017). "Vertex-frequency analysis: A way to localize graph spectral components [Lecture Notes]". *IEEE Signal Processing Magazine*. 34(4): 176–182.

Stanković, L., M. Daković, and E. Sejdić (2019a). "Introduction to graph signal processing". In: *Vertex-Frequency Analysis of Graph Signals*. Springer. 3–108.

Stanković, L., M. Daković, and E. Sejdić (2019b). "Vertex-frequency energy distributions". In: *Vertex-Frequency Analysis of Graph Signals*. Ed. by L. Stanković and E. Sejdić. Springer. 377–415.

Stanković, L., M. Daković, and T. Thayaparan (2014). *Time-Frequency Signal Analysis with Applications*. Artech House.

Stanković, L., D. P. Mandic, M. Dakovic, I. Kisil, E. Sejdic, and A. G. Constantinides (2019). "Understanding the basis of graph signal processing via an intuitive example-driven approach [Lecture Notes]". *IEEE Signal Processing Magazine*. 36(6): 133–145.

Stanković, L., D. P. Mandic, M. Dakovic, and I. Kisil (2020). "Demystifying the coherence index in compressive sensing [Lecture Notes]". *IEEE Signal Processing Magazine*. 37(1): 152–162.

Stanković, L., D. Mandic, M. Daković, B. Scalzo, M. Brajović, E. Sejdić, and A. G. Constantinides (2020). "Vertex-frequency graph signal processing: A comprehensive review". *Digital Signal Processing*: 102802.

Stanković, L., E. Sejdić, and M. Daković (2018a). "Reduced interference vertex-frequency distributions". *IEEE Signal Processing Letters*. 25(9): 1393–1397.

Stanković, L., E. Sejdić, and M. Daković (2018b). "Vertex-frequency energy distributions". *IEEE Signal Processing Letters*. 25(3): 358–362.

Stanković, L., E. Sejdić, S. Stanković, M. Daković, and I. Orović (2018c). "A tutorial on sparse signal reconstruction and its applications in signal processing". *Circuits, Systems, and Signal Processing*: 1–58.

Tanaka, Y. and Y. C. Eldar (2020). "Generalized sampling on graphs with subspace and smoothness priors". *IEEE Transactions on Signal Processing*. 68: 2272–2286.

Tanaka, Y. and A. Sakiyama (2014). "M-channel oversampled graph filter banks". *IEEE Transactions Signal Processessing*. 62(14): 3578–3590.

Tepper, M. and G. Sapiro (2016). "A short-graph Fourier transform via personalized pagerank vectors". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4806–4810.

Tremblay, N. and P. Borgnat (2016). "Subgraph-based filterbanks for graph signals". *IEEE Transactions Signal Processing*. 64(15): 3827–3840.

Tsitsvero, M., S. Barbarossa, and P. Di Lorenzo (2016). "Signals on graphs: Uncertainty principle and sampling". *IEEE Transactions Signal Processing*. 64(18): 539–554.

Venkitaraman, A., S. Chatterjee, and P. Händel (2016). "Hilbert transform, analytic signal, and modulation analysis for graph signal processing". *arXiv preprint arXiv:1611.05269*.

Vetterli, M., J. Kovačević, and V. Goyal (2014). *Foundations of Signal Processing*. Cambridge University Press.

Wang, X., J. Chen, and Y. Gu (2016). "Local measurement and reconstruction for noisy bandlimited graph signals". *Signal Processing*. 129: 119–129.

Wang, X., P. Liu, and Y. Gu (2015). "Local-set-based graph signal reconstruction". *IEEE Transactions on Signal Processing*. 63(9): 2432–2444.

Yan, X., B. M. Sadler, R. J. Drost, P. L. Yu, and K. Lerman (2017). "Graph filters and the Z-Laplacian". *IEEE Journal of Selected Topics in Signal Processing*. 11(6): 774–784.

Zhang, C., D. Florêncio, and P. A. Chou (2015). "Graph signal processing—A probabilistic framework". *Microsoft Research, Redmond, WA, USA, Tech. Rep. MSR-TR-2015-31*.

Zheng, X.-W., Y. Y. Tang, J.-T. Zhou, H.-L. Yuan, Y.-L. Wang, L.-N. Yang, and J.-J. Pan (2016). "Multi-windowed graph Fourier frames". In: *Proc. IEEE International Conference on Machine Learning and Cybernetics (ICMLC)*. Vol. 2. 1042–1048.