Signals on Networks: Random Asynchronous and Multirate Processing, and Uncertainty Principles

Thesis by

Oguzhan Teke

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2020
Defended July 24th, 2020

To my wife, my parents, and my sister

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Prof. P. P. Vaidyanathan for providing me with the opportunity to enjoy Caltech. I cannot thank him enough for giving me the freedom to pursue my ideas in research and providing unconditional support at every stage. His encouragement for studying fundamental problems formed the backbone of my doctoral work, for which I am indebted to him forever. My years working with him made me not just a better researcher, but a better person in general. It was a great journey, and I will always remember these years with joy.

I would like to thank the members of my defense and candidacy committees: Prof. Venkat Chandrasekaran, Prof. Adam Wierman, Prof. Yaser Abu-Mostafa, and Prof. Babak Hassibi. I truly appreciate their feedback and comments on my research, and future plans. It was a great pleasure to have them in my committees. I would like to also thank them for the valuable courses they had offered throughout the years at Caltech, which formed a solid foundation for my theoretical understanding.

I would like to thank past and present members of my group Prof. Chun-Lin Liu, Dr. Srikanth Tenneti, Po-Chih Chen and Pranav Kulkarni for their encouragement and valuable feedback. I will truly miss our weekly group meetings! I would like to also thank to Prof. Piya Pal, a former member of our group, for her support and valuable discussions.

I would like to thank Caroline Murphy, Tanya Owen, Carol Sosnowski, and Angie Riley for their administrative assistance.

I would like to thank my friends Sinan Kefeli, Şahin Lale, Hikmet Yıldız, Recep Yavaş, Kordağ Kılıç, Taylan Kargın, and Dr. Utkan Candoğan. I will truly miss our afternoon coffee breaks, walking around the campus, and discussing about almost anything. I would like to thank to Zeynep Turan, Hoang Le, and Corina Panda for their friendship.

Finally, I would like to express my sincere gratitude to my father Süleyman, my mother Nursel, and my sister Hande for their continuous support and encouragement. I want to also thank to my wife Halime for her understanding and continuous support throughout my years at Caltech while pursing her own career.

ABSTRACT

The processing of signals defined on graphs has been of interest for many years, and finds applications in a diverse set of fields such as sensor networks, social and economic networks, and biological networks. In graph signal processing applications, signals are not defined as functions on a uniform time-domain grid but they are defined as vectors indexed by the vertices of a graph, where the underlying graph is assumed to model the irregular signal domain. Although analysis of such networked models is not new (it can be traced back to the consensus problem studied more than four decades ago), such models are studied recently from the view-point of signal processing, in which the analysis is based on the "graph operator" whose eigenvectors serve as a Fourier basis for the graph of interest. With the help of graph Fourier basis, a number of topics from classical signal processing (such as sampling, reconstruction, filtering, etc.) are extended to the case of graphs.

The main contribution of this thesis is to provide new directions in the field of graph signal processing and provide further extensions of topics in classical signal processing. The first part of this thesis focuses on a random and asynchronous variant of "graph shift," i.e., localized communication between neighboring nodes. Since the dynamical behavior of randomized asynchronous updates is very different from standard graph shift (i.e., state-space models), this part of the thesis focuses on the convergence and stability behavior of such random asynchronous recursions. Although non-random variants of asynchronous state recursions (possibly with non-linear updates) are well-studied problems with early results dating back to the late 60's, this thesis considers the convergence (and stability) in the statistical mean-squared sense and presents the precise conditions for the stability by drawing parallels with switching systems. It is also shown that systems exhibit unexpected behavior under randomized asynchronicity: an unstable system (in the synchronous world) may be stabilized simply by the use of randomized asynchronicity. Moreover, randomized asynchronicity may result in a lower total computational complexity in certain parameter settings. The thesis presents applications of the random asynchronous model in the context of graph signal processing including an autonomous clustering of network of agents, and a node-asynchronous communication protocol that implements a given rational filter on the graph.

The second part of the thesis focuses on extensions of the following topics in classical signal processing to the case of graph: multirate processing and filter banks, discrete uncertainty principles, and energy compaction filters for optimal filter design. The thesis also considers an application to the heat diffusion over networks.

Multirate systems and filter banks find many applications in signal processing theory and implementations. Despite the possibility of extending 2-channel filter banks to bipartite graphs, this thesis shows that this relation cannot be generalized to M-channel systems on M-partite graphs. As a result, the extension of classical multirate theory to graphs is nontrivial, and such extensions cannot be obtained without certain mathematical restrictions on the graph. The thesis provides the necessary conditions on the graph such that fundamental building blocks of multirate processing remain valid in the graph domain. In particular, it is shown that when the underlying graph satisfies a condition called M-block cyclic property, classical multirate theory can be extended to the graphs.

The uncertainty principle is an essential mathematical concept in science and engineering, and uncertainty principles generally state that a signal cannot have an arbitrarily "short" description in the original basis and in the Fourier basis simultaneously. Based on the fact that graph signal processing proposes two different bases (i.e., vertex and the graph Fourier domains) to represent graph signals, this thesis shows that the total number of nonzero elements of a graph signal and its representation in the graph Fourier domain is lower bounded by a quantity depending on the underlying graph. The thesis also presents the necessary and sufficient condition for the existence of 2-sparse and 3-sparse eigenvectors of a connected graph. When such eigenvectors exist, the uncertainty bound is very low, tight, and independent of the global structure of the graph.

The thesis also considers the classical spectral concentration problem. In the context of polynomial graph filters, the problem reduces to the polynomial concentration problem studied more generally by Slepian in the 70's. The thesis studies the asymptotic behavior of the optimal solution in the case of narrow bandwidth. Different examples of graphs are also compared in order to show that the maximum energy compaction and the optimal filter depends heavily on the graph spectrum.

In the last part, the thesis considers the estimation of the starting time of a heat diffusion process from its noisy measurements when there is a single point source located on a known vertex of a graph with unknown starting time. In particular, the Cramér-Rao lower bound for the estimation problem is derived, and it is shown that for graphs with higher connectivity the problem has a larger lower bound making the estimation problem more difficult.

PUBLISHED CONTENT AND CONTRIBUTIONS

All the papers below resulted from research carried out by Oguzhan Teke and Professor Palghat P. Vaidyanathan. Oguzhan Teke was the primary contributor of these papers. He proposed the main results, proved theorems and lemmas, conducted experimental validations, wrote papers, and corresponded with editors and reviewers. Professor Palghat P. Vaidyanathan applied for research grants, supervised research projects, discussed research topics with Oguzhan Teke, and reviewed the main results, paper drafts, and the correspondences with reviewers.

[1] Oguzhan Teke and Palghat P. Vaidyanathan. "IIR Filtering on Graphs with Random Node-Asynchronous Updates". In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 3945–3960. DOI: 10.1109/TSP.2020.3004912.

[2] Oguzhan Teke and Palghat P. Vaidyanathan. "Node-Asynchronous Spectral Clustering on Directed Graphs". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2020, pp. 5325–5329. DOI: 10.1109/ICASSP40776.2020.9054241.

[3] Oguzhan Teke and Palghat P. Vaidyanathan. "Node-asynchronous Implementation of Rational Filters on Graphs". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2019, pp. 7530–7534. DOI: 10.1109/ICASSP.2019.8682946.

[4] Oguzhan Teke and Palghat P. Vaidyanathan. "Random Node-Asynchronous Updates on Graphs". In: *IEEE Transactions on Signal Processing* 67.11 (June 2019), pp. 2794–2809. DOI: 10.1109/TSP.2019.2910485.

[5] Oguzhan Teke and Palghat P. Vaidyanathan. "Randomized Asynchronous Recursions with a Sinusoidal Input". In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2019, pp. 1491–1495. DOI: 10.1109/IEEECONF44664.2019.9048998.

[6] Oguzhan Teke and Palghat P. Vaidyanathan. "The Random Component-Wise Power Method". In: *Proceedings of SPIE, Wavelets and Sparsity XVIII*. Vol. 11138. Sept. 2019, pp. 473–488. DOI: 10.1117/12.2530511.

[7] Oguzhan Teke and Palghat P. Vaidyanathan. "Asynchronous Nonlinear Updates on Graphs". In: *Asilomar Conference on Signals, Systems, and Computers*. Oct. 2018, pp. 998–1002. DOI: 10.1109/ACSSC.2018.8645351.

[8] Oguzhan Teke and Palghat P. Vaidyanathan. "Energy Compaction Filters on Graphs". In: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Nov. 2018, pp. 783–787. DOI: 10.1109/GlobalSIP.2018.8646570.

[9] Oguzhan Teke and Palghat P. Vaidyanathan. “The Asynchronous Power Iteration: A Graph Signal Perspective”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Apr. 2018, pp. 4059–4063. DOI: 10.1109/ICASSP.2018.8461872.

[10] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending classical multirate signal processing theory to graphs”. In: *Proceedings SPIE, Wavelets and Sparsity XVII*. Vol. 10394. Aug. 2017, pp. 354–365. DOI: 10.1117/12.2272362.

[11] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending Classical Multirate Signal Processing Theory to Graphs – Part I: Fundamentals”. In: *IEEE Transactions on Signal Processing* 65.2 (Jan. 2017), pp. 409–422. DOI: 10.1109/TSP.2016.2617833.

[12] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending Classical Multirate Signal Processing Theory to Graphs – Part II: M-Channel Filter Banks”. In: *IEEE Transactions on Signal Processing* 65.2 (Jan. 2017), pp. 423–437. DOI: 10.1109/TSP.2016.2620111.

[13] Oguzhan Teke and Palghat P. Vaidyanathan. “On the Role of the Bounded Lemma in the SDP Formulation of Atomic Norm Problems”. In: *IEEE Signal Processing Letters* 24.7 (July 2017), pp. 972–976. DOI: 10.1109/LSP.2017.2700442.

[14] Oguzhan Teke and Palghat P. Vaidyanathan. “Sparse Eigenvectors of Graphs”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2017, pp. 3904–3908. DOI: 10.1109/ICASSP.2017.7952888.

[15] Oguzhan Teke and Palghat P. Vaidyanathan. “Time Estimation for Heat Diffusion on Graphs”. In: *Asilomar Conference on Signals, Systems, and Computers*. Oct. 2017, pp. 1963–1967. DOI: 10.1109/ACSSC.2017.8335709.

[16] Oguzhan Teke and Palghat P. Vaidyanathan. “Uncertainty Principles and Sparse Eigenvectors of Graphs”. In: *IEEE Transactions on Signal Processing* 65.20 (Oct. 2017), pp. 5406–5420. DOI: 10.1109/TSP.2017.2731299.

[17] Oguzhan Teke and Palghat P. Vaidyanathan. “Discrete uncertainty principles on graphs”. In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2016, pp. 1475–1479. DOI: 10.1109/ACSSC.2016.7869622.

[18] Oguzhan Teke and Palghat P. Vaidyanathan. “Graph filter banks with M-channels, maximal decimation, and perfect reconstruction”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2016, pp. 4089–4093. DOI: 10.1109/ICASSP.2016.7472446.

[19] Oguzhan Teke and Palghat P. Vaidyanathan. “Linear systems on graphs”. In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Dec. 2016, pp. 385–389. DOI: 10.1109/GlobalSIP.2016.7905869.

[20] Oguzhan Teke and Palghat P. Vaidyanathan. “Fundamentals of multirate graph signal processing”. In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2015, pp. 1791–1795. DOI: 10.1109/ACSSC.2015.7421460.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Chapter 1

INTRODUCTION

With the availability of various type of sensors and the increased level of connectivity, we live in a world where it is very easy to collect and store almost every detail in daily life, from hourly blood pressure levels (through wearable devices) to social interactions (through various types of social networks). In most cases, such kind of data are interrelated with each other, and they present complex dependency structures. Since time-series analysis is no longer applicable to such data combinations, researchers have been studying different scientific approaches in order to incorporate the underlying dependencies into the analysis of such data.

Recent years have witnessed an elevated interest in data models where the underlying dependencies are represented via *graphs*. This is a very broad model that can be found in a variety of different contexts such as social, economic, and biological networks, among others [134, 90]. Although analysis of such networked models is not new (it can be traced back to the consensus problem studied more than four decades ago [48]), such models are studied recently from *the view-point of signal processing*, in which the analysis is based on the "graph operator" whose eigenvectors serve as a *Fourier basis for the graph* of interest [160, 151, 153, 152]. Examples of graph operators include the Laplacian matrices [160], the adjacency matrix [151], or a signal covariance matrix [114]. With the help of graph Fourier basis, a number of topics such as sampling and reconstruction, multirate filter banks, and uncertainty principles have been extended to the case of graphs in [113, 129, 126, 172, 173, 207, 2, 187]. Studies in [147, 146, 203] also use the graph operator for regression and learning over graphs.

Due to their rich structure, graphs present very interesting theoretical challenges and insights when analyzing data defined on them. This thesis aims at revealing fundamental limitations as well as opportunities brought by the underlying graph structure in the context of data and signal processing. In particular, this thesis has shown how the randomized asynchronicity of the interactions between nodes on graphs opens up new dimensions in our understanding of linear systems. The main contributions of this thesis can be summed up as follows:

1. *Random node-asynchronous computations:* The first part of this thesis focuses on asynchronous communications between network of agents. The thesis has modeled such communications as *random asynchronous fixed point iterations*, where the signal on the graph (data held by the agents) is expected to converge to a fixed point of the model. By carefully designing the update scheme, convergence of the signal opens up successful applications such as autonomous clustering of networks, asynchronous implementation of graph filters, and more. This thesis has analyzed the underlying model rigorously and provided the necessary and sufficient conditions for the mean-squared convergence under different scenarios depending on the input signal. In particular, it is proven that conditions for the mean-squared convergence is *more relaxed* than the condition for the convergence of synchronous updates. Such updates are also considered from linear system theory view-point, and it is proven that some unstable systems can get stabilized simply by randomly and asynchronously updating the state variables. These will be described in Chapters 2, 3, 4, and 5 of the thesis.

2. *Multirate processing of graph signals:* In this part, we have considered the extension of classical multirate processing ideas (downsampling, interpolation, analysis/synthesis filter banks, and so on) to the case of graphs. It is shown that the extension of classical multirate theory to graphs is nontrivial, and requires certain mathematical restrictions on the graph. The thesis provides the necessary conditions on the graph such that these ideas remain valid in the graph domain. In particular, it is shown that when the underlying graph satisfies a condition called M-block cyclic property, classical multirate theory can be extended to the graphs. These will be described in Chapter 6.

3. *Uncertainty principles on graphs:* This part of the thesis has presented a new way to formulate the uncertainty principle for signals and their Fourier transforms defined over graphs, by using a nonlocal measure based on the notion of sparsity. It is shown that the total number of nonzero elements of a graph signal and its graph Fourier transform is lower bounded by a quantity that depends on the underlying graph. It is also shown that graphs can have sparse eigenvectors, in which case the uncertainty bound is very low, tight, and independent of the global structure of the graph. These will be described in Chapter 7.

4. *Other problems in graph signal processing:* In this part of the thesis we

consider two additional problems in the context of graph signal processing. The first one considers the spectral concentration problem for polynomial graph filters. This is a graph extension of a well-studied problem for the optimal design of FIR filters, whose solution is known as the prolate sequence. This will be presented in Chapter 8. The second problem considers the heat diffusion process over the graph. In particular, we will assume that there is a point heat source that starts to diffuse at a known location but at an unknown time. We sample the heat distribution over the graph and consider the estimation of the starting time of the diffusion process when there is additive noise. This will be presented in Chapter 9.

The chapter outline is as follows. Section 1.1 defines the notation used in this thesis. Section 1.2 briefly reviews the area of graph signal processing and describes the basic notions, then Section 1.3 gives the outline and the scope of this thesis.

1.1 Notations

We will use $\mathbb{C}^N$ and $\mathbb{R}^N$ to denote N-dimensional complex and real vector spaces, respectively. Similarly, we will use $\mathbb{C}^{N\times M}$ and $\mathbb{R}^{N\times M}$ to denote the space of matrices of size $N \times M$ with complex and real entries, respectively. Vectors and matrices are denoted by lower-case letters in bold face (such as $\mathbf{x}$) and upper-case letters in bold face (such as $\mathbf{X}$), respectively.

We will use $\mathbf{I}$ to denote the identity matrix; we will use $\mathbf{0}$ denote the all-zeros vector, or matrix; we will use $\mathbf{1}$ to denote all-ones vector; we will use $\mathbf{e}_i$ to denote the i^{th} standard vector that has 1 at the i^{th} index and 0 elsewhere. Dimensions of these vectors and matrices should follow from the context.

For a given (possibly complex valued) matrix $\mathbf{X}$, we will use $\mathbf{X}^{\mathrm{T}}$ to denote its transpose, we will use $\mathbf{X}^{\mathrm{H}}$ to denote its conjugate-transpose, and we will use $\mathbf{X}^*$ to denote its element-wise conjugate.

For a matrix $\mathbf{X}$, we will use $X_{i,j}$, or $(\mathbf{X})_{i,j}$ to denote the $(i,j)^{th}$ element of the matrix. The i^{th} column of the matrix will be denoted as $\mathbf{x}_i$, and the i^{th} row of the matrix will be denoted as $\mathbf{x}_{(i)}$.

We will use $\mathcal{T}$ to denote a subset of $\{1, \cdots, N\}$. Given a subset $\mathcal{T}$, its corresponding index-selection matrix will be denoted as $\mathbf{D}_{\mathcal{T}} \in \mathbb{R}^{N\times N}$, which is a diagonal matrix that has value 1 only at the indices specified by the set $\mathcal{T}$. That is,

$$\mathbf{D}_{\mathcal{T}} = \sum_{i\in\mathcal{T}} \mathbf{e}_i\, \mathbf{e}_i^{\mathrm{H}}, \qquad \text{and} \qquad \mathrm{tr}(\mathbf{D}_{\mathcal{T}}) = |\mathcal{T}|, \tag{1.1}$$

where $|\mathcal{T}|$ denotes the size of $\mathcal{T}$.

We will use $\succ$ and $\succeq$ to denote the positive definite (PD) and the positive semi-definite (PSD) ordering, respectively. Namely, $\mathbf{X} \succeq \mathbf{Y}$ and $\mathbf{X} \succ \mathbf{Y}$ imply $\mathbf{X} - \mathbf{Y}$ is positive semi-definite and positive definite, respectively.

1.1.1 Norms

For a vector $\mathbf{x}$ we will use $\|\mathbf{x}\|_p$ to denote its ℓ_p-norm. Namely, $\|\mathbf{x}\|_p = \left(\sum_i |x_i|^p\right)^{(1/p)}$. In particular, $\|\mathbf{x}\|_\infty$ denotes the largest element of $\mathbf{x}$ in absolute sense. We will use $\|\mathbf{x}\|_0$ to denote the number of non-zero elements of the vector $\mathbf{x}$. Note that $\|\mathbf{x}\|_0$ is not a norm.

For a matrix $\mathbf{X}$ (possibly with complex entries), we will use $\sigma_{\min}(\mathbf{X})$ and $\sigma_{\max}(\mathbf{X})$ to denote its the smallest and the largest singular values, respectively. When $\mathbf{X}$ is a Hermitian matrix (i.e., $\mathbf{X}^{\mathrm{H}} = \mathbf{X}$), we will use $\lambda_{\min}(\mathbf{X})$ and $\lambda_{\max}(\mathbf{X})$ to denote its the smallest and the largest eigenvalues, respectively. We will use $\|\mathbf{X}\|_2$ to denote the spectral norm, i.e., the largest singular value. So, $\|\mathbf{X}\|_2 = \sigma_{\max}(\mathbf{X})$. We will use $\|\mathbf{X}\|_{\mathrm{F}}$ to denote the Frobenius norm; we will use $\|\mathbf{X}\|_\infty$ to denote the largest absolute row-sum; we will use $\|\mathbf{X}\|_{\max}$ to denote the largest element magnitude-wise; we will use $\mathrm{tr}(\mathbf{X})$ to denote the trace; we will use $\rho(\mathbf{X})$ to denote the spectral radius (the largest eigenvalue in magnitude) of the matrix $\mathbf{X}$.

1.1.2 Operations and Products

We will use $\mathbb{P}[\cdot]$ and $\mathbb{E}[\cdot]$ to denote the probability and expectation, respectively.

For a given matrix $\mathbf{X}$, we will use $|\mathbf{X}|$ to denote the matrix obtained by replacing the elements of $\mathbf{X}$ by their absolute values. Namely, $(|\mathbf{X}|)_{i,j} = |X_{i,j}|$.

For a matrix $\mathbf{X} \in \mathbb{C}^{M\times N}$, we will use $\mathrm{vec}(\mathbf{X}) \in \mathbb{C}^{MN}$ to denote the vector obtained by stacking the columns of $\mathbf{X}$. Namely,

$$\mathrm{vec}(\mathbf{X}) = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}. \tag{1.2}$$

We will use $\mathrm{vec}^{-1}(\cdot)$ to denote the inverse of the vectorization operator, where the dimensions follow from the context. Thus, $\mathrm{vec}^{-1}\left(\mathrm{vec}(\mathbf{X})\right) = \mathbf{X}$.

For a vector $\mathbf{x} \in \mathbb{C}^N$, we will use $\text{diag}(\mathbf{x}) \in \mathbb{C}^{N \times N}$ to denote the diagonal matrix with the i^{th} diagonal entry being the i^{th} index of the vector $\mathbf{x}$.

For a matrix $\mathbf{X} \in \mathbb{C}^{N \times N}$, we will use $\text{diag}(\mathbf{X}) \in \mathbb{C}^{N \times N}$ to denote the diagonal masking of $\mathbf{X}$, that is, $(\text{diag}(\mathbf{X}))_{i,i} = X_{i,i}$ and $(\text{diag}(\mathbf{X}))_{i,j} = 0$ for $i \neq j$. In other words, $\text{diag}(\mathbf{X}) = \mathbf{X} \odot \mathbf{I}$, where $\odot$ denotes the Hadamard product described below.

Given two matrices $\mathbf{X} \in \mathbb{C}^{N \times M}$ and $\mathbf{Y} \in \mathbb{C}^{N \times M}$, we will use $\mathbf{X} \odot \mathbf{Y}$ to denote the Hadamard product of the matrices. Namely, $(\mathbf{X} \odot \mathbf{Y})_{i,j} = X_{i,j}\, Y_{i,j}$.

We will use $\otimes$ to denote the Kronecker product with the following definition:

$$\mathbf{X} \otimes \mathbf{Y} = \begin{bmatrix} X_{1,1}\,\mathbf{Y} & \cdots & X_{1,M}\,\mathbf{Y} \\ \vdots & \ddots & \vdots \\ X_{N,1}\,\mathbf{Y} & \cdots & X_{N,M}\,\mathbf{Y} \end{bmatrix} \in \mathbb{C}^{(NP)\times(MQ)}, \tag{1.3}$$

where $\mathbf{X} \in \mathbb{C}^{N \times M}$ and $\mathbf{Y} \in \mathbb{C}^{P \times Q}$. We also note that the Kronecker product has the following mixed-product property for matrices $\mathbf{A}, \mathbf{B}, \mathbf{X}, \mathbf{Y}$ with conforming sizes:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{X} \otimes \mathbf{Y}) = (\mathbf{AX}) \otimes (\mathbf{BY}). \tag{1.4}$$

1.1.3 Graph Related Notation

For a given graph with N nodes, we will assume that $\mathbf{A} \in \mathbb{C}^{N \times N}$ is an operator on the graph. We consider $\mathbf{A}$ to be a *local* operator, that is, $A_{i,j} = 0$ when the nodes i and j are not neighbors. In particular, $A_{i,j}$ denotes the weight of the edge from the j^{th} node to the i^{th} node. We allow $A_{i,i}$ to be non-zero. Hence, the operator $\mathbf{A}$ can be the adjacency matrix, the Laplacian, the normalized Laplacian, and so on. We will use $\mathcal{N}_{\text{in}}(i)$ and $\mathcal{N}_{\text{out}}(i)$ to denote the incoming and outgoing neighbors of the i^{th} node. More precisely we have

$$\mathcal{N}_{\text{in}}(i) = \{j \mid A_{i,j} \neq 0\}, \qquad \mathcal{N}_{\text{out}}(i) = \{j \mid A_{j,i} \neq 0\}. \tag{1.5}$$

When the graph operator $\mathbf{A}$ is a diagonalizable matrix, we consider its eigenvalue decomposition as follows:

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{-1}, \tag{1.6}$$

where $\mathbf{V}$ is a matrix consisting of eigenvectors of $\mathbf{A}$, and $\mathbf{\Lambda}$ is the diagonal matrix with the eigenvalues, which may be complex in general. Given a signal $\mathbf{x}$, its graph Fourier transform (GFT), $\widehat{\mathbf{x}}$, on the operator $\mathbf{A}$ is defined as

$$\widehat{\mathbf{x}} = \mathbf{V}^{-1}\,\mathbf{x}, \qquad \text{or,} \qquad \mathbf{x} = \sum_{i=1}^{N} \widehat{x}_i\; \mathbf{v}_i, \tag{1.7}$$

where $\mathbf{v}_i$'s are the eigenvectors of $\mathbf{A}$. If the graph operator $\mathbf{A}$ is nondiagonalizable, we consider its Jordan decomposition and use its generalized eigenvectors as the graph Fourier basis.

1.2 A Brief Review of Graph Signal Processing

In this section we will present some of the fundamental concepts in graph signal processing. We will start from the notion of frequency in the graph setting and describe how the eigenvectors of the graph can be interpreted as a Fourier basis. Based on this interpretation, we will describe the notion of filtering. In particular, we will explain polynomial graph filters by drawing parallels to the FIR filters in the classical signal processing. Then, we will describe how the notion of "graph shift" plays a central role in the theoretical understanding as well as in distributed processing applications. This section is intended to be a brief overview, and we refer to [160, 151, 138] for more detailed introduction to the field.

In order to present the main ideas concisely in this section, we will assume that graphs have undirected edges with non-negative edge weights.

1.2.1 Basic Graph Definitions

For a given graph with N nodes, we will use $A_{i,j} \geq 0$ to denote the weight of the edge between the node i and the node j. When $A_{i,j} = 0$, it implies that the nodes i and j are not connected (i.e., there is no edge between them). We use *the adjacency matrix*, denoted with $\mathbf{A} \in \mathbb{R}^{N\times N}$, to represent the connectivity structure of the graph in a matrix form. Namely, the $(i,j)^{th}$ entry of the matrix $\mathbf{A}$ is given by $A_{i,j}$. Since the graph is assumed to be undirected (i.e., edges are bi-directional) we have $A_{i,j} = A_{j,i}$ so that $\mathbf{A}$ is a symmetric matrix.

The degree of a node is the total weight of the edges connected to that node. Notice that when the graph is unweighted (i.e., edge weights are either 0 or 1), the degree of a node is the number of edges connected to that node. More precisely, the degree of the node i is defined as follows:

$$d_i = \sum_{j=1}^{N} A_{i,j}. \tag{1.8}$$

Then, *the degree matrix* is defined as a diagonal matrix with the i^{th} diagonal entry being the degree of the node i. More precisely,

$$\mathbf{D} = \operatorname{diag}\Big(\begin{bmatrix} d_1 & d_2 & \cdots & d_N \end{bmatrix}\Big) \in \mathbb{R}^{N\times N}. \tag{1.9}$$

Based on the adjacent matrix and the degree matrix, *the graph Laplacian matrix* is defined as follows:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}. \tag{1.10}$$

Then, *the normalized graph Laplacian* has the following definition:

$$\mathcal{L} = \mathbf{D}^{-1/2}\, \mathbf{L}\, \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\, \mathbf{A}\, \mathbf{D}^{-1/2}. \tag{1.11}$$

Graph Laplacians possess many interesting properties. In fact, their spectral properties are tightly connected to the algebraic structure of the underlying graph. This is part of a well-studied field known as spectral graph theory [39].

As a final note, we would like to mention that both Laplacians are positive semi-definite matrices, whose eigenvalues can be bounded as follows:

$$\mathbf{0} \preceq \mathbf{L} \preceq 2 \left(\max_i d_i \right) \mathbf{I}, \qquad\qquad \mathbf{0} \preceq \mathcal{L} \preceq 2\,\mathbf{I}. \tag{1.12}$$

1.2.2 Quadratic Variation

For a given graph with N nodes, let $x_i \in \mathbb{C}$ denote the value associated with the i^{th} node of the graph. Then, the corresponding *signal on the graph* is a vector of length N consisting of the values associated to the nodes. More precisely,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \in \mathbb{C}^N. \tag{1.13}$$

So, when we say $\mathbf{x}$ is a signal on the graph, it means that the i^{th} element of $\mathbf{x}$ is the value held by the i^{th} node.

For a given graph and a signal on the graph, the (normalized) quadratic variation of the signal on the graph is given as follows:

$$s(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|_2^2} \sum_{i \geq j} A_{i,j}\ |x_i - x_j|^2, \tag{1.14}$$

where the normalization with $\|\mathbf{x}\|_2^2$ is included to make sure that the variation of a signal is not increased simply by scaling the signal. Thus, $s(\alpha\mathbf{x}) = s(\mathbf{x})$ for any $\alpha \neq 0$. In words, the quadratic variation is the sum of all the squared differences in the values of *adjacent nodes* weighted by the edge connecting the nodes. If the adjacent nodes have similar values, then corresponding graph signal has a smaller quadratic variation. Non-adjacent nodes having different values does not contribute to the variation. As a result, the variation of a signal is tightly connected to the underlying graph structure.

Simple rearrangement of the terms in (1.14) reveals that the quadratic variation can be equivalently represented as follows:

$$s(\mathbf{x}) = \frac{\mathbf{x}^{\mathrm{H}}\,\mathbf{L}\,\mathbf{x}}{\mathbf{x}^{\mathrm{H}}\,\mathbf{x}} \geq 0, \tag{1.15}$$

which is known as *the Rayleigh quotient*. We note that the all ones vector **1** belongs to the null-space of the graph Laplacian, i.e., $\mathbf{L}\,\mathbf{1} = \mathbf{0}$. So, a signal $\mathbf{x}$ has a zero variation over the network when $\mathbf{x} = c\mathbf{1}$ for some nonzero c, i.e., all the nodes have the same value c.

It is also worth noting that **1** is not an eigenvector of the normalized Laplacian matrix in general. Instead, the normalized Laplacian satisfies $\mathcal{L}\,\mathbf{d} = \mathbf{0}$, where the i^{th} element of the vector $\mathbf{d}$ is the square-root of the degree of the i^{th} node. That is, $(\mathbf{d})_i = \sqrt{d_i}$. If the graph is regular (i.e., all the nodes have the same degree), then the normalized Laplacian satisfies $\mathcal{L}\,\mathbf{1} = \mathbf{0}$.

1.2.3 Notion of Frequency

Given the definition of the quadratic variation in (1.14), it is natural to look for the signals that have the largest (or, the smallest) amount of variation over the network. In this regard, we can consider the following optimization problem:

$$\max_{\mathbf{x}} \; s(\mathbf{x}), \tag{1.16}$$

whose solution is given as the largest eigenvalue of the graph Laplacian **L**.

For a given graph Laplacian, we consider its eigenvalue decomposition as follows:

$$\mathbf{L} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{\mathrm{H}} = \sum_{i=1}^{N} \lambda_i\,\mathbf{v}_i\,\mathbf{v}_i^{\mathrm{H}}, \qquad \text{where} \qquad \mathbf{v}_i^{\mathrm{H}}\,\mathbf{v}_j = \delta(i-j), \tag{1.17}$$

where it is assumed that eigenvalues are in increasing order:

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N. \tag{1.18}$$

Then, the maximum value and the maximizer of the problem (1.16) are as follows:

$$\lambda_N = \max_{\mathbf{x}} \; s(\mathbf{x}), \qquad\qquad \mathbf{v}_N = \arg\max_{\mathbf{x}} \; s(\mathbf{x}). \tag{1.19}$$

Moreover, the quadratic variation of an eigenvector of the graph Laplacian is equivalent to the corresponding eigenvalue of the eigenvector. Namely,

$$s(\mathbf{v}_i) = \lambda_i. \tag{1.20}$$

So, an eigenvector has a larger amount of variation over the graph when its corresponding eigenvalue is larger. We will illustrate this point in Figure 1.1 next.

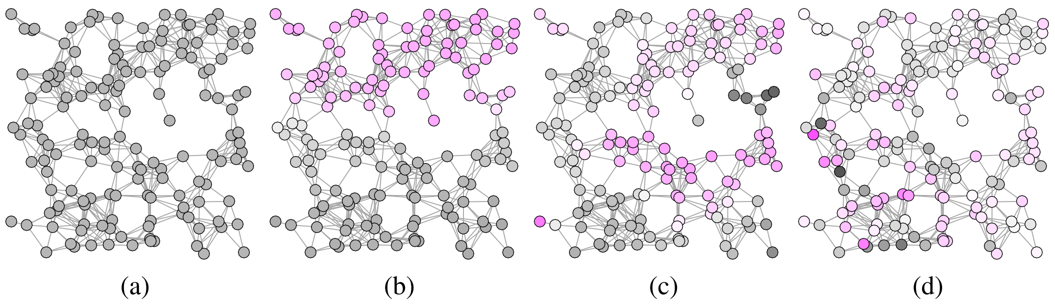


Figure 1.1: Visualization of some of the eigenvectors of the graph Laplacian on the graph. Colors black and pink represent positive and negative values, respectively. Intensity of a color represents the magnitude. (a) $\mathbf{v}_1$, (b) $\mathbf{v}_2$, (c) $\mathbf{v}_{10}$, (d) $\mathbf{v}_{50}$.

Figure 1.1 provides an example of a random geometric graph on $N = 150$ nodes, in which nodes are distributed over the region $[0\ 1] \times [0\ 1]$ uniformly at random. Two nodes are connected to each other if the distance between them is less than 0.15, and the graph is undirected. There are 650 unweighted edges on the graph. In Figures 1.1a, 1.1b, 1.1c, 1.1d, we visualize the eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_{10}$, $\mathbf{v}_{50}$, respectively, of the graph Laplacian. The figure visually shows that the variation of an eigenvector increases as its eigenvalue increases.

In addition to the clear relationship between the quadratic variation of an eigenvector and its corresponding eigenvalue as given in (1.20), the eigenvalues are tightly connected to different definitions of variations as well. In order to demonstrate this point, we consider the following two quantities:

$$z(\mathbf{x}) = \sum_{i \geq j} \mathbb{1}_{\{x_i x_j < 0\}}, \qquad s_1(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|_1} \sum_{i \geq j} A_{i,j}\ |x_i - x_j|, \tag{1.21}$$

where $s_1(\mathbf{x})$ considers the absolute difference between the values of the adjacent nodes, and $z(\mathbf{x})$ counts the number of "zero crossings" in the signal $\mathbf{x} \in \mathbb{R}^N$. Here, a zero-crossing is defined as two adjacent nodes having values with opposite signs.

Figure 1.2 presents variations of the eigenvectors (of the graph in Figure 1.1) quantified as in (1.21). In particular, Figure 1.2a shows the number of zero-crossings of an

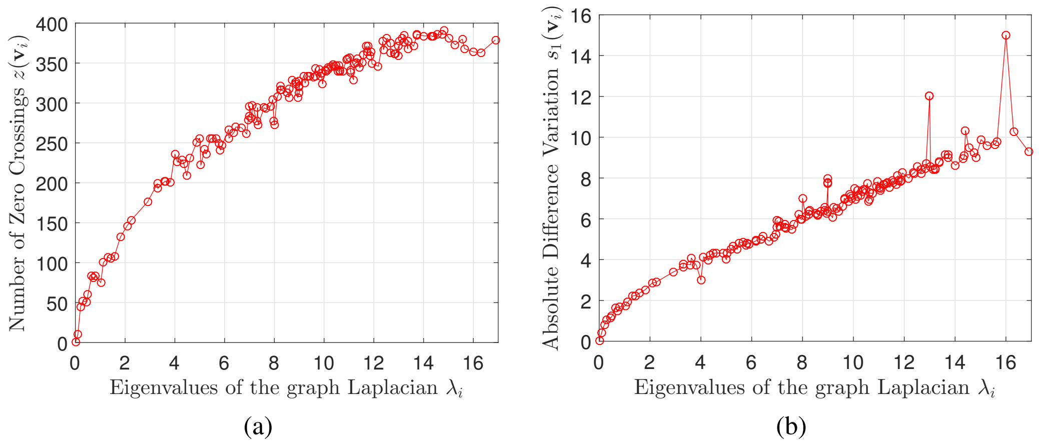


Figure 1.2: For a given eigenvalue-eigenvector pair, (a) number of zero-crossings, (b) absolute difference based variation of eigenvectors of the graph in Figure 1.1.

eigenvector as a function its eigenvalue, and Figure 1.2b shows the sum of absolute differences in an eigenvector (normalized with the ℓ_1-norm of the vector) as a function its eigenvalue. For a given eigenpair $(\mathbf{v}, \lambda)$ (of the graph Laplacian) the figure shows that the relationship between $z(\mathbf{v})$, $s_1(\mathbf{v})$ and λ is not monotonic. Namely, a larger eigenvalue does not necessarily imply that the corresponding eigenvector has a larger number of zero crossings (or, larger absolute differences). Nevertheless, the figure clearly shows that there is a very strong correlation between how much an eigenvector varies over the graph and its corresponding eigenvalue.

Based on the relation in (1.20) and the observations in Figure 1.2, it is possible to interpret an eigenvalue as the *frequency* of the corresponding eigenvector. Thus, eigenvectors with large eigenvalues correspond to high-frequency components, and eigenvectors with small eigenvalues correspond to low-frequency (slowly varying) components. This interpretation is also consistent with the fact that the smallest eigenvalue of a graph Laplacian is $\lambda_1 = 0$ with the corresponding eigenvector $\mathbf{v}_1 = \mathbf{1}$. Since $\mathbf{v}_1$ does not vary over a graph, it is meaningful to interpret $\mathbf{v}_1$ to have zero frequency. Based on this notion it is possible to construct a Fourier transform on the graph, as we shall elaborate next.

1.2.4 Eigenvectors as a Fourier Basis on the Graph

The Fourier transform is one of the most fundamental tools used in the area of signal processing with vastly different application areas. Origins of the Fourier transform go back to partial differential equations describing the diffusion of heat. In order to solve the heat equation for an arbitrary function, Fourier's intention was to represent the given function as a linear combination of sines and cosines, as sine and cosine were known to be "simple" solutions of the heat equation [53].

More precisely, consider the the following differential equation:

$$\frac{\mathrm{d}^2}{\mathrm{d}\,t^2}\, f(t) = -\omega^2\, f(t), \tag{1.22}$$

where a second-order differential operator is known as the *Laplace operator*. We note that the Laplace operator appears in many mathematical models describing physical phenomena such as heat diffusion. The solution to (1.22) can be obtained as follows:

$$f(t) = c_1\, e^{j\omega t} + c_2\, e^{-j\omega t}, \tag{1.23}$$

which shows that complex exponentials in the form of $e^{j\omega t}$ are *eigenfunctions* of the Laplace operator with the corresponding eigenvalue $-\omega^2$. So, the Fourier transform corresponds to the representation of a signal $x(t)$ in terms of the eigenfunctions of the Laplace operator. Namely,

$$x(t) = \int \widehat{x}(\omega)\, e^{j\omega t}\, \mathrm{d}\,\omega, \qquad \text{where} \qquad \widehat{x}(\omega) = \int e^{-j\omega t}\, x(t)\, \mathrm{d}\,t. \tag{1.24}$$

We now note that the graph Laplacian matrix defined in (1.10) can be considered as a discretized version of a differential operator over the graph. Namely,

$$\big(\mathbf{L}\,\mathbf{x}\big)_i = \sum_j A_{i,j}\, (x_i - x_j). \tag{1.25}$$

Thus, signals that remain "invariant" under the operator $\mathbf{L}$ satisfy the following difference equation:

$$\sum_j A_{i,j}\, (x_i - x_j) = \lambda\, x_i \qquad \forall i, \tag{1.26}$$

whose solutions correspond to the eigenvectors of the graph Laplacian matrix $\mathbf{L}$.

Drawing parallels to the original intention of Fourier transform, a given graph signal $\mathbf{x}$ can be represented in terms of the eigenvectors of the graph Laplacian matrix:

$$\mathbf{x} = \sum_{i=1}^{N} \widehat{x}_i\, \mathbf{v}_i = \mathbf{V}\, \widehat{\mathbf{x}}, \qquad \text{where} \qquad \widehat{x}_i = \mathbf{v}_i^{\mathrm{H}}\, \mathbf{x}, \tag{1.27}$$

where $\widehat{\mathbf{x}} = \mathbf{V}^{\mathrm{H}}\, \mathbf{x}$ is said to be the *graph Fourier transform (GFT)* of $\mathbf{x}$ on the graph.

We note that the graph Fourier coefficient $\widehat{x}_i$ is an inner product between the i^{th} eigenvector $\mathbf{v}_i$ and the graph signal $\mathbf{x}$ in a way similar to the Fourier transform $\widehat{x}(\omega)$

being an inner product between the eigenfunction $e^{j\omega t}$ and the signal $x(t)$. Since the graph Laplacian is a PSD matrix, it has an orthonormal set of eigenvectors that span the entire $\mathbb{C}^N$. Therefore, graph Fourier transform exists for any signal $\mathbf{x}$.

We also note that the frequency interpretation of eigenvalues discussed in Section 1.2.3 is consistent with the definition of the graph Fourier transform. It is clear from (1.22) that the eigenvalue of the eigenfunction $e^{j\omega t}$ of the Laplace operator is related to the frequency of the eigenfunction. So, it is meaningful to treat eigenvalues of the graph Laplacian as the frequency of the corresponding eigenvectors.

Although the definition of the graph Fourier transform in (1.27) is consistent with the classical notion of Fourier transform, we will mention three points that require some attention:

1) Eigenvectors have a sign (or, phase) ambiguity due to their definition. It is even possible for graphs to have repeated eigenvalues so that eigenspaces have dimension more than one. Therefore, a graph may have multiple sets of eigenvectors resulting in multiple choices for the graph Fourier transform. When discussing the graph Fourier transform, it is generally assumed that the set of eigenvectors is fixed and the GFT is defined with respect to that fixed set of eigenvectors. More generally, it is possible to extend the definition of graph Fourier transform using projections on invariant subspaces of the graph Laplacian matrix [49].

2) Although the discussion in this section is based on the graph Laplacian, similar arguments and interpretations hold true for the normalized graph Laplacian and the adjacency matrix. In Figures 1.3a and 1.3b, we demonstrate the number of zero crossings of the eigenvectors of the normalized graph Laplacian and the adjacency matrix, respectively, for the graph in Figure 1.1. It is clear from the figure that eigenvalues strongly correlate with the "variations" of the corresponding eigenvectors. For a given graph, the choice of graph matrix (adjacency, Laplacian, normalized Lapacian, etc.), and the choice of a specific set of eigenvectors, is usually guided by the application at hand [160, 151, 138].

3) It is, so far, assumed that the underlying graph has undirected edges, which in turn implies that the graph matrices are symmetric. So, eigenvectors can be selected to be orthonormal, and they span the entire $\mathbb{C}^N$. When the graph has directed edges, its matrix representation is no longer symmetric. If the matrix representation has a linearly independent set of eigenvectors (i.e., diagonalizable), then its eigenvectors can still be used as a Fourier basis on the graph. However, in the extreme case

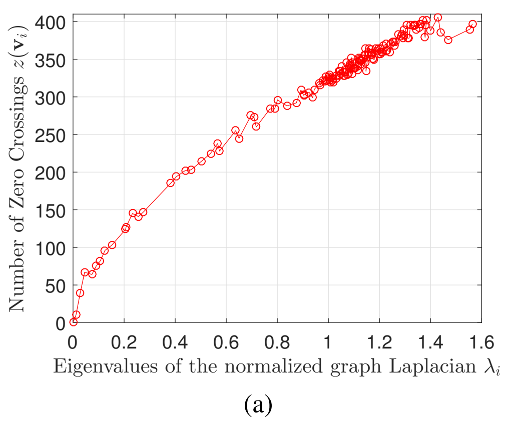


Figure 1.3: Number of zero crossings of the eigenvectors of (a) the normalized graph Laplacian, (b) the adjacency matrix as a function of their corresponding eigenvalues of the graph in Figure 1.1.

of nondiagonalizable graph matrices, the use of generalized eigenvectors (i.e., the Jordan form) has been proposed [153].

1.2.5 Notion of Filtering Over Graphs

Given the definition of the graph Fourier transform in (1.27), it is readily shown that the quadratic variation of a graph signal $\mathbf{x}$ can be written in terms of its graph Fourier transform as follows:

$$s(\mathbf{x}) = \left(\sum_{i=1}^{N} \lambda_i \, |\widehat{x}_i|^2 \right) \Big/ \left(\sum_{i=1}^{N} |\widehat{x}_i|^2 \right). \tag{1.28}$$

Here, the quantity $|\widehat{x}_i|^2$ denotes how much energy the signal $\mathbf{x}$ has on the i^{th} graph Fourier basis component, and λ_i denotes the variation of the i^{th} basis component. When the energy of a signal $\mathbf{x}$ is confined in "low-frequency" components of the graph Fourier basis, the signal has a lower amount of variation on the graph. Based on this observation, it is possible to "smooth" a graph signal in a way similar to low-pass filtering. This can be achieved if a given graph signal is "filtered" in such a way that it has less energy on the high-frequency components.

More precisely, given a graph signal $\mathbf{x}$ consider the signal $\mathbf{y}$ constructed as follows:

$$\widehat{y}_i = \begin{cases} \widehat{x}_i, & i \le M, \\ 0, & i > M, \end{cases} \tag{1.29}$$

where M can be considered as the "cut-off" point. Then, it is readily verified that the signal $\mathbf{y}$ is smoother on the graph than the signal $\mathbf{x}$. That is,

$$s(\mathbf{y}) = \left(\sum_{i=1}^{M} \lambda_i \, |\widehat{x}_i|^2\right) \Big/ \left(\sum_{i=1}^{M} |\widehat{x}_i|^2\right) \leq s(\mathbf{x}). \tag{1.30}$$

One can also show that the residual $\mathbf{x} - \mathbf{y}$ has a higher amount of variation than the signal $\mathbf{x}$, that is, $s(\mathbf{x} - \mathbf{y}) \geq s(\mathbf{x})$ under the assumption that $\mathbf{x} \neq \mathbf{y}$.

The relation between $\mathbf{y}$ and $\mathbf{x}$ defined in (1.29) can also be represented as follows:

$$\mathbf{y} = \mathbf{H}\,\mathbf{x}, \qquad \text{where} \qquad \mathbf{H} = \mathbf{V}\,\mathbf{D}_M\,\mathbf{V}^{\mathrm{H}}, \tag{1.31}$$

and $\mathbf{D}_M$ denotes a diagonal matrix with only the first M diagonal entries being one, and the rest being zero. Drawing parallels to classical signal processing, the matrix $\mathbf{H}$ can be considered as an *ideal low-pass filter on the graph*, and the signal $\mathbf{y}$ can be considered as a *low-pass filtered* version of the signal $\mathbf{x}$.

The effect of low-pass filtering is presented in Figure 1.4, where we visualize a graph signal $\mathbf{x}$ in Figure 1.4a, and the output of the ideal low-pass filters in Figures 1.4b and 1.4c with the cut-off $M = 40$ and $M = 5$, respectively. It is clear that as the cut-off point decreases, the filtered signal becomes smoother on the graph.

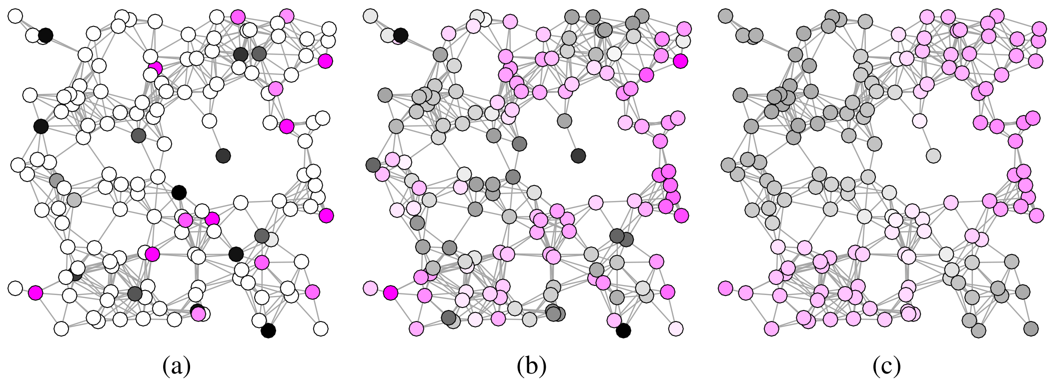


Figure 1.4: (a) A graph signal $\mathbf{x}$ on the graph with $N = 150$ nodes. Output of the ideal low-pass filtering with the cut-off (b) $M = 40$, (c) $M = 5$.

1.2.6 Polynomial Filters and the Graph Shift

Although the ideal low-pass graph filter defined in (1.31) is well motivated, it has some limitations in terms of practical implementations. This is due to the fact that eigenvectors of the graph Laplacian matrix need to be computed in order to construct the matrix $\mathbf{H}$. When the graph consists of too many nodes (i.e., N is very large), the computation of eigenvectors becomes infeasible. Furthermore, the overall filtering operation requires a centralized (or, off-line) computation, and the filter matrix $\mathbf{H}$ may not be sparse even when the graph Laplacian is.

The main approach in addressing these problems is to use a *polynomial approximation* of the ideal-filter. That is, a set of coefficients are selected in such a way that a polynomial (of order K) of the graph Laplacian is a good approximation of the ideal filter. More precisely,

$$H(\mathbf{L}) = \sum_{n=0}^{K} h_n \, \mathbf{L}^n \approx \mathbf{H}. \tag{1.32}$$

In the graph Fourier domain (i.e., the spectral domain) the polynomial approximation can be described as follows:

$$H(\lambda_i) \approx \begin{cases} 1, & i \leq M, \\ 0, & i > M. \end{cases} \tag{1.33}$$

Here, $H(\lambda_i)$ can be considered as the "response" of the polynomial filter, as it scales the i^{th} component of graph Fourier transform. Namely, $H(\mathbf{L})\,\mathbf{x} = \sum_i H(\lambda_i)\, \widehat{x}_i \, \mathbf{v}_i$. So, the coefficients $\{h_0, \cdots, h_K\}$ are selected in such a way that the response of the polynomial filter approximates the ideal response. This is indeed very similar to the design of FIR filters in classical signal processing. The only difference is that frequencies correspond to the unit circle in classical signal processing, whereas frequencies lie on the real line in the graph case.

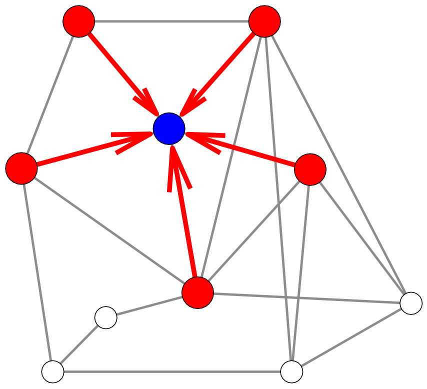


Figure 1.5: The graph shift is equivalent to a node collecting information from its neighbors.

The main motivation for polynomial filtering follows from its localized and distributed implementation over a graph. Namely, consider the term $\mathbf{L}\,\mathbf{x}$. Here, the multiplication with the matrix $\mathbf{L}$ is said to be "graph shift," where the notion of "shift" is equivalent to the adjacent nodes communicating with each other, as the shifted signal **Lx** can be written explicitly as in (1.25). See Figure 1.5 for an illustrative example. So, the quantity **Lx** has a distributed implementation requiring a communication between only the adjacent nodes.

Since a polynomial filtering of a signal **x** can be written as follows:

$$H(\mathbf{L})\,\mathbf{x} = h_0\,\mathbf{x} + h_1\,\mathbf{L}\,\mathbf{x} + h_2\,\mathbf{L}^2\,\mathbf{x} + \cdots + h_K\,\mathbf{L}^K\,\mathbf{x}, \tag{1.34}$$

an order K polynomial filter is equivalent to K successive communications between the adjacent nodes (i.e., the graph shift) and computing a weighted average according to the coefficients of the filter. The shift-and-add structure of polynomial graph filters resembles the direct form implementation of FIR digital filters in classical signal processing [200]. So, polynomial graph filters can be thought of as an extension of digital FIR filters to the case of graphs. We will discuss more about polynomial filters later in Section 6.3 of Chapter 6.

1.2.7 An Asynchronous Graph Shift

Although the graph shift translates as a localized communication between neighboring nodes, the graph shift forces all nodes to communicate simultaneously. When consecutive graph shifts are required (which is the key point in polynomial graph filtering), these localized communications need to be synchronized, which causes a delay in large networks, or it may not be even possible in the case of autonomous networks. In order to eliminate this limitation, an *asynchronous variant* of the graph shift is considered as follows [181, 176]:

$$(\mathbf{y})_i = \begin{cases} (\mathbf{L}\,\mathbf{x})_i, & i \in \mathcal{T}, \\ (\mathbf{x})_i, & i \notin \mathcal{T}, \end{cases} \tag{1.35}$$

which is very similar to the synchronous graph shift, i.e., $\mathbf{y} = \mathbf{L}\,\mathbf{x}$, except for the fact that only the nodes indexed by the set $\mathcal{T}$ update their values, and the remaining ones stay unchanged. So, consecutive asynchronous shifts still have a distributed implementation on the graph, but no synchronization is required among the nodes. Furthermore, these type of updates are equivalent to the nodes communicating with their neighbors repetitively at random time instances when the update set $\mathcal{T}$ is selected randomly.

Although asynchronous graph shift appears to be very similar to the synchronous one, the dynamical behavior of randomized asynchronous updates is, in fact, very different, and classical results from synchronous linear system theory no longer apply. In order to demonstrate this point consider an eigenpair of the graph Laplacian matrix, i.e., $\mathbf{L}\mathbf{v} = \lambda\,\mathbf{v}$, and observe the following:

$$\mathbf{v} \xrightarrow{\text{synchronous}} \underbrace{\begin{bmatrix} \lambda\, v_1 \\ \lambda\, v_2 \\ \lambda\, v_3 \\ \vdots \\ \lambda\, v_N \end{bmatrix}}_{\text{an eigenvector}}, \qquad \mathbf{v} \xrightarrow{\text{asynchronous}} \underbrace{\begin{bmatrix} v_1 \\ \lambda\, v_2 \\ v_3 \\ \vdots \\ v_N \end{bmatrix}}_{\text{not an eigenvector}}, \tag{1.36}$$

where the asynchronous shift updates only the second index of the vector.

When an eigenvector is shifted synchronously, it remains as the eigenvector by definition. On the contrary, the output of an asynchronous shift is no longer an eigenvector even when the input is. Thus, asynchronous updates do not respect eigenvectors of the matrix **L**. This behavior has a profound effect on stability of the random asynchronous updates. Chapters 2, 3, and 4 of this thesis focus on a randomized and asynchronous graph variant of the graph shift, study its statistical behavior, and present applications in the context of graph signal processing.

1.3 Outline and Scope of the Thesis

This thesis studies various aspects of signal processing techniques in network settings. The first part of the thesis (Chapters 2, 3, 4, and 5) focuses on randomized and asynchronous variants of state recursions and studies their behavior in a statistical sense. The main theoretical finding of these sections is that random asynchronous implementations can stabilize systems that are otherwise unstable. The second part of the thesis studies extensions (and re-interpretation) of classical signal processing techniques to the case of graphs. Namely, Chapter 6 considers multirate signal processing techniques (and filter banks) for the graph case. Chapter 7 visits uncertainty principles and shows that time-frequency localization phenomena does not always extend to the case of graphs. Chapter 8 studies the optimal polynomial filter design problem (that maximizes the bass-band energy) for the case of graphs. Finally, Chapter 9 considers the heat-diffusion process over networks, and studies the estimation of the starting time of a diffusion. In this section, the scope of each chapter will be briefly outlined.

1.3.1 Random Node-Asynchronous Updates (Chapter 2)

This chapter introduces a node-asynchronous communication protocol in which an agent in a network wakes up randomly and independently, collects states of its neighbors, updates its own state, and then broadcasts back to its neighbors. This protocol differs from consensus algorithms and it allows distributed computation of an arbitrary eigenvector of the network, in which communication between agents is allowed to be directed. In order to analyze the scheme, this chapter studies a random asynchronous variant of the power iteration where the update matrix is selected to be the graph operator of interest. Under this random asynchronous model, an initial signal is proven to converge to an eigenvector of eigenvalue 1 (a fixed point) even in the case of operator having spectral radius larger than unity. In particular, the convergence region for the eigenvalues gets larger as the updates get less synchronous. The rate of convergence is shown to depend not only on the eigenvalue gap but also on the eigenspace geometry of the operator as well as the amount of asynchronicity of the updates. In particular, the rate of convergence is affected by the phase of the eigenvalues, and the randomized updates favor negative eigenvalues over positive ones. Random asynchronous updates are also interpreted from the graph signal perspective, and it is shown that a non-smooth signal on the graph converges to the smoothest signal under the random model. Polynomials of the operator are used to achieve convergence to an arbitrary eigenvector of the operator. When the eigenvalues are real, second order polynomials are shown to be sufficient for this. Using second order polynomials in the randomized update model, the chapter formalizes the node-asynchronous communication model whose convergence is readily proven. As an application, the protocol is used to compute the Fiedler vector of a network to achieve autonomous clustering. As another application, this chapter considers a reformulation of the component-wise updates revealing a randomized algorithm that is proven to converge to the dominant left and right singular vectors of a normalized data matrix. The algorithm is also extended to handle large-scale distributed data when computing an arbitrary rank approximation of an arbitrary data matrix. Numerical simulations verify the convergence of the proposed algorithms under different parameter settings.

1.3.2 IIR Filtering with Random Node-Asynchronous Updates (Chapter 3)

This chapter proposes a node-asynchronous implementation of rational filters on arbitrary graphs. In the proposed algorithm nodes follow a randomized collect-compute-broadcast scheme: if a node is in the passive stage it collects the data sent by its incoming neighbors and stores only the most recent data. When a node gets into the active stage at a random time instance, it does the necessary filtering computations locally, and broadcasts a state vector to its outgoing neighbors. For the analysis of the algorithm, this chapter first considers a general case of randomized asynchronous state recursions and presents a sufficiency condition for its convergence. Based on this result, the proposed algorithm is proven to converge to the filter output in the mean-squared sense when the filter, the graph operator and the update rate of the nodes satisfy a certain condition. The proposed algorithm is simulated using rational and polynomial filters, and its convergence is demonstrated for various different cases, which also shows the robustness of the algorithm to random communication failures.

1.3.3 Random Asynchronous Linear Systems (Chapter 4)

This chapter extends the random asynchronous state recursions studied in Chapters 2 and 3 to a setting where the input is time-dependent, such as complex sinusoids. It is based on a randomized asynchronous variant of linear discrete-time state-space models, in which each state variable gets updated with a non-zero probability independently (and asynchronously) in every iteration. This chapter shows that such randomized systems behave very similar to the synchronous non-random counterpart in a statistical sense. In particular, the output of the randomized system with a sinusoidal input is still a sinusoid in expectation with the same frequency. So, it is possible to consider the "frequency response" of such systems. This chapter also presents the necessary and sufficient condition for the mean-squared stability of the randomized system. It is shown that stability of the underlying state transition matrix is neither necessary nor sufficient for the mean-squared stability of the randomized asynchronous recursions. However, randomization introduces an error depending on the update probabilities and the amount of variation (frequency) in the input signal. It is also shown that eigenvectors (and not just eigenvalues) of the state transition matrix are important in determining the stability of the randomized system, i.e., stability (or instability) property can be altered with a similarity transform. The chapter also revisits the special case of constant input, in which case the randomized system becomes randomized fixed-point iterations studied in Chapter 3 and stability conditions are less restrictive than in the non-random case.

1.3.4 Randomized Algorithms as Switching Systems (Chapter 5)

This chapter considers the random asynchronous update model studied in Chapters 2, 3, and 4 from the viewpoint of switching systems and shows that convergence (and stability) properties of these models follow directly from the stability theory already developed for switching systems. The chapter further shows that randomized versions of Kaczmarz's method, Gauss-Seidel iterations, and asynchronous fixed-point iterations can be represented as specific instances of randomly switching systems. Then, the chapter presents alternative proofs for the mean-squared and almost sure convergence of randomized Kaczmarz and Gauss-Seidel methods. The necessary and sufficient condition for the mean-squared convergence of random asynchronous fixed-point iterations is also provided.

1.3.5 Extending Classical Multirate Signal Processing Theory (Chapter 6)

This chapter extends classical multirate signal processing ideas to graphs and revisits ideas such as noble identities, aliasing, and polyphase decompositions in graph multirate systems. It is shown that the extension of classical multirate theory to graphs is nontrivial, and requires certain mathematical restrictions on the graph. For example, classical noble identities cannot be taken for granted. Similarly, one cannot claim that the so-called delay chain system is a perfect reconstruction system (as in classical filter banks). It will also be shown that M-partite extensions of the bipartite filter bank results will not work for M-channel filter banks, but a more restrictive condition called M-block cyclic property should be imposed. Such graphs are studied in detail. Building upon the basic theory of multirate systems for graph signals, M-channel polynomial filter banks on graphs are studied. The behavior of such graph filter banks differs from that of classical filter banks in many ways, the precise details depending on the eigenstructure of the adjacency matrix. If the adjacency matrix is actually M-block cyclic then perfect-reconstruction (PR) filter banks become practical, i.e., arbitrary filter polynomial orders are possible, and there are robustness advantages. In this case the PR condition is identical to PR in classical filter banks – any classical PR example can be converted to a graph PR filter bank on an M-block cyclic graph. Polyphase representations are developed for graph filter banks and utilized to develop alternate conditions for alias cancellation and perfect reconstruction, again for graphs with specific eigenstructures. It is then shown that the eigenvector condition on the graph can be relaxed by using similarity transforms.

1.3.6 Uncertainty Principles and Sparse Eigenvectors (Chapter 7)

This chapter advances a new way to formulate the uncertainty principle for signals defined over graphs, by using a non-local measure based on the notion of sparsity. To be specific, the total number of nonzero elements of a graph signal and its corresponding graph Fourier transform (GFT) is considered. A theoretical lower bound for this total number is derived, and it is shown that a nonzero graph signal and its GFT cannot be arbitrarily sparse simultaneously. When the graph has repeated eigenvalues, the graph Fourier basis (GFB) is not unique. Since the derived lower bound depends on the selected GFB, a method that constructs a GFB with the minimal uncertainty bound is provided. In order to find signals that achieve the derived lower bound (i.e. the most compact on the graph and in the GFB), sparse eigenvectors of the graph are investigated. It is shown that a connected graph has a 2-sparse eigenvector (of the graph Laplacian) when there exist two nodes with the same neighbors. In this case the uncertainty bound is very low, tight, and independent of the global structure of the graph. For several examples of classical and real-world graphs it is shown that 2-sparse eigenvectors, in fact, exist.

1.3.7 Energy Compaction Filters (Chapter 8)

In classical signal processing spectral concentration is an important problem that was first formulated and analyzed by Slepian. The solution to this problem gives the optimal FIR filter that can confine the largest amount of energy in a specific bandwidth for a given filter order. The solution is also known as the prolate sequence. This chapter investigates the same problem for polynomial graph filters. The problem is formulated in both graph-free and graph-dependent fashions. The graph-free formulation assumes a continuous graph spectrum, in which case it becomes the polynomial concentration problem. This formulation has a universal approach that provides a theoretical reference point. However, in reality graphs have discrete spectrum. The graph-dependent formulation assumes that the eigenvalues of the graph are known and formulates the energy compaction problem accordingly. When the eigenvalues of the graph have a uniform distribution, the graph-dependent formulation is shown to be asymptotically equivalent to the graph-free formulation. However, in reality eigenvalues of a graph tend to have different densities across the spectrum. Thus, the optimal filter depends on the underlying graph operator, and a filter cannot be universally optimal for every graph.

1.3.8 Time Estimation for Heat Diffusion (Chapter 9)

This chapter studies the estimation of the starting time of a diffusion process from its noisy measurements when there is a single point source located on a known vertex of a graph with unknown starting time. The diffusion process is assumed to be governed by the heat equation. In particular, the Cramér-Rao lower bound (CRLB) for the problem is derived. It is shown that the problem has a larger CRLB for graphs with higher connectivity. Closed form expression of the bound is derived for some graphs. The Maximum Likelihood estimator is numerically verified to be unbiased, and achieves the CRLB for some graphs.

Chapter 2

RANDOM NODE-ASYNCHRONOUS UPDATES ON GRAPHS

2.1 Introduction

The main goal of this chapter is to construct a communication protocol under which an arbitrary eigenvector of the underlying graph operator can be found by the agents. In this protocol, agents communicate with each other in an asynchronous manner. More precisely, we consider the following collect-compute-broadcast scenario: states of the agents as a whole will be considered as a graph signal. At a random time instance, an agent wakes up independently and collects states of its neighbors. Then, the agent updates its own state as a linear combination of the received data, which is assumed to be described precisely by the graph operator. Then, the agent broadcasts the amount of change in its state to its neighbors. It is important to emphasize three points regarding this scenario: 1) The signal over the network is driven only by the initial conditions (and the graph operator). Agents do not take measurements, they only exchange data (their states) between each other. 2) This is an iterative scheme, and the graph signal converges to the desired eigenvector through repeated communications. 3) Unlike an edge-asynchronous protocol, in which only a connected pair of agents communicates with each other, the scenario we consider here is node-asynchronous where an agent wakes up randomly and communicates with all of its neighbors.

2.1.1 Assumptions on the Graph Operator

In this chapter we will not require the graph operator to be a symmetric matrix, that is, edges in the network are allowed to be directed, possibly with unequal edge weights. Self-loops are also allowed. However, *we do require the operator to be a normal matrix (equivalently, a unitary-diagonalizable matrix).* We note that symmetric (Hermitian) matrices are necessarily normal. Thus, results here are applicable to any graph with undirected edges. On the contrary, an arbitrary directed graph may not have a normal graph operator; it may even be non-diagonalizable (like the cases studied in [49]).

We assume that the graph operator (hence, the underlying network) is not time dependent. Only the node behaviors are time-varying (in a random fashion), but not their connectivity structure. We also assume that each node knows its neighbors.

2.1.2 Connections with Asynchronous Fixed Point Iterations

In this chapter we will study the described node-asynchronous network model from a fixed point iteration view-point, in which the notion of graph shift (discussed in Section 1.2) will be considered as the update step. In this approach, successively graph shifted signals can be examined as a sequence generated by a fixed point algorithm similar to the well-known power method. However, the traditional notion of graph shift is not directly applicable to the node-asynchronous model considered here since a graph shift requires all the nodes to communicate at the same time instance, which contradicts with the asynchronicity assumption. In order to tackle this problem, we will focus on an asynchronous variant, in which only a subset of indices are updated in each iteration.

We note that asynchronous fixed point iterations are well-studied problems [20]. The papers in [19, 14] presented important convergence results for general non-linear update models. In fact, the first analysis of the problem can be traced back to the study in [32] (and references therein) under the name "chaotic relaxation," in which a linear model with an input was investigated. In these studies the convergence is guaranteed mainly under two assumptions: 1) The update step is a contraction. 2) The indices are updated frequently enough. (See [14, 19, 32] for precise descriptions of these assumptions.)

In this chapter, we will consider linear updates without an input (power method on the graph operator) in a probabilistic setting in which a *random* subset of indices is updated in each iteration. In this setting, we prove that iterations converge to a fixed point of the graph operator in the mean-squared sense even in the case of the graph operator not being a contraction. More precise conditions will be spelled out later.

It is also important to note that the random asynchronous variant of the power method studied here can be thought of as a special case of coordinate descent algorithms [210]. In particular, coordinate-wise (or, asynchronous as we refer here) power iteration was studied recently in [106, 206]. Both techniques are demonstrated to perform well on data sets when computing the dominant eigenvectors, however they are not directly applicable to the autonomous network model we consider here. Furthermore, our results show that iterations do not necessarily converge to the dominant eigenvector, but they converge to an eigenvector of the unit eigenvalue (a fixed point) *even if there are other eigenvalues with magnitudes greater than unity.*

2.1.3 Connections with Gossip Algorithms and Consensus

The most common form of gossip protocols assumes that an adjacent pair of nodes share their current states with each other at random time instances and update their states by an averaging [52]. Due to their versatility, gossip-like algorithms have been studied extensively for distributed parameter estimation and optimization problems [52, 24, 91, 137, 167, 27, 96, 97, 98]. Further examples include push-sum [193, 102] and subgradient-push [131] (where the updates are synchronous, but the network is time-varying.) We refer the reader to [194, 130, 89, 22] for more general distributed/asynchronous optimization problems.

Although gossip protocols allow asynchronous communications, the canonical examples have edge-asynchronous (or, random link) behavior in which an edge gets activated randomly and a pair of nodes, linked by the edge, communicate with each other. See [88] for a treatment in random filtering in the context of graph signal processing. On the contrary, the model we consider here is node-asynchronous, that is, a node wakes up randomly and communicates with all of its neighbors. Thus, our analysis here deviates from the known results on gossip protocols.

More importantly, gossip protocols are designed in such a way that nodes reach a consensus. Depending on the problem formulation, the value of the consensus may be the estimated parameter(s), or the optimal solution of the objective function in general. For the consensus, graph operators, e.g., average connectivity matrix, the averaging matrix etc., have the property that the constant vector is the unique fixed point of the operator (assuming the graph is connected [52]). A signal over a network operating under a gossip protocol converges to the constant vector, which means that nodes reach a consensus. For example, the study in [211] searches for the optimal graph operator (for the fastest distributed averaging) under the constraint that the operator has constant vector as an eigenvector of eigenvalue 1. Differently in this study, we assume that the given graph operator has a fixed point, i.e., eigenvalue 1 exists, and show that randomized node-asynchronous updates converge to an eigenvector of eigenvalue 1 (a fixed point). *However, the eigenvector need not be a constant vector.* Thus, nodes may not reach a consensus. We will also show how to use polynomial filters in order to obtain convergence to an *arbitrary eigenvector* of the given graph operator. From this perspective, the problem we consider is more general than the consensus. In fact, the consensus can be considered as a particular instance of the problem studied here, in which the graph operator has the constant vector as an eigenvector.

2.1.4 Outline of the Chapter and Contributions

In Section 2.2 we first define the deterministic asynchronous update. This scheme is like the synchronous power iteration but values of only a subset of indices are updated. Afterwards, we consider the case where the update sets are selected at random. We impose a statistical model on the asynchronous updates, and derive, for later use, the expected value of the random update mechanism (Lemma 2.1). In Section 2.3 we consider cascades of random updates, and first analyze the expected signal in terms of the eigenvectors of the operator (Theorem 2.1). In order to prove the convergence of the updates, we consider the residual signal and bound its expected squared ℓ_2-norm (Theorem 2.2). Using this result we provide a sufficient condition (Corollary 2.2) and a necessary condition (Corollary 2.4) on the operator such that the signal is guaranteed to converge to an eigenvector of the unit eigenvalue through random updates in the mean-squared sense. We also show that asynchronous updates are better than the synchronous ones in terms of the convergence region of the eigenvalues (Corollary 2.3). In Section 2.4 we demonstrate how the eigenspace geometry of the operator plays a role in the convergence of the random asynchronous updates. In Section 2.5 we consider the problem from the graph signal processing point of view (Theorem 2.3). Since an arbitrary nonzero signal is proven to converge to the steady state, the signal in this state is said to be a "typical graph signal." We can interpret the typical signal as the smoothest signal on the graph (with respect to the graph operator). In Section 2.6 we consider polynomials of the graph operator in order to make the iteration converge to other eigenvectors (corresponding to non unit eigenvalues) (Theorem 2.4). By an explicit construction, we prove that second order polynomials are sufficient to achieve this purpose in the case of real eigenvalues (Theorem 2.5). Then, in Section 2.6.5, we formally present the node-asynchronous communication protocol (Algorithm 1) that implements a second order polynomial of the graph operator. In Section 2.7 we use the proposed algorithm to compute the Fiedler vector of a network in order to achieve an autonomous clustering via localized communication. In Section 2.8 we consider a reformulation of the power method and propose an algorithm that is proven to converge to the left and right dominant singular vectors of a normalized data matrix (Algorithm 2). Then, we extend the proposed algorithm in order to obtain an arbitrary rank approximation of an arbitrary data matrix (Algorithm 3), and we leverage the component-wise and asynchronous nature of the proposed algorithm in order to compute the singular vectors of distributed data with distributed computations (Algorithm 4).

The content of this chapter is mainly drawn from [181], and parts of it have been presented in [184, 179, 185, 168].

2.1.5 Preliminaries and Notation

In this chapter, we always assume that $\mathbf{A}$ is a normal matrix, i.e., $\mathbf{A}\,\mathbf{A}^{\mathrm{H}} = \mathbf{A}^{\mathrm{H}}\,\mathbf{A}$.

We will use $\mathcal{T}$ to denote a subset of $\{1, \cdots, N\}$, and its size is denoted as $t = |\mathcal{T}|$. We will use the notation $\sum_{\mathcal{T}}$ to denote the summation over all subsets of $\{1, \cdots, N\}$ of a fixed size t where the value of t should follow from the context. The index selection matrix of the set $\mathbf{D}_{\mathcal{T}}$ satisfies the following identities for a given size t:

$$\sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}} = \binom{N-1}{t-1}\,\mathbf{I}, \tag{2.1}$$

$$\frac{1}{\binom{N}{t}} \sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}}\,\mathbf{A}\,\mathbf{D}_{\mathcal{T}} = \frac{t\,(N-t)\ \mathrm{diag}(\mathbf{A}) + t\,(t-1)\,\mathbf{A}}{N(N-1)}, \tag{2.2}$$

which will be used in the subsequent proofs.

2.2 Asynchronous Power Iteration

Given a matrix of interest $\mathbf{A}$ and an initial signal $\mathbf{x}_0$ the conventional power iteration has the following form:

$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k\text{-}1}, \qquad \text{so that} \qquad \mathbf{x}_k = \mathbf{A}^k\,\mathbf{x}_0, \tag{2.3}$$

where the updates here are considered without normalizing the signal at each iteration. Normalization is avoided here intentionally to preserve the local nature of the updates as will be elaborated next.

In the context of graph signal processing, the matrix $\mathbf{A}$ is assumed to be a local graph operator (shift matrix) and the signal $\mathbf{Ax}$ is referred to as the shifted version of $\mathbf{x}$ on the graph (See Section 1.2). From this perspective $\mathbf{x}_k$ in (2.3) is the graph shifted version of $\mathbf{x}_{k\text{-}1}$. Since $\mathbf{A}$ is assumed to be a local operator a single shift can be implemented on the graph as a data exchange between the neighboring nodes. That is,

$$(\mathbf{x}_k)_i = \sum_{j\in\mathcal{N}_{\mathrm{in}}(i)} A_{i,j}\ (\mathbf{x}_{k\text{-}1})_j \qquad\qquad \forall\, i. \tag{2.4}$$

Notice that a norm of the signal depends on values of all of the nodes in the graph. Therefore, a norm cannot be known locally in the graph setting, which is why we have avoided normalization in (2.3).

Although a "graph shift" can be performed locally, the model in (2.3) forces all the nodes to send and receive data at the same time. Therefore, the graph shift does not have an autonomous implementation since it requires a centralized timing mechanism (synchronization) over the underlying graph.

In this study we will consider a variation of the power iteration, in which not all but a subset of indices, denoted by $\mathcal{T}$, are updated simultaneously and the remaining ones stay unchanged. More precisely, given an update set $\mathcal{T}$ we consider the following *asynchronous* (coordinate-wise) power iteration:

$$y_i = \begin{cases} (\mathbf{A}\mathbf{x})_i, & i \in \mathcal{T}, \\ x_i, & i \notin \mathcal{T}, \end{cases} \tag{2.5}$$

where $\mathbf{x}$ is the vector before update, and $\mathbf{y}$ is the vector after the update. In words, this update computes the multiplication $\mathbf{A}\mathbf{x}$, but it only updates the values of the elements indexed by the set $\mathcal{T}$, and keeps the remaining elements the same. In short, (2.3) is a "synchronous" update, whereas (2.5) is asynchronous. Both (2.3) and (2.5) are also referred to as *state recursions*, where the graph signal $\mathbf{x}$ is regarded as the state of a system. The model in (2.5) was also studied in [14, 19, 32] with slight differences. Furthermore, (2.5) is reminiscent of the Hopfield neural network [84], with the difference that there is no nonlinearity in (2.5). (Studies in [14, 19] have non-linearity.)

The asynchronous update defined in (2.5) can be written as a matrix-vector multiplication as follows:

$$\mathbf{y} = \sum_{i \notin \mathcal{T}} \mathbf{e}_i\, \mathbf{e}_i^{\mathrm{H}}\, \mathbf{x} + \sum_{i \in \mathcal{T}} \mathbf{e}_i\, \mathbf{e}_i^{\mathrm{H}}\, \mathbf{A}\, \mathbf{x} = \mathbf{Q}(\mathcal{T})\ \mathbf{x}, \tag{2.6}$$

where $\mathbf{Q}(\mathcal{T})$ is the matrix representing the asynchronous update on a set $\mathcal{T}$, and it can be written as follows:

$$\mathbf{Q}(\mathcal{T}) = \mathbf{I} + \mathbf{D}_{\mathcal{T}}\ (\mathbf{A} - \mathbf{I}). \tag{2.7}$$

In the next few sections, where we perform a convergence analysis, $\mathbf{A}$ can be treated as a generic matrix without considering specific relations to graphs. The relation to graph signals will be considered in Section 2.5. When the model in (2.5) is implemented on a graph (i.e., $\mathbf{A}$ is a graph operator), only the nodes in the update set $\mathcal{T}$ need to be synchronized. If the update set is selected as $\mathcal{T} = \{1, \cdots, N\}$, then $\mathbf{Q}(\mathcal{T}) = \mathbf{A}$. That is, the asynchronous update in (2.5) reduces to the classical synchronous update (graph shift) in (2.3). On the other extreme, if a single node is updated, $|\mathcal{T}|=1$, then *no synchronization is required at all* and the nodes are allowed to behave autonomously. We would like to note that the relation in (2.4) appears as if a node collects states of its neighbors, update its own state, and sits still. However, as we shall describe later in Section 2.6, we will consider the updates on a polynomial of the given graph operator, which will require the nodes to follow a collect-compute-broadcast scheme. These details will be elaborated in Section 2.6.5.

2.2.1 Randomized Asynchronous Updates

In this chapter we will study the behavior of a cascade of asynchronous updates where the update set $\mathcal{T}$ is assumed to be selected at random in each iteration. More precisely, we assume that the k^{th} iteration has the following form:

$$\mathbf{x}_k = \mathbf{Q}\, \mathbf{x}_{k\text{-}1}, \tag{2.8}$$

where $\mathbf{x}_k$ denotes the signal at the k^{th} iteration, and $\mathbf{Q}$ is a random matrix due to the fact that the underlying update set is selected at random.

It should be noted that $\mathbf{Q}$ and $\mathbf{Q}(\mathcal{T})$ are different from each other. The matrix $\mathbf{Q}(\mathcal{T})$ in (2.7) is a *deterministic* matrix. Given an update set $\mathcal{T}$, $\mathbf{Q}(\mathcal{T})$ represents the asynchronous update of (2.5). On the other hand, $\mathbf{Q}$ in (2.8) is a random variable *whose outcomes are in the form of* $\mathbf{Q}(\mathcal{T})$. More precisely, we consider the following probabilistic model:

$$\mathbb{P}[\,\mathbf{Q} = \mathbf{Q}(\mathcal{T})\,] = p_t \binom{N}{t}^{-1}, \quad \text{where} \quad t = |\mathcal{T}|, \tag{2.9}$$

where p_t denotes the probability of $\mathcal{T}$ having size t, that is,

$$p_t = \mathbb{P}\big[\,|\mathcal{T}| = t\,\big], \tag{2.10}$$

and $\sum_{t=1}^{N} p_t = 1$.

According to the model in (2.9), subsets of equal size are selected with equal probabilities. Therefore, the update scheme does not have any bias toward any node(s). To put differently, all the nodes are treated equally in the network. When $p_N = 1$, the model in (2.9) reduces to the regular power iteration in (2.3). When $p_1 = 1$, only one node is selected uniformly at random, which corresponds to the autonomous network model of interest.

The number of nodes to be updated, $\mathsf{T} = |\mathcal{T}|$, is a discrete random variable whose distribution will be shown to determine the behavior of the asynchronous updates.

We will see later in Section 2.3.2 that the following definition is very useful in our quantitative analysis:

$$\delta_{\mathsf{T}} = \frac{\mathbb{E}[\,\mathsf{T}\,(N-\mathsf{T})\,]}{\mathbb{E}[\,\mathsf{T}\,(N-1)\,]} = \frac{N-\mu_{\mathsf{T}}-\sigma_{T}^{2}/\mu_{\mathsf{T}}}{N-1}, \tag{2.11}$$

where μ_{T} and σ_T^2 denote the mean and the variance of the random quantity T, respectively. It can be verified that $0 \le \delta_{\mathsf{T}} \le 1$ with $\delta_{\mathsf{T}} = 0$ if and only if all the nodes are updated in each iteration (synchronous power iteration), and $\delta_{\mathsf{T}} = 1$ if and only if exactly one node is updated in each iteration. As a result of this, δ_{T} will be referred to as *the amount of asynchronicity* of iterations in the rest of the chapter.

We now prove:

Lemma 2.1. *Expectation of the random matrix* $\mathbf{Q}$ *in* (2.9) *is*

$$\mathbb{E}[\mathbf{Q}] = \frac{\mu_{\mathsf{T}}}{N}\,\mathbf{A} + \left(1-\frac{\mu_{\mathsf{T}}}{N}\right)\mathbf{I}. \tag{2.12}$$

Proof. The expectation of $\mathbf{Q}$ can be written as

$$\mathbb{E}[\mathbf{Q}] = \mathbb{E}\big[\,\mathbb{E}[\mathbf{Q}|\mathsf{T}]\,\big], \tag{2.13}$$

where the outer expectation is with respect to T (size of the sets), and the inner expectation is with respect to the content of the subsets of size T. Using (2.9) we have that

$$\mathbb{E}[\mathbf{Q}\,|\,\mathsf{T}] = \sum_{\mathcal{T}} \mathbb{P}[\,\mathbf{Q}=\mathbf{Q}(\mathcal{T})\,|\,\mathsf{T}\,]\;\;\mathbf{Q}(\mathcal{T}) = \sum_{\mathcal{T}} \frac{1}{\binom{N}{\mathsf{T}}}\Big(\mathbf{I}+\mathbf{D}_{\mathcal{T}}\,(\mathbf{A}-\mathbf{I})\Big), \tag{2.14}$$

where $\sum_{\mathcal{T}}$ denotes a summation over subsets of size T. Then,

$$\begin{aligned}
\mathbb{E}[\mathbf{Q}\,|\,\mathsf{T}] &= \frac{1}{\binom{N}{\mathsf{T}}}\sum_{\mathcal{T}}\mathbf{I} + \frac{1}{\binom{N}{\mathsf{T}}}\sum_{\mathcal{T}}\mathbf{D}_{\mathcal{T}}\,(\mathbf{A}-\mathbf{I}) = \mathbf{I} + \frac{1}{\binom{N}{\mathsf{T}}}\binom{N\text{-}1}{\mathsf{T}\text{-}1}\,\mathbf{I}\,(\mathbf{A}-\mathbf{I}), && (2.15)\\
&= \mathsf{T}/N\;\mathbf{A} + (1-\mathsf{T}/N)\;\mathbf{I}, && (2.16)
\end{aligned}$$

where (2.15) follows from (2.1). Due to (2.13), we have

$$\mathbb{E}[\mathbf{Q}] = \mathbb{E}[\;\mathsf{T}/N\;\mathbf{A} + (1-\mathsf{T}/N)\;\mathbf{I}\;] = \mu_{\mathsf{T}}/N\;\mathbf{A} + (1-\mu_{\mathsf{T}}/N)\;\mathbf{I}, \tag{2.17}$$

which gives the result in (2.12). □

Notice that $\mathbb{E}[\mathbf{Q}]$ is a convex combination of the operator $\mathbf{A}$ and the identity matrix. The quantity μ_{T}/N is the average fraction of the nodes that are updated simultaneously per iteration, and it appears as the weight of the operator $\mathbf{A}$ in $\mathbb{E}[\mathbf{Q}]$. The case of $\mu_{\mathsf{T}} = N$ results in $\mathbb{E}[\mathbf{Q}] = \mathbf{A}$, which corresponds to the case of synchronous power iteration.

2.3 Cascade of Asynchronous Updates

For the most practical scenarios we are interested in studying a sequence of random updates and the effect of the underlying matrix $\mathbf{A}$ on the convergence of the iterations. In the case of the synchronous (unnormalized) power iteration in (2.3), it is well known that an arbitrary nonzero initial signal $\mathbf{x}_0$ converges to a nonzero $\mathbf{x}$ if and only if $\mathbf{x}$ satisfies $\mathbf{Ax} = \mathbf{x}$ (i.e., 1 is an eigenvalue of $\mathbf{A}$), and the remaining eigenvalues of $\mathbf{A}$ satisfy $|\lambda| < 1$. If there is another eigenvalue satisfying $|\lambda| = 1$, then the signal $\mathbf{x}_k$ may fall into limit cycles, and if $|\lambda| > 1$, the signal grows in an unbounded manner through the iterations.

The random asynchronous update in (2.8) has very different convergence properties as we shall see. Iteratively using (2.8), the signal at the k^{th} iteration can be written as

$$\mathbf{x}_k = \mathbf{Q}_k \, \mathbf{Q}_{k\text{-}1} \cdots \, \mathbf{Q}_2 \, \mathbf{Q}_1 \, \mathbf{x}_0, \tag{2.18}$$

where $\mathbf{x}_0$ denotes the initial signal, and $\mathbf{Q}_i$'s are *independent* and *identically* distributed copies of the random matrix $\mathbf{Q}$ in (2.9). It should be noted that $\mathbf{x}_k$ is a random vector due to the fact that $\mathbf{Q}_i$'s are random variables.

2.3.1 Expected Amount of Projection onto the Eigenvectors

In order to characterize the behavior of $\mathbf{x}_k$, we first define the following quantity:

$$\widehat{x}_{k,\,j} = \mathbf{v}_j^{\mathrm{H}} \, \mathbf{x}_k, \tag{2.19}$$

which is the amount of projection on the j^{th} eigenvector (or, the j^{th} graph Fourier coefficient) of $\mathbf{x}_k$ at the k^{th} iteration. Due to the randomness of the updates, $\widehat{x}_{k,j}$ are random quantities as well. The following theorem gives the expected value of these coefficients:

Theorem 2.1. *Let* $\mathbf{v}_j$ *and* λ_j *be an eigenpair of* $\mathbf{A}$. *Then,*

$$\mathbb{E}[\widehat{x}_{k,\,j}] = \left(1 + \frac{\mu_{\mathrm{T}}}{N}\,(\lambda_j - 1)\right)^k \, \widehat{x}_{0,\,j}. \tag{2.20}$$

Proof. From (2.18) and (2.19) we have

$$\mathbb{E}[\widehat{x}_{k,\,j}] = \mathbf{v}_j^{\mathrm{H}} \, \mathbb{E}[\mathbf{Q}_k \, \mathbf{x}_{k\text{-}1}] = \mathbf{v}_j^{\mathrm{H}} \, \mathbb{E}[\mathbf{Q}] \, \mathbb{E}[\mathbf{x}_{k\text{-}1}], \tag{2.21}$$

where the last equality follows from the fact that $\mathbf{Q}_k$'s are independent and identically distributed random variables. Using the spectral decomposition of $\mathbf{A}$ in (1.6), $\mathbb{E}[\mathbf{Q}]$ in (2.12) can be written as

$$\mathbb{E}[\mathbf{Q}] = \mathbf{V}\left(\frac{\mu_{\mathrm{T}}}{N}\,\mathbf{\Lambda} + \left(1 - \frac{\mu_{\mathrm{T}}}{N}\right)\mathbf{I}\right)\mathbf{V}^{\mathrm{H}}. \tag{2.22}$$

Therefore, (2.21) results in the following:

$$\begin{aligned}\mathbb{E}[\widehat{x}_{k,j}] &= \mathbf{v}_j^{\mathrm{H}}\, \mathbf{V} \left(\frac{\mu_{\mathrm{T}}}{N}\mathbf{\Lambda} + \left(1 - \frac{\mu_{\mathrm{T}}}{N}\right)\mathbf{I}\right) \mathbf{V}^{\mathrm{H}}\, \mathbb{E}[\mathbf{x}_{k\text{-}1}], \\ &= \left(1 + \mu_{\mathrm{T}}/N\,(\lambda_j - 1)\right)\, \mathbb{E}[\widehat{x}_{k\text{-}1,j}]. \end{aligned} \tag{2.23}$$

Iterative use of (2.23) gives the result in (2.20). □

Theorem 2.1 shows that expected value of the graph Fourier coefficients depends not only on the corresponding eigenvalues but on the average number of nodes updated in each iteration as well. An immediate corollary is as follows:

Corollary 2.1. *Let* $\mathbf{v}_j$ *and* λ_j *be an eigenpair of* $\mathbf{A}$. *If*

$$\left|1 + \frac{\mu_{\mathrm{T}}}{N}\,(\lambda_j - 1)\right| < 1. \tag{2.24}$$

Then,

$$\lim_{k\to\infty} \mathbb{E}[\widehat{x}_{k,j}] = 0. \tag{2.25}$$

Proof. This follows from Theorem 2.1 and (2.19). □

In the case of synchronous updates $\mu_{\mathrm{T}} = N$, hence the relation in (2.20) reduces to $\mathbb{E}[\widehat{x}_{k,j}] = \widehat{x}_{k,j} = \lambda_j^k\, \widehat{x}_{0,j}$ as expected. Furthermore, (2.24) reduces to $|\lambda| < 1$, which is the well-known condition for the convergence of the power iteration of (2.3).

When the updates are asynchronous we have $\mu_{\mathrm{T}} < N$, and the region of convergence (2.24) in the complex eigenvalue plane is larger. In particular, one can readily verify the following:

$$\begin{matrix}|\lambda_j| \leq 1 \\ \lambda_j \neq 1\end{matrix} \quad \Longrightarrow \quad \left|1 + \frac{\mu_{\mathrm{T}}}{N}\,(\lambda_j - 1)\right| < 1, \tag{2.26}$$

which implies that $\mathbb{E}[\widehat{x}_{k,j}]$ converges to zero even if λ_j is *on* the unit circle, except when $\lambda_j = 1$. This is very much unlike to the synchronous case where the coefficients corresponding to unit magnitude eigenvalues do not die out through the iterations.

The unit eigenvalue $\lambda_j = 1$ deserves a specific attention since it has $\mathbb{E}[\widehat{x}_{k,j}] = \widehat{x}_{0,j}$ irrespective of the value of μ_{T}. If the initial signal $\mathbf{x}_0$ has a nonzero projection onto the eigenspace of the unit eigenvalue, then $\mathbb{E}[\mathbf{x}_k]$ always has a nonzero projection onto the eigenspace of the unit eigenvalue. Furthermore, the following lemma shows that a signal is invariant to the random asynchronous updates if and only if the signal lies in the eigenspace of the unit eigenvalue.

Lemma 2.2. *A signal* $\mathbf{x}$ *is invariant under all asynchronous updates if and only if it is invariant under the synchronous update. That is,*

$$\mathbf{A}\mathbf{x} = \mathbf{x} \qquad \Longleftrightarrow \qquad \mathbf{Q}(\mathcal{T})\,\mathbf{x} = \mathbf{x} \quad \forall \mathcal{T}. \tag{2.27}$$

Proof. ($\Leftarrow$) Assume that $\mathbf{Q}(\mathcal{T})\,\mathbf{x} - \mathbf{x} = \mathbf{0}$ for all $\mathcal{T}$. Since it is true for *any* subset $\mathcal{T}$, it should hold true for all subsets of a fixed size t as well. Then,

$$\mathbf{0} = \sum_{\mathcal{T}} \mathbf{Q}(\mathcal{T})\,\mathbf{x} - \mathbf{x} = \sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}}\,(\mathbf{A} - \mathbf{I})\,\mathbf{x} = \binom{N-1}{t-1}\,(\mathbf{A} - \mathbf{I})\,\mathbf{x}, \tag{2.28}$$

which proves that $\mathbf{A}\mathbf{x} = \mathbf{x}$.

The converse ($\Rightarrow$) simply follows from (2.7). □

Lemma 2.2 shows that if the random signal $\mathbf{x}_k$ ever reaches a steady-state point $\mathbf{x}$ through iterations, then $\mathbf{x}$ should be in the eigenspace of the unit eigenvalue. The result of Corollary 2.1 supports this claim as well. This is not a surprising result as the eigenspace of eigenvalue 1 consists of non-zero fixed points of the operator. However, neither Lemma 2.2 nor Corollary 2.1 says anything about the convergence of the random asynchronous iteration as k increases. In the following section, we will prove that $\mathbf{x}_k$ indeed converges to an eigenvector of the unit eigenvalue (a fixed point) as k goes to infinity.

2.3.2 Convergence in Mean-Squared Sense

In the following we will assume that $\mathbf{A}$ has a unit eigenvalue with multiplicity $M \geq 1$. This assumption ensures that the asynchronous update equation has a fixed point (Lemma 2.2). Without loss of generality we will order the eigenvalues of $\mathbf{A}$ such that $\lambda_j \neq 1$ for $1 \leq j \leq N\text{-}M$. Notice that non-unit eigenvalues are allowed to be complex in general, and complex eigenvalues on or outside the unit circle are not ruled out. Then, the eigenvalue decomposition of $\mathbf{A}$ can be written as

$$\mathbf{A} = [\mathbf{U} \;\; \mathbf{V}_1]\,\mathrm{diag}\big([\lambda_1 \; \cdots \; \lambda_{N\text{-}M} \; 1 \; \cdots \; 1]\big)[\mathbf{U} \;\; \mathbf{V}_1]^{\mathrm{H}}, \tag{2.29}$$

where $\mathbf{V}_1 \in \mathbb{C}^{N\times M}$ is an orthonormal basis for the eigenspace of the unit eigenvalue, and $\mathbf{U} \in \mathbb{C}^{N\times(N\text{-}M)}$ corresponds to the eigenvectors of the non-unit eigenvalues. Since $\mathbf{A}$ is assumed to be a normal matrix, we have $\mathbf{U}^{\mathrm{H}}\,\mathbf{V}_1 = \mathbf{0}$, and $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$.

We now define the following quantities:

$$\rho = \lambda_{\max}\big(\mathbf{U}^{\mathrm{H}}\;\mathrm{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\big), \tag{2.30}$$

$$\bar{\rho} = \lambda_{\min}\big(\mathbf{U}^{\mathrm{H}}\;\mathrm{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\big), \tag{2.31}$$

which will play a crucial role in the analysis of convergence. Notice that ρ and $\bar{\rho}$ do not depend on the particular selection of the basis matrix $\mathbf{U}$. Just the column space of $\mathbf{U}$ determines their values. More importantly, we have the following property:

Lemma 2.3. *The following holds true for any* $\mathbf{U} \in \mathbb{C}^{N\times(N\text{-}M)}$ *with* $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$*:*

$$\frac{1}{N}\mathbf{I} \;\preceq\; \mathbf{U}^{\mathrm{H}}\ \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)\mathbf{U} \;\preceq\; \mathbf{I}. \tag{2.32}$$

Proof. We note that $\mathbf{U} \in \mathbb{C}^{N\times(N\text{-}m)}$ has orthonormal columns, i.e., $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$ and prove the upper bound in (2.32) first. Note that $\mathbf{U}\,\mathbf{U}^{\mathrm{H}} \preceq \mathbf{I}$. Then we can write the following:

$$(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})_{i,i} = \mathbf{e}_i^{\mathrm{H}}\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{e}_i \leq 1 \implies \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right) \preceq \mathbf{I} \implies \mathbf{U}^{\mathrm{H}}\ \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)\mathbf{U} \preceq \mathbf{I}, \tag{2.33}$$

where $\mathbf{e}_i$ denotes the i^{th} column of the identity matrix of dimension N.

We now prove the lower bound in (2.32). Let $\mathbf{u}_{(i)}$ denote the i^{th} row of $\mathbf{U}$, then it is clear that $(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})_{i,j} = \mathbf{u}_{(i)}\,\mathbf{u}_{(i)}^{\mathrm{H}}$. Let $\mathbf{x} \in \mathbb{C}^N$ be an arbitrary vector. Then,

$$\begin{aligned}
\mathbf{x}^{\mathrm{H}}\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x} &= \left|\mathbf{x}^{\mathrm{H}}\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x}\right| = \left|\sum_{i=1}^{N}\sum_{j=1}^{N} x_i^*\,(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})_{i,j}\,x_j\right| = \left|\sum_{i=1}^{N}\sum_{j=1}^{N} x_i^*\,\mathbf{u}_{(i)}\,\mathbf{u}_{(j)}^{\mathrm{H}}\,x_j\right| \\
&\leq \sum_{i=1}^{N}\sum_{j=1}^{N} |x_i|\,\left|\mathbf{u}_{(i)}\,\mathbf{u}_{(j)}^{\mathrm{H}}\right|\,|x_j| \leq \sum_{i=1}^{N}\sum_{j=1}^{N} |x_i|\;\|\mathbf{u}_{(i)}\|_2\;\|\mathbf{u}_{(j)}\|_2\;|x_j| \\
&= \left(\sum_{i=1}^{N} |x_i|\;\|\mathbf{u}_{(i)}\|_2\right)^2 \leq N\sum_{i=1}^{N} |x_i|^2\;\|\mathbf{u}_{(i)}\|_2^2 = N\,\mathbf{x}^{\mathrm{H}}\ \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)\mathbf{x}.
\end{aligned} \tag{2.34}$$

Then, the inequity (2.34) implies that

$$\mathbf{U}\,\mathbf{U}^{\mathrm{H}} \preceq N\ \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right) \qquad\implies\qquad \mathbf{U}^{\mathrm{H}}\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{U} \;\preceq\; N\,\mathbf{U}^{\mathrm{H}}\ \mathrm{diag}\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)\mathbf{U}, \tag{2.35}$$

which proves the lower bound due to the fact that $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$. □

So, Lemma 2.3 implies the following inequality regarding the quantities $\bar{\rho}$ and ρ:

$$\frac{1}{N} \leq \bar{\rho} \leq \rho \leq 1. \tag{2.36}$$

For an arbitrary $\mathbf{x}_k$, let $\mathbf{r}_k$ denote the residual from the projection of $\mathbf{x}_k$ onto the column space of $\mathbf{V}_1$. That is,

$$\mathbf{r}_k = \mathbf{x}_k - \mathbf{V}_1\,\mathbf{V}_1^{\mathrm{H}}\,\mathbf{x}_k = \mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_k. \tag{2.37}$$

Then, the convergence of $\mathbf{x}_k$ to an eigenvector of the unit eigenvalue is equivalent to the convergence of $\mathbf{r}_k$ to zero. The following theorem, whose proof is presented in Appendix 2.10.2, provides bounds for $\mathbf{r}_k$ as follows:

Theorem 2.2. *The expected squared ℓ_2-norm of the residual at the k^{th} iteration is bounded as follows:*

$$\psi^k \, \|\mathbf{r}_0\|_2^2 \;\leq\; \mathbb{E}\big[\, \|\mathbf{r}_k\|_2^2 \,\big] \;\leq\; \Psi^k \, \|\mathbf{r}_0\|_2^2, \tag{2.38}$$

where

$$\Psi = \max_{1 \leq j \leq N\text{-}M} \; c\,(\lambda_j), \qquad\qquad \psi = \min_{1 \leq j \leq N\text{-}M} \; \bar{c}\,(\lambda_j), \tag{2.39}$$

$$\begin{aligned} c\,(\lambda) &= 1 + \frac{\mu_{\mathsf{T}}}{N}\left(|\lambda|^2 - 1 + \delta_{\mathsf{T}}\,(\rho - 1)\,|\lambda - 1|^2\right), \\ \bar{c}\,(\lambda) &= 1 + \frac{\mu_{\mathsf{T}}}{N}\left(|\lambda|^2 - 1 + \delta_{\mathsf{T}}\,(\bar{\rho} - 1)\,|\lambda - 1|^2\right). \end{aligned} \tag{2.40}$$

The importance of Theorem 2.2 is twofold: First, it reveals the effect of the eigenvalues (λ_j), the eigenspace geometry ($\rho, \bar{\rho}$), and the amount of asynchronicity of the updates (δ_{T}) on the rate of convergence. In the synchronous case $\delta_{\mathsf{T}} = 0$ and $\mu_{\mathsf{T}} = N$, hence we have $\Psi = \max_{1 \leq j \leq N\text{-}M} |\lambda_j|^2$. This result is consistent with the well-known fact that the rate of convergence of the power iteration is determined by the second largest eigenvalue. However, in the asynchronous case ($\delta_{\mathsf{T}} > 0$), not just the eigenvalues but the eigenspace geometry of $\mathbf{A}$ has an effect. As a result, similar matrices may have different convergence rates due to their different eigenspaces. This point will be elaborated in Section 2.4. Furthermore, in order to guarantee that $\mathbb{E}[\|\mathbf{r}_k\|_2^2] \leq \varepsilon \, \|\mathbf{r}_0\|_2^2$ for a given error threshold ε, inequalities in (2.38) show that it is necessary to have at least $\lfloor \log(\varepsilon)/\log(\psi) \rfloor$ iterations, and sufficient to have $\lceil \log(\varepsilon)/\log(\Psi) \rceil$ iterations.

Secondly, Theorem 2.2 reveals a region for the eigenvalues such that the residual error through asynchronous updates is guaranteed to convergence to zero in the mean-squared sense. The following corollary presents this result formally.

Corollary 2.2. *Assume that all non-unit eigenvalues of* $\mathbf{A}$ *satisfy the following condition:*

$$\left|\lambda - \frac{\alpha}{\alpha + 1}\right| < \frac{1}{\alpha + 1}, \tag{2.41}$$

where

$$\alpha = \delta_{\mathrm{T}}\ (\rho - 1). \tag{2.42}$$

Then,

$$\lim_{k\to\infty}\quad \mathbb{E}\big[\ \|\mathbf{r}_k\|_2^2\ \big] = 0. \tag{2.43}$$

Proof. From (2.39) it is clear that $\Psi < 1$ if and only if

$$|\lambda|^2 - 1 + \alpha\,|\lambda - 1|^2 < 0, \tag{2.44}$$

for all non-unit eigenvalues λ. The inequality in (2.44) can be equivalently written as in (2.41). Since it implies that $\Psi < 1$, Theorem 2.2 guarantees the convergence of $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ to zero as the number of updates, k, goes to infinity. □

An important remark is as follows: Corollary 2.2 provides a condition under which $\mathbf{r}_k$ is guaranteed to converge to a point (zero) as k goes to infinity. On the other hand, $\mathbf{x}_k$ itself only converges to a random variable defined over the eigenspace of the unit eigenvalue. This is illustrated in Figure 2.1 where the eigenspace of the unit eigenvalue is spanned by the vector $[1\ \ 1]^{\mathrm{H}}$, and $\mathbf{x}_0 = [\text{-}1\ \ 1]^{\mathrm{H}}$. In the synchronous case the signal converges to a point through a deterministic trajectory as shown in Figure 2.1a. For the random asynchronous case, Figure 2.1b illustrates the trajectories of the signals for different realizations. Convergence of $\mathbf{r}_k$ to zero implies that the limit of $\mathbf{x}_k$ always lie in the eigenspace of the unit eigenvalue (with a random orientation). Since any point in the eigenspace is an eigenvector, we can safely say that $\mathbf{x}_k$ converges to an eigenvector of the unit eigenvalue.

Notice that the convergence region for the eigenvalues defined in (2.41) is parametrized by α, and it is a disk on the complex plane with radius $1/(\alpha+1)$ centered at $\alpha/(\alpha+1)$. This region is visualized in Figure 2.2. Notice that $0 \leq \delta_{\mathrm{T}} \leq 1$ and $0 < \rho \leq 1$ always hold true. As a result α satisfies $-1 < \alpha \leq 0$. The key observation is that the region in (2.41) grows as α approaches -1, and it is the smallest (and corresponds to the unit disk) when $\alpha = 0$. The quantity β and the large circle in Figure 2.2 will be explained after Corollary 2.4.

Corollary 2.2 reveals the combined effect of the eigenspace geometry of $\mathbf{A}$ (quantified with ρ) and the amount of asynchronicity (quantified with δ_{T}) on the convergence of the iterations. In the case of $\delta_{\mathrm{T}} = 0$ the region reduces to the unit disk, which is the well-known condition on the eigenvalues for the synchronous updates to converge. This is an expected result since the case of $\delta_{\mathrm{T}} = 0$ corresponds to the synchronous

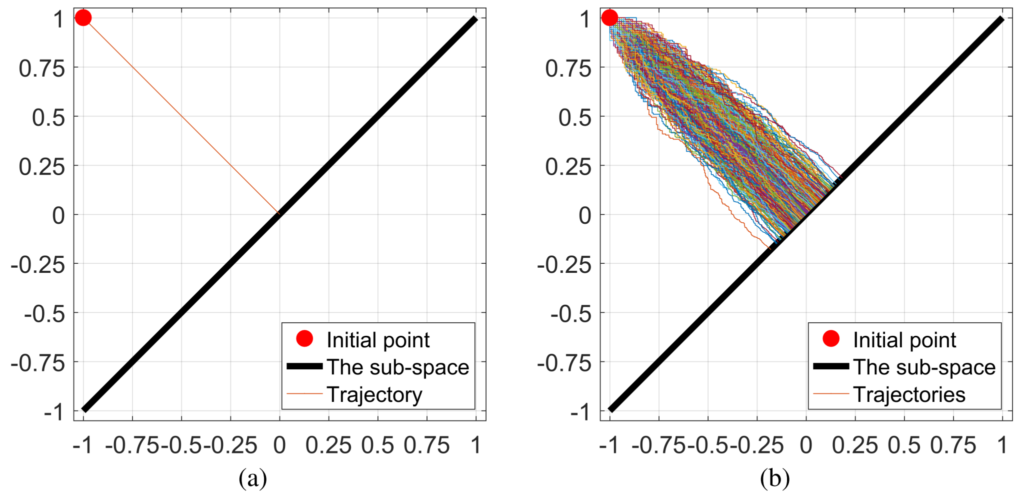


Figure 2.1: Some realizations of the trajectories of the signal through updates for (a) the non-random synchronous case, (a) the random asynchronous case.

update itself. More importantly, the synchronous updates imply $\alpha = 0$ independent of the eigenspace geometry of $\mathbf{A}$. Therefore, the convergence is determined entirely by the eigenvalues of $\mathbf{A}$ in the synchronous case.

On the other hand, the case of asynchronous updates results in a *larger* convergence region for the eigenvalues. First of all, it should be noted that asynchronous updates increase the convergence region if the eigenspace geometry of $\mathbf{A}$ permits. If $\rho = 1$ then $\alpha = 0$, and the region of convergence is not improved by asynchronous iterations. However, if $\rho < 1$ (which is the case in most practical applications), then it is possible to enlarge the region of convergence using asynchronous iterations. As δ_{T} gets larger (less number of nodes are updated concurrently), α gets smaller, hence the convergence region gets larger. Even if one index is left unchanged in *some* iterations, we have $\delta_{\mathsf{T}} > 0$, and the residual $\mathbf{r}_k$ can converge to zero, *even when non-unit eigenvalues outside the unit circle might exist.* This is a remarkable property of the asynchronous updates since the residual (hence the signal itself) would blow up in the case of synchronous updates. Notice that in the extreme case of $\delta_{\mathsf{T}} = 1$, the region of convergence is the largest possible. That is to say, *updating exactly one node in each iteration maximizes the region of convergence of the eigenvalues.* On the other extreme, the synchronous update is the most restrictive case, which is formally stated in the following corollary:

Corollary 2.3. *If the synchronous updates on* $\mathbf{A}$ *converge, then*

$$\lim_{k\to\infty} \quad \mathbb{E}\big[\, \|\mathbf{r}_k\|_2^2 \,\big] = 0, \tag{2.45}$$

for random updates on $\mathbf{A}$ *with any amount of asynchronicity.*

Proof. If the synchronous updates converge, then all non-unit eigenvalues of $\mathbf{A}$ satisfy $|\lambda| < 1$. Hence, they also satisfy (2.41) for any value of α. Therefore, Corollary 2.2 ensures the convergence of the updates irrespective of the value of δ_{T}. □

It should be clear that converse of Corollary 2.3 is not true. Thus consider a scenario in which a signal over a network of nodes with autonomous (asynchronous) behavior stays in the steady-state. If the nodes start to operate synchronously, then it is possible for the signal to blow up. This happens if some of the eigenvalues fall outside of the reduced convergence region due to the reduction in the amount of asynchronicity. In fact, the study in [142] claims that large-scale synchronization of neurons is an underlying mechanism of epileptic seizures. Similarly, the study in [196] presents the relation between increased neural synchrony and epilepsy as well as Parkinson's disease. It should be noted that neural networks follow nonlinear models whereas the model we consider here is linear. Thus, results presented here do not apply to brain networks. Nevertheless, these neurobiological observations are consistent with the implications of Corollary 2.2 and Corollary 2.3 from a conceptual point of view.

Apart from the convergence of the iterations, Theorem 2.2 is also useful to characterize the case of non-converging iterations. In this regard, the following corollary presents a region for the eigenvalues such that asynchronous updates are guaranteed not to converge.

Corollary 2.4. *Assume that all non-unit eigenvalues of* $\mathbf{A}$ *satisfy the following:*

$$\left|\lambda - \frac{\beta}{\beta+1}\right| \geq \frac{1}{\beta+1}, \tag{2.46}$$

where

$$\beta = \delta_{\mathrm{T}}\ (\bar{\rho} - 1). \tag{2.47}$$

Then,

$$\mathbb{E}[\ \|\mathbf{r}_k\|_2^2\ ] \geq \|\mathbf{r}_0\|_2^2. \tag{2.48}$$

Furthermore, if (2.46) *is satisfied with strict inequality, then* $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ *grows unboundedly as* k *goes to infinity.*

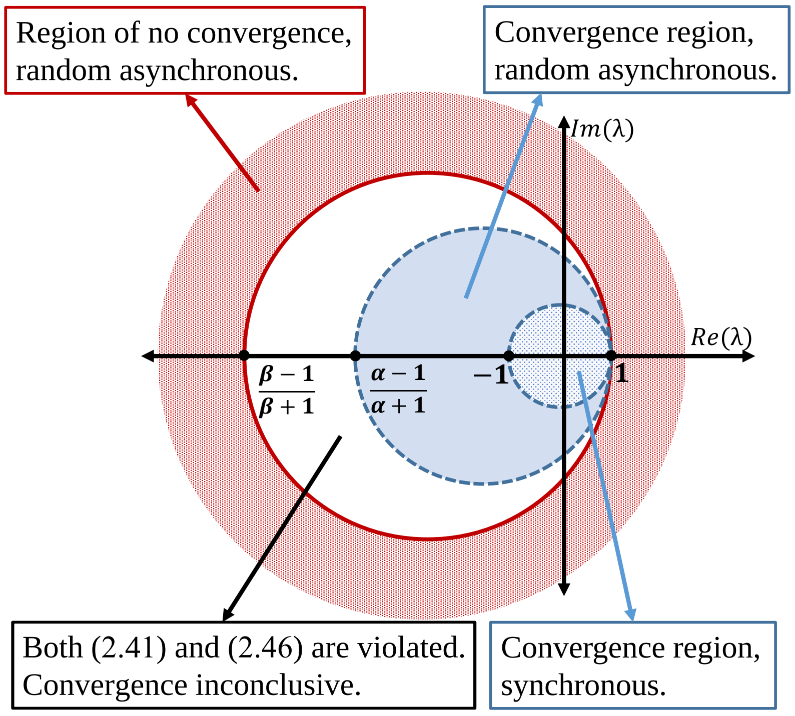

Region of no convergence, random asynchronous.
Convergence region, random asynchronous.
$Im(\lambda)$
$Re(\lambda)$
$\frac{\beta-1}{\beta+1}$
$\frac{\alpha-1}{\alpha+1}$
-1
1
Both (2.41) and (2.46) are violated. Convergence inconclusive.
Convergence region, synchronous.

Figure 2.2: Regions (given in (2.41) and (2.46)) for the eigenvalues such that random asynchronous updates are guaranteed to converge and diverge, respectively.

Proof. From (2.39) it is clear that $\psi \geq 1$ if and only if

$$|\lambda|^2 - 1 + \beta\,|\lambda - 1|^2 \geq 0, \tag{2.49}$$

for all non-unit eigenvalues λ. The inequality in (2.49) can be equivalently written as in (2.46). Since (2.46) implies that $\psi \geq 1$, Theorem 2.2 indicates that $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ is lower bounded by $\|\mathbf{r}_0\|_2^2$. If (2.46) is satisfied strictly, then $\psi > 1$. As a result, $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ grows unboundedly as k goes to infinity. □

From the definitions in (2.42) and (2.47) note that $\alpha \geq \beta$ is always true due to the fact that $\rho \geq \bar{\rho}$. Therefore, the conditions in (2.41) and (2.46) describe disjoint regions on the complex plane. See Figure 2.2. Corollary 2.4 also shows that the condition $|\lambda - \beta/(\beta+1)| < 1/(\beta+1)$ is *necessary* for the iterations to converge, whereas the condition in (2.41) is *sufficient* for the convergence (both in the mean square sense). If there exists an eigenvalue that violates both (2.41) and (2.46), then convergence is inconclusive. This region is also indicated in Figure 2.2.

At this point it is important to compare the implications of Corollary 2.2 with the classical result presented in [14, 32]. Under the mild assumption that all the indices are selected sufficiently often (see [32] for precise definition), the study [32] showed that the linear asynchronous model in (2.5) converges for any index sequence if and only if the spectral radius of $|\mathbf{A}|$ is strictly less than unity, where $|\mathbf{A}|$ denotes a matrix with element-wise absolute values of $\mathbf{A}$. On the other hand, our Corollary 2.2 allows eigenvalues with magnitudes grater than unity. Although these two results appear to be contradictory (when $\mathbf{A}$ consists of non-negative elements), the key difference is the notion of convergence. As an example, consider the matrix $\mathbf{A}_2$ defined in (2.55). Its spectral radius is exactly 1, and [32] proved that *there exists* a sequence of indices under which iterations on $\mathbf{A}_2$ do not converge. For example, assuming N is odd, consider the index sequence generated as $i = (2k-1) \pmod{N} + 1$. However, Corollary 2.2 proves the convergence in a statistical *mean-square averaged* sense. (See Figure 2.5.) In short, when compared with [32], Corollary 2.2 requires a weaker condition on $\mathbf{A}$ and guarantees a convergence in a weaker (and probabilistic) sense.

2.3.3 Rate of Convergence

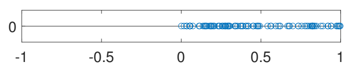
Although the region of convergence can only be expanded by random component-wise updates as explained in the previous subsection, the rate of convergence has a more intricate behavior that requires a detailed discussion. In the regular power method, the rate of convergence is determined by the eigenvalue gap, which is the difference between the *magnitudes* of the two largest eigenvalues of the matrix $\mathbf{A}$. Thus, the sign (or, the phase in the complex case) of the eigenvalues are not important. Unlike the regular power method, sign of the eigenvalues do matter in the random component-wise updates. As a result, the eigenvalue spectrum has an asymmetric impact on the rate of convergence.

In order to explain the difference between the random component-wise and regular power methods in terms of the rate of convergence, we will first present a numerical example that summarizes our key observations. Then, we will explain the effect of the asynchronicity on the rate of convergence theoretically..

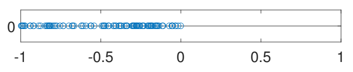
Numerical Observations

In this simulation, random component-wise updates select exactly one index per iteration ($\delta_{\mathrm{T}} = 1$), in which case a single inner product is computed per iteration. On the contrary, the regular power method computes N inner products per iteration. For a fair comparison between the two, we fix *the total number of inner product*, which will be denoted by K. Thus, the regular power method will run $\lceil K/N \rceil$ iterations, whereas the component-wise variant will run K iterations.

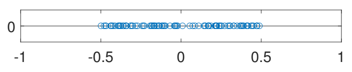
For the numerical experiment we consider three symmetric matrices of size $N = 100$. All three matrices are constructed such that $\lambda_N = 1$ is an eigenvalue with multiplicity $M = 1$, and the remaining N-1 eigenvalues are selected to be $|\lambda_i| < 1$ so that the power method (hence any random variant) is guaranteed to converge to an eigenvector of the eigenvalue $\lambda_N = 1$. (See Corollary 2.2.) In the first two examples we consider a pair of simultaneously diagonalizable matrices. The non-unit eigenvalues of the first matrix are selected to be positive (visualized in Figure 2.3a), and the non-unit eigenvalues of the second matrix are selected to be the negative of those of the first matrix (visualized in Figure 2.3b). In the third example we take a random symmetric matrix with non-unit eigenvalues satisfying $-0.5 < \lambda_i < 0.5$ (visualized in Figure 2.3c). Figures 2.3d, 2.3e and 2.3f show the value of $\mathbb{E}[\|\mathbf{r}_K\|_2^2]/\|\mathbf{r}_0\|_2^2$ as a function of K for the three matrices described above.



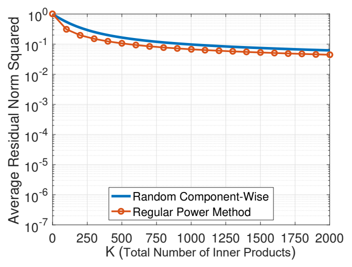
(a) Eigenvalue gap: 0.0026



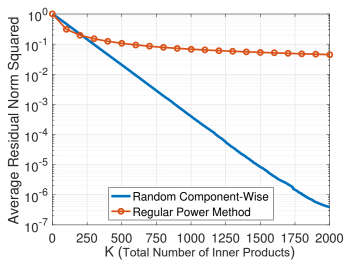
(b) Eigenvalue gap: 0.0026



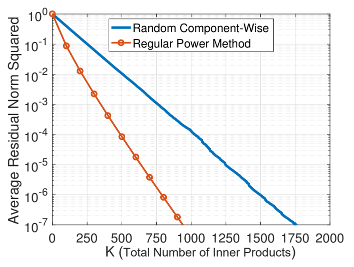
(c) Eigenvalue gap: 0.5011



(d)



(e)



(f)

Figure 2.3: Non-unit eigenvalues of the (a) first, (b) second, and (c) third examples. Eigenvalue gap is defined as the difference between 1 and the magnitude of the largest non-unit eigenvalue. Normalized residual errors in the (d) first, (e) second, and (f) third examples. Since the regular power method requires N inner products per iteration, the residual error appears only at integer multiples of $N = 100$. Results are obtained by averaging over 10^4 independent runs.

We first compare the results in Figures 2.3d and 2.3e. Since the eigenvalues have the same magnitudes, the regular power method behaves the same in both cases. Although the eigenvalue gap is the same in both cases, the random component-wise method converges *significantly faster* when the second dominant eigenvalue is negative. When the second dominant eigenvalue is positive, both the regular and the component-wise updates behave similarly. In the third example, Figure 2.3f, the matrix has a large eigenvalue gap, in which case the random component-wise updates do not converge as fast as the regular power method.

Theoretical Justification

In order to explain the behavior in Figure 2.3, in this section we will assume a slightly simplified stochastic model for the selection of the update sets in (2.5). Namely, we will assume that the scheme (2.5) updates *exactly* μ_{T} indices per iteration. Thus, the random variable T (which denotes the size of the update sets) becomes a deterministic quantity, and $\sigma_{\mathsf{T}}^2 = 0$. So, the parameter δ_{T} (the amount of asynchronicity) reduces to the following form:

$$\delta_{\mathsf{T}} = \frac{N - \mu_{\mathsf{T}}}{N - 1}. \tag{2.50}$$

In this setting we note that an update in the form of (2.5) requires μ_{T} inner products per iteration. So, the cost of a single power iteration, which requires N inner products, is equivalent to the cost of N/μ_{T} asynchronous iterations in which μ_{T} indices are updates simultaneously. Since the associated cost of an eigenvalue defined in (2.40) disregards the cost of an iteration, we consider the following quantity instead:

$$r\big(\lambda;\ \mu_{\mathsf{T}}, \rho\big) = \left(1 + \frac{\mu_{\mathsf{T}}}{N}\Big(|\lambda|^2 - 1 + \delta_{\mathsf{T}}\,(\rho - 1)\,|\lambda - 1|^2\Big)\right)^{N/\mu_{\mathsf{T}}}, \tag{2.51}$$

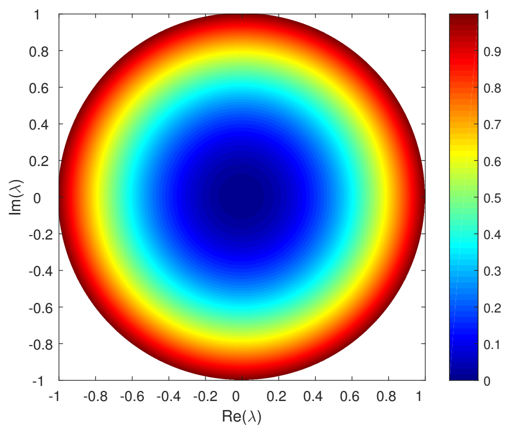
which results in a fair comparison among the component-wise updates with different amount of asynchronicity. The quantity $r\big(\lambda;\ \mu_{\mathsf{T}}, \rho\big)$ can be interpreted as the amount of reduction in the residual error when eigenvalue λ is present in the matrix $\mathbf{A}$ with the eigenspace parameter ρ, and the model (2.5) updates μ_{T} indices simultaneously. Thus, smaller values of $r\big(\lambda;\ \mu_{\mathsf{T}}, \rho\big)$ indicate a better (faster) convergence of the randomized scheme in (2.5).

We first note that the quantity $r\big(\lambda;\ \mu_{\mathsf{T}},\ \rho\big)$ can be equivalently re-written as follows:

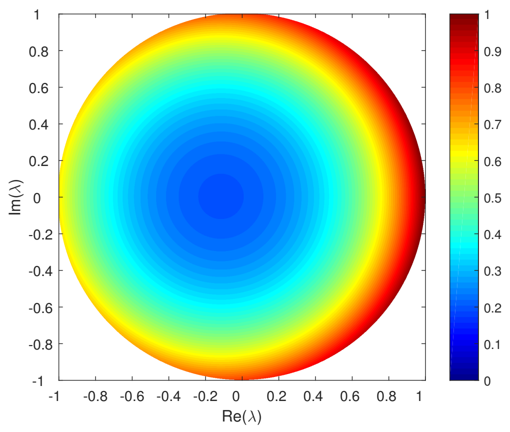
$$r\big(\lambda;\ \mu_{\mathsf{T}}, \rho\big) = \left(1 + \frac{\mu_{\mathsf{T}}}{N}\ (\alpha + 1)\ \left(\left|\lambda - \frac{\alpha}{\alpha + 1}\right|^2 - \frac{1}{(\alpha + 1)^2}\right)\right)^{N/\mu_{\mathsf{T}}}, \tag{2.52}$$

where $\alpha = \delta_{\mathsf{T}}\ (\rho - 1)$ as in Corollary 2.2. Then, it is clear that the point $\lambda^{\star} = \alpha/(\alpha + 1)$ minimizes $r\big(\lambda;\ \mu_{\mathsf{T}},\ \rho\big)$ over the variable λ, and $r\big(\lambda;\ \mu_{\mathsf{T}},\ \rho\big)$ (as a function of λ) is circularly symmetric with respect to the point $\lambda^{\star}$. In addition, the inequality

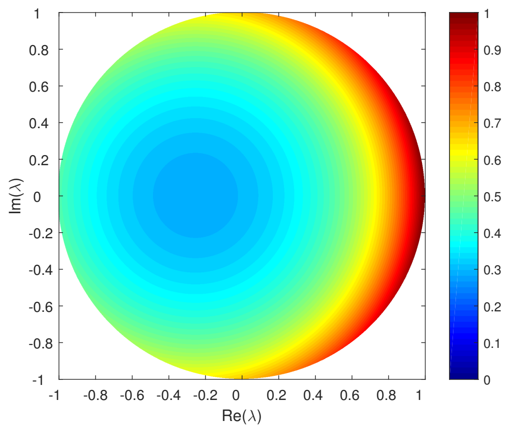
(2.36) ensures that $r\big(\lambda;\ \mu_\mathsf{T}, \rho\big) \geq 0$. In order to demonstrate its behavior, we evaluate $r(\lambda;\ \mu_\mathsf{T},\ \rho)$ numerically over the unit disk (as a function of λ) for different values of μ_T and ρ. These computations are visualized in Figure 2.4.



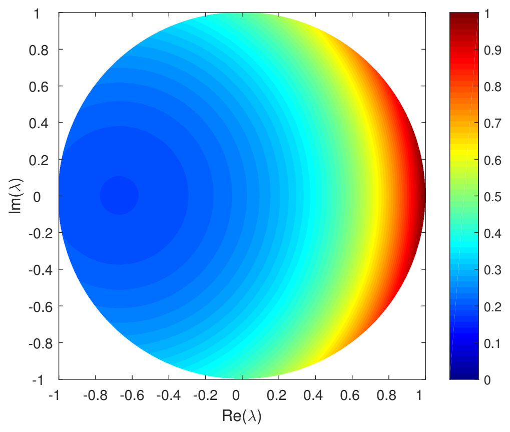
(a) $r(\lambda;\ N,\ 0.8)$



(b) $r(\lambda;\ N/2,\ 0.8)$



(c) $r(\lambda;\ 1,\ 0.8)$



(d) $r(\lambda;\ 1,\ 0.6)$

Figure 2.4: Numerical evaluation of $r\big(\lambda, \mu_\mathsf{T}\big)$ for various different values of μ_T and ρ. The value of N is set to be $N = 100$.

In the case of synchronous updates we have $\mu_\mathsf{T} = N$, thus $\alpha = 0$, and the quantity defined in (2.51) reduces to $r\big(\lambda;\ N,\ \rho\big) = |\lambda|^2$ irrespective of the value of ρ, which can be seen clearly from Figure 2.4a. Thus, as $|\lambda|$ approaches 1, the value of $r\big(\lambda;\ N, \rho\big)$ approaches 1 irrespective of the phase of λ. So, only the magnitude of an eigenvalue affects the convergence rate of the regular power iteration, which is a well-known result.

In the case of asynchronous updates we have $\mu_\mathsf{T} < N$, and we will assume $\rho < 1$ (which is the case in most practical applications). Thus, we have $\alpha < 0$, and the phase of an eigenvalue becomes important since $r\big(\lambda;\ \mu_\mathsf{T},\ \rho\big)$ is no longer a circularly symmetric function of λ with respect to the origin. Figures 2.4b, 2.4c and 2.4d visualize this behavior clearly. In particular, note that as λ approaches 1, the quantity $r(\lambda;\ \mu_{\mathrm{T}},\ \rho)$ approaches 1 as well. On the other hand, as λ approaches -1, the quantity $r(\lambda;\ \mu_{\mathrm{T}},\ \rho)$ stays bounded away from 1. More precisely,

$$r(1;\ \mu_{\mathrm{T}},\rho) = 1, \qquad r(-1;\ \mu_{\mathrm{T}},\rho) = \left(1 + \frac{\mu_{\mathrm{T}}}{N}\,4\,\alpha\right)^{N/\mu_{\mathrm{T}}}. \tag{2.53}$$

So, eigenvalues that are close to 1 result in a slower convergence, whereas eigenvalues can be arbitrarily close to -1, yet the convergence does not necessarily slow down. Therefore, in light of (2.53) and Figure 2.4 we can conclude that *the random component-wise updates favor negative eigenvalues over positive ones.* This conclusion is consistent with the numerical observations made in Figures 2.3d and 2.3e: when the second dominant eigenvalue is close to 1, both the random component-wise updates and the regular power iteration converge slowly. On the contrary, when the second dominant eigenvalue is close to -1, the random component-wise updates converge faster than the synchronous (regular) counter-part. In fact, it is possible to construct a matrix $\mathbf{A}$ (by placing the second dominant eigenvalue sufficiently close to -1) such that the randomized updates converge *arbitrarily faster* than the regular power iteration.

Although random component-wise updates converge faster when the second dominant eigenvalue is close to -1, Figure 2.3f shows that randomized updates are not always faster than the synchronous counter-part. In order to explain the behavior observed in Figure 2.3f, we consider $r(\lambda;\ \mu_{\mathrm{T}},\rho)$ evaluated at $\lambda = 0$. More precisely,

$$r(0;\ \mu_{\mathrm{T}},\rho) = \left(1 + \frac{\mu_{\mathrm{T}}}{N}\Big(\delta_{\mathrm{T}}\,(\rho - 1) - 1\Big)\right)^{N/\mu_{\mathrm{T}}} \geq \left(1 - \frac{\mu_{\mathrm{T}}}{N}\right)^{2N/\mu_{\mathrm{T}}}, \tag{2.54}$$

where the lower bound follows from (2.36). As long as the updates are randomized (the case of $\mu_{\mathrm{T}} < N$), it is clear from (2.54) that $r(0;\ \mu_{\mathrm{T}},\rho)$ is bounded away from zero. Figures 2.4b, 2.4c and 2.4d visualize this behavior as well. Then, we can conclude that *in the case of random component-wise updates the associated cost of an eigenvalue is bounded away from zero even when the eigenvalue itself is close to zero.* This conclusion is consistent with the simulation results presented in Figure 2.3f. When the non-unit eigenvalues are close to zero, regular power iteration converges faster than its randomized variant.

As a concluding remark, we note that the results presented in this section are valid when $\mathbf{A}$ is a normal matrix, i.e, $\mathbf{A}$ is unitarily diagonalizable. The results of this section may not hold true when $\mathbf{A}$ is an arbitrary matrix. Nevertheless, the normality condition is not a loss of generality when dealing with undirected graphs as in Section 2.7, or if our goal is to construct a random component-wise method that can compute the singular vectors of an arbitrary matrix as in Section 2.8.

2.4 The Importance of the Eigenspace Geometry

Corollary 2.2 presented in the previous section showed that the signal converges to an eigenvector of the unit eigenvalue via random asynchronous updates even in the case of $\mathbf{A}$ having other eigenvalues with magnitudes larger than one. Thus, asynchronous updates can enlarge the convergence region of the eigenvalues if the eigenspace geometry of $\mathbf{A}$ permits. In this section we will consider a simple example to demonstrate the effect of the eigenspace geometry on the convergence of the asynchronous updates. For this purpose we will consider the following matrices of size N:

$$\mathbf{A}_1 = \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_N \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix}, \tag{2.55}$$

where the diagonal entries of $\mathbf{A}_1$ are $A_n = e^{j2\pi n/N}$, and $\mathbf{A}_2$ is the cyclic permutation matrix. Note that $\mathbf{A}_1$ and $\mathbf{A}_2$ are related by a similarity, so they have identical eigenvalues. In particular they have eigenvalue 1 with multiplicity $M = 1$. However, they have different eigenspace geometries that affect the behavior of asynchronous iterations as demonstrated next.

Notice that an update with $\mathbf{A}_1$ corresponds to element-wise multiplication with complex exponentials that does not change the magnitude of the entries. As a result, iterations with $\mathbf{A}_1$ do not converge or blow up, whether the updates are synchronous or asynchronous. Indeed in this case $\rho = \bar{\rho} = 1$ and $\beta = 0$, which lead to the same conclusion by Corollary 2.4.

Next, $\mathbf{A}_2$ is circulant matrix, and the normalized DFT matrix diagonalizes it. In this case $\rho = \bar{\rho} = 1 - 1/N$. In the synchronous case $\delta_{\mathrm{T}} = 0$, hence $\beta = 0$, and Corollary 2.4 shows that the residual error is bounded below. In fact, the power iteration on $\mathbf{A}_2$ neither converges nor blows up since it corresponds to the cyclic shift of the vector, and the initial signal repeats itself after every N iterations. In the asynchronous case, $\mathbf{A}_2$ has $\alpha = -\delta_{\mathrm{T}}/N$, hence any nonzero amount of asynchronicity implies $\alpha < 0$, in which case the convergence region in (2.41) contains the unit closed disk. Thus, the initial signal converges to an eigenvector of the unit eigenvalue. As an example consider the case of updating exactly one index in each iteration. This is equivalent to randomly selecting an index and assigning its value to the next one. As random updates are applied repeatedly the signal will have more and more duplicate elements until all the elements are the same. The final vector is $\eta\,\mathbf{1}$ where η is a random variable. Thus, the initial signal converges to a constant vector, which is the eigenvector of $\mathbf{A}_2$ with the unit eigenvalue. (See Figure 2.1b.) Theorem 2.2 verifies this convergence in the mean-squared sense with the following bounds on the rate:

$$\Psi = \max_{1\le n\le N\text{-}1} 1 - \frac{1}{N}\frac{1}{N}\left|e^{j2\pi n/N} - 1\right|^2 \approx 1 - 4\pi^2/N^4, \tag{2.56}$$

and

$$\psi = \min_{1\le n\le N\text{-}1} 1 - \frac{1}{N}\frac{1}{N}\left|e^{j2\pi n/N} - 1\right|^2 \approx 1 - 4/N^2. \tag{2.57}$$

In order to compare the bounds in (2.56) and (2.57) with the actual rate of convergence, asynchronous iterations with $\delta_{\mathrm{T}} = 1$ are simulated on $\mathbf{A}_2$ of size $N = 32$. Figure 2.5 visualizes the expected squared ℓ_2-norm of the residual as a function of the iteration index as well as the bounds given by Theorem 2.2. The result is obtained by averaging over 10^7 independent runs.

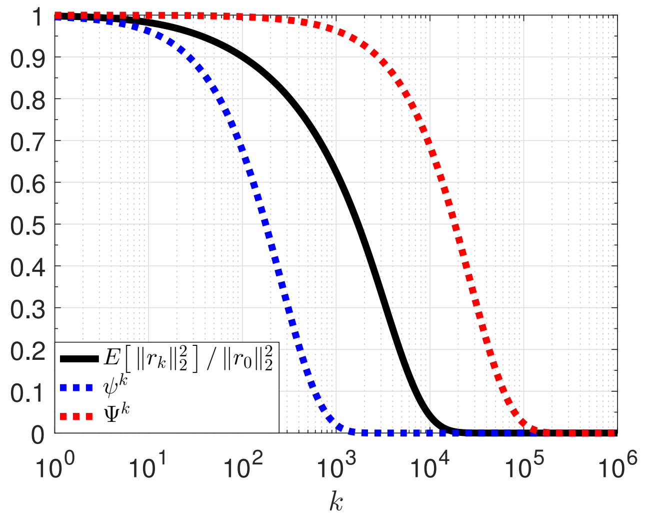


Figure 2.5: Simulated convergence of random asynchronous updates on the cyclic shift matrix of size $N = 32$ together with the bounds provided by Theorem 2.2.

Simulations show that the bounds suggested by Theorem 2.2 are off by an order of magnitude in this specific example. Since the random updates converge faster than the upper bound given by Theorem 2.2, i.e., (2.56), tighter bounds on the rate of convergence may be possible to obtain.

2.5 Interpretation in the Context of Graph Signal Processing

In Sections 2.2 and 2.3 we defined and studied the random asynchronous updates from a linear algebraic point of view where the results (Theorems 2.1, 2.2, and their corollaries) are general enough to apply them to an arbitrary normal matrix. When $\mathbf{A}$ is assumed to be a graph operator, the asynchronous iterations offer an insightful way to interpret a "typical" graph signal. Consider a network of fully autonomous nodes (agents) and assume that a node updates its value via retrieving information from its neighbors. This update scheme can be modeled with (2.5) where the update set $\mathcal{T}$ has only one element ($\delta_\mathrm{T} = 1$). In this model, the local graph operator $\mathbf{A}$ models the way nodes update their values. If a node computes the sum of its neighbors, then $\mathbf{A}$ is the adjacency matrix; if a node computes the sum of differences with its neighbors, then $\mathbf{A}$ is the graph Laplacian; if a node computes a weighted average of its neighbors, then $\mathbf{A}$ is a weighted adjacency matrix. Therefore, the matrix $\mathbf{A}$ describes what the nodes compute, the update scheme in (2.5) describes the time dynamics of the nodes, and the values held by the nodes are considered as a signal on the graph (with respect to the operator $\mathbf{A}$). In the following, we will interpret random asynchronous updates in this context and argue that a "typical signal" on a graph is necessarily a "smooth signal" with respect to the graph operator. For this purpose first notice that the GFT of $\mathbf{Ax} - \mathbf{x}$ is given by $(\mathbf{\Lambda} - \mathbf{I})\,\widehat{\mathbf{x}}$. Then, we define a notion of smoothness accordingly as follows:

Definition 2.1 (Smoothness Set)**.** *A graph signal* $\mathbf{x}$ *belongs to the set* $\mathcal{S}_\epsilon$ *if its graph Fourier transform* $\widehat{\mathbf{x}} = \mathbf{V}^\mathrm{H}\,\mathbf{x}$ *satisfies*

$$|\widehat{x}_i|\;|\lambda_i - 1| \;\leq\; \epsilon \qquad\qquad \forall\, i, \tag{2.58}$$

for the given graph operator $\mathbf{A}$.

A signal $\mathbf{x}$ belongs to $\mathcal{S}_\epsilon$ if the differences between the graph Fourier coefficients of $\mathbf{x}$ and $\mathbf{Ax}$ are not larger than ϵ in absolute sense. That is, small values of ϵ implies that $\mathbf{x}$ and $\mathbf{Ax}$ are similar to each other. Hence, we can interpret ϵ as a scale of the smoothness of the signal $\mathbf{x}$ with respect to the operator $\mathbf{A}$. Here, the smoothness is quantified according to the total variation (TV) of the eigenvectors, $\mathrm{TV}(\mathbf{v}_i) = |\lambda_i - 1|$, as introduced in [153]. For a given value of ϵ, the set $\mathcal{S}_\epsilon$ describes the signals with $|\widehat{x}_i| \leq \epsilon \,/\, |\lambda_i - 1|$. So, for a smooth signal, the Fourier coefficient $\widehat{x}_i$

should be smaller for those λ_i whose $\mathbf{v}_i$ has large total variation. The condition in (2.58) is equivalent to bounding a weighted max-norm of the GFT of $\mathbf{x}$, that is,

$$\mathbf{x} \in \mathcal{S}_\epsilon \quad \Longleftrightarrow \quad \| \, (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}} \, \|_\infty \leq \epsilon, \tag{2.59}$$

where the weight matrix is selected as $|\mathbf{\Lambda} - \mathbf{I}|$. The set $\mathcal{S}_\epsilon$ depends on the underlying graph operator. *A signal that is smooth on one graph may not be smooth on another graph.*

In the following we will consider the effect of a single asynchronous update on the smoothness of the signal. For this purpose let $\mathbf{x}$ be the initial signal and $\mathbf{y}$ be the signal after an update. According to (2.7) they are related as $\mathbf{y} = \mathbf{Q}(\mathcal{T}) \, \mathbf{x}$, which can be equivalently written in the graph Fourier domain as follows:

$$\widehat{\mathbf{y}} = \mathbf{V}^{\rm H} \, \mathbf{Q}(\mathcal{T}) \, \mathbf{V} \, \widehat{\mathbf{x}} = \widehat{\mathbf{x}} + \sum_{i \in \mathcal{T}} \mathbf{V}^{\rm H} \mathbf{e}_i \, \mathbf{e}_i^{\rm H} \, \mathbf{V} \, (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}}. \tag{2.60}$$

The following theorem reveals a relation between smooth graph signals and a single asynchronous update of (2.5).

Theorem 2.3. *Assume that the signal* $\mathbf{x}$ *belongs to* $\mathcal{S}_\epsilon$ *of a graph with operator* $\mathbf{A}$. *Then, the signal* $\mathbf{y}$ *computed as in* (2.5) *satisfies the following:*

$$\|\widehat{\mathbf{y}} - \widehat{\mathbf{x}}\|_\infty \quad \leq \quad \epsilon \; |\mathcal{T}| \; \|\mathbf{V}\|_{\max} \; \|\mathbf{V}\|_\infty. \tag{2.61}$$

Proof. Assume that $\mathbf{x} \in \mathcal{S}_\epsilon$. Then, we can write the following set of inequalities for a fixed index j:

$$\begin{aligned}
|\widehat{y}_j - \widehat{x}_j| &= \left| \mathbf{e}_j^{\rm H} \sum_{i \in \mathcal{T}} \mathbf{V}^{\rm H} \mathbf{e}_i \, \mathbf{e}_i^{\rm H} \, \mathbf{V} \, (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}} \right|, && (2.62)\\
&\leq \sum_{i \in \mathcal{T}} \left| \mathbf{e}_j^{\rm H} \mathbf{V}^{\rm H} \mathbf{e}_i \right| \; \left| \mathbf{e}_i^{\rm H} \, \mathbf{V} \, (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}} \right|, && (2.63)\\
&\leq \|\mathbf{V}\|_{\max} \sum_{i \in \mathcal{T}} \left| \mathbf{e}_i^{\rm H} \, \mathbf{V} \, (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}} \right|, && (2.64)\\
&\leq \|\mathbf{V}\|_{\max} \sum_{i \in \mathcal{T}} \left\| \mathbf{V}^{\rm H} \, \mathbf{e}_i \right\|_1 \; \left\| (\mathbf{\Lambda} - \mathbf{I}) \, \widehat{\mathbf{x}} \right\|_\infty, && (2.65)\\
&\leq \|\mathbf{V}\|_{\max} \; |\mathcal{T}| \; \|\mathbf{V}\|_\infty \; \epsilon, && (2.66)
\end{aligned}$$

where (2.62) follows from (2.60), (2.63) follows from the triangle inequality, (2.64) follows from the definition of $\|\mathbf{V}\|_{\max}$, (2.65) follows from the Hölder inequality, and (2.66) follows from the fact that $\|\mathbf{V}\|_\infty$ is the largest ℓ_1-norm of the rows of $\mathbf{V}$. Then, we have the following:

$$\|\widehat{\mathbf{y}} - \widehat{\mathbf{x}}\|_\infty = \max_j |\widehat{y}_j - \widehat{x}_j| \leq \epsilon \ |\mathcal{T}| \ \|\mathbf{V}\|_{\max} \ \|\mathbf{V}\|_\infty, \tag{2.67}$$

where the inequality follows from the fact that the bound in (2.66) is valid for any index j. □

If the underlying graph is circulant, then $\mathbf{V}$ is the normalized DFT matrix, and Theorem 2.3 reduces to the following simple form as a corollary: $\|\widehat{\mathbf{y}} - \widehat{\mathbf{x}}\|_\infty \leq \epsilon \ |\mathcal{T}|$ [184].

The bound given by Theorem 2.3 is not tight in general. Nevertheless, it provides a useful interpretation: if the initial signal is smooth on the graph (belongs to $\mathcal{S}_\epsilon$ with ϵ being small), then the amount of change in each GFT coefficient is also small, hence a smooth signal remains to be (relatively) smooth on the graph after a single asynchronous update. If the initial signal is not smooth (ϵ is large), then the right-hand-side of (2.61) is a large quantity, and thus we cannot reach a conclusion regarding the effect of an asynchronous update.

We will say that *a signal* $\mathbf{x}$ *is typical to the graph (with respect to the operator* $\mathbf{A}$*) if it satisfies* $\mathbf{Ax} = \mathbf{x}$. This definition is motivated by the following three observations. First of all Lemma 2.2 shows that a typical signal is invariant (stationary) under any asynchronous update. Secondly, a signal is typical if and only if it is the smoothest signal (w.r.t. Definition 2.1):

$$\mathbf{x} \in \mathcal{S}_0 \quad \Longleftrightarrow \quad \mathbf{A}\ \mathbf{x} = \mathbf{x}. \tag{2.68}$$

This is consistent with the studies in [152, 160, 113] that consider typical signals to be smooth on the graph. Thirdly, and most importantly, Corollary 2.2 proves that when the nodes communicate autonomously for a sufficiently long time, the signal becomes typical to the graph irrespective of the starting point. Once the signal gets typical it stays the same over the network. Therefore, a steady signal over an autonomous network is necessary typical with respect to the operator $\mathbf{A}$.

The equivalence given in (2.68) also shows the smoothing effect of the asynchronous updates. An arbitrary initial signal is not expected to be smooth over a graph, but Corollary 2.2 proves that the signal over an autonomous network eventually becomes typical, hence the smoothest. However, it should be noted that the smoothing (convergence) does not happen monotonically unlike the power iteration of (2.3).

Some updates might be adversarial that increase the variation of the signal over the graph. See Figure 1 of [184] for an illustrative example. Nevertheless, some other updates cancel them out in the long run as proven by Theorem 2.2.

2.6 Asynchronous Polynomial Filters on Graphs

Lemma 2.2 of Section 2.3 showed that there exists a signal invariant under an asynchronous update if and only if $\mathbf{A}$ has an eigenvalue 1. In this case, according to Corollary 2.2, an arbitrary nonzero initial signal converges to an eigenvector with eigenvalue 1. These results show that asynchronous updates on $\mathbf{A}$ can only converge to the eigenspace of $\mathbf{A}$ with the unit eigenvalue. In this section we will challenge this limitation: what if we want asynchronous updates to converge to an eigenspace of $\mathbf{A}$ with eigenvalue other than 1? In order to approach this problem we will start with an L^{th} order polynomial of the given operator, that is,

$$h(\mathbf{A}) = \sum_{n=0}^{L} h_n \, \mathbf{A}^n, \tag{2.69}$$

for some set of coefficients h_n's, and consider asynchronous updates on $h(\mathbf{A})$ that are defined as follows:

$$(\,\mathbf{x}_{k+1}\,)_i = \begin{cases} \big(\, h(\mathbf{A}) \ \mathbf{x}_k \,\big)_i, & i \in \mathcal{T}, \\ (\,\mathbf{x}_k\,)_i, & i \notin \mathcal{T}. \end{cases} \tag{2.70}$$

Polynomials of a graph operator are useful to consider because of the following two reasons. Firstly, they are localized. Computation of $(h(\mathbf{A})\,\mathbf{x})_i$ requires the i^{th} node to retrieve information only from its L-hop neighbors. If the polynomial is of low order (L has a small value), then $h(\mathbf{A})\,\mathbf{x}$ can be computed locally, which is crucial to the autonomous model we consider in this study. Secondly, $\mathbf{A}$ and $h(\mathbf{A})$ have the same eigenvectors, that is,

$$h(\mathbf{A}) = \mathbf{V} \ h(\mathbf{\Lambda}) \ \mathbf{V}^{\mathrm{H}}. \tag{2.71}$$

Therefore, a carefully constructed polynomial can manipulate the eigenvalues of $\mathbf{A}$ in such a way that asynchronous iterations on $h(\mathbf{A})$ can be guaranteed to converge to a desired eigenspace of $\mathbf{A}$ even though iterations on $\mathbf{A}$ itself fail to do so. This idea is formally presented in the following theorem.

Theorem 2.4. *Let λ_i denote the eigenvalues of a given graph operator* $\mathbf{A}$. *For a specific target eigenvalue λ_j, assume that a polynomial $h(\cdot)$ satisfies the following conditions:*

$$h(\lambda_j) = 1 \qquad \textit{and} \qquad |h(\lambda_i)| < 1 \quad \forall\, \lambda_i \neq \lambda_j. \tag{2.72}$$

Then, random updates on $h(\mathbf{A})$ *as in* (2.70) *converge to an eigenvector of* $\mathbf{A}$ *with eigenvalue* λ_j *for any amount of asynchronicity* $0 \leq \delta_{\mathrm{T}} \leq 1$.

Proof. Since $h(\cdot)$ is assumed to satisfy (2.72), $h(\mathbf{A})$ has a unit eigenvalue, and the non-unit eigenvalues are strictly less than one in magnitude. Hence, Corollary 2.2 ensures that the updates converge to a point in the eigenspace of $h(\mathbf{A})$ with the eigenvalue 1, which is equivalent to the eigenspace of $\mathbf{A}$ with the eigenvalue λ_j due to the property in (2.72). □

Polynomial filters play an important role in the area of graph signal processing. Starting from the early works [153, 151, 160], polynomial filters are designed to achieve a desired frequency response on the graph. On the contrary, the condition (2.72) disregards the overall response of the filter since its objective is to isolate a single eigenvector. Thus, the design procedure and the properties of the polynomials (to be presented in the subsequent sections) differ from the polynomial approximation ideas considered in [153, 151, 160].

Theorem 2.4 tells that an *arbitrary* eigenvector of the graph can be computed in a decentralized manner if a low order polynomial satisfying (2.72) is constructed. In this regard the updates on polynomial filters resemble the beam steering in antenna arrays [202]: assume that the order of the polynomial, L, is fixed. Then, the communication pattern between the nodes is completely determined by the operator $\mathbf{A}$ and the order L (See Algorithm 1). Once the nodes start to update their values randomly and asynchronously, the behavior of the signal is controlled by the polynomial coefficients. By changing the coefficients in the light of (2.72), one can *steer* the signal over the graph to desired "directions," which happen to be the eigenvectors of the operator. Here, L corresponds to the number of elements (sensors) in the array, and the filter coefficients serve the purpose of steering in both cases.

Note that the condition in (2.72) is not necessary in general for the convergence of random updates with a fixed amount of asynchronicity. (See Section 2.4.) However, (2.72) is necessary to guarantee the convergence for all levels of asynchronicity including the most restrictive case of power iteration.

Notice that if the operator $\mathbf{A}$ is the graph Laplacian, and the target eigenvalue is $\lambda_j = 0$, then the corresponding eigenvector is the constant vector. In this case the problem reduces to the consensus problem, and the condition in (2.72) becomes a relaxed version of the polynomial considered in [150].

In the following sections we will assume that eigenvalues of $\mathbf{A}$ are real valued, which is the case for undirected graphs, and assume that $h_n \in \mathbb{R}$. Although the complex case can also be considered in this framework, as we shall see, some of the results do not extend to the complex case.

2.6.1 The Optimal Polynomial

In this section we consider the construction of the polynomial that has the largest gap between the unit eigenvalue and the rest. In order to represent the condition (2.72) in the matrix-vector form we will use a vector of length $L + 1$ to denote the polynomial in (2.69), that is, $\mathbf{h} = [h_0 \ \cdots \ h_L]^{\mathrm{T}}$. In addition, let $\mathbf{\Phi}$ be a Vandermonde matrix constructed with the eigenvalues of $\mathbf{A}$ in the following form:

$$\mathbf{\Phi} = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^L \\ 1 & \lambda_2 & \lambda_2^2 & \cdots & \lambda_2^L \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \lambda_N & \lambda_N^2 & \cdots & \lambda_N^L \end{bmatrix} \in \mathbb{R}^{N\times(L+1)}. \tag{2.73}$$

In the case of repeated eigenvalues, we assume that the repeated rows of the matrix $\mathbf{\Phi}$ are removed. Let $\phi_{(j)}$ denote the row of $\mathbf{\Phi}$ corresponding to the target eigenvalue λ_j, and let $\bar{\mathbf{\Phi}}_j$ denote the remaining rows of $\mathbf{\Phi}$.

In order to find the optimal L^{th} order polynomial satisfying (2.72), we consider the following optimization problem:

$$\max_{c,\ \mathbf{h}} \quad c \qquad \text{s.t.} \qquad \begin{aligned} \phi_{(j)}\ \mathbf{h} &= 1, \\ \left|\bar{\mathbf{\Phi}}_j\ \mathbf{h}\right| &\leq (1-c)\ \mathbf{1}. \end{aligned} \tag{2.74}$$

First of all notice that the constraints in (2.74) are linear due to the fact that $\mathbf{\Phi}$ and $\mathbf{h}$ are real valued. The objective function is linear as well. Hence, (2.74) is a linear programming that can be solved efficiently given the eigenvalue matrix $\mathbf{\Phi}$.

The constraints of (2.74) enforce the polynomial to satisfy the desired condition in (2.72) while the objective function maximizes the distance between the unit and non-unit eigenvalues of $h(\mathbf{A})$. Therefore, the formulation in (2.74) searches for the polynomial that yields the fastest rate of convergence on $h(\mathbf{A})$ among all polynomials of order L satisfying (2.72). Hence, we will refer to the solution of (2.74) as *the optimal polynomial*.

It should be noted that the solution of (2.74) is optimal with respect to the worst case scenario of the synchronous updates. In general, the polynomial generated via (2.74) may not give the fastest rate of convergence for an arbitrary amount of asynchronicity.

2.6.2 Sufficiency of Second Order Polynomials

In order to make use of the construction in (2.74), the order of the polynomial, L, should be selected appropriately so that the problem is feasible and admits a solution. One way to guarantee the feasibility is to select $L = N - 1$, in which case a solution always exists due to the invertibility of the Vandermonde matrix $\mathbf{\Phi}$ (by disregarding the multiplicity of eigenvalues). In this case, however, updates are no longer localized, which prevents the asynchronous model of (2.5) from being useful. At the other extreme, the case of $L = 1$ is insufficient to ensure the condition (2.72) in general. Therefore, nonlocal updates are required for the sake of flexibility in the eigenvectors. Nevertheless, the locality of the updates needs only to be compromised marginally, as the following theorem shows that $L = 2$ is in fact *sufficient* to satisfy (2.72).

Theorem 2.5. *Assume that the operator* $\mathbf{A}$ *has real eigenvalues* $\lambda_i \in \mathbb{R}$. *For a given target eigenvalue* λ_j, *the condition in* (2.72) *is satisfied by the following second order polynomial:*

$$h(\lambda) = 1 - 2\,\epsilon\,(\lambda - \lambda_j)^2/s_j^2, \tag{2.75}$$

for any ϵ *in* $0 < \epsilon < 1$ *and* s_j *satisfying the following:*

$$s_j \geq \max_{1 \leq i \leq N} |\lambda_i - \lambda_j|. \tag{2.76}$$

Proof. It is clear that $h(\lambda_j) = 1$. In the following we will show that $-1 < h(\lambda_i) < 1$ for all $\lambda_i \neq \lambda_j$. For the upper bound note that $(\lambda_i - \lambda_j)^2 > 0$ for all $\lambda_i \neq \lambda_j$. Therefore,

$$1 - h(\lambda_i) = 2\,\epsilon\,(\lambda_i - \lambda_j)^2/s_j^2 > 0, \tag{2.77}$$

which proves that $h(\lambda_i) < 1$ for all $\lambda_i \neq \lambda_j$. For the lower bound notice that we have $s_j^2 \geq (\lambda_i - \lambda_j)^2$ for all λ_i by the condition in (2.76). Therefore we have

$$h(\lambda_i) = 1 - 2\,\epsilon\,(\lambda_i - \lambda_j)^2/s_j^2 \geq 1 - 2\epsilon > -1, \tag{2.78}$$

for all λ_i. □

Notice that ϵ in (2.75) is a free parameter which can be tuned to increase the gap between the eigenvalues. Thus, the polynomial given in (2.75) is *not* guaranteed to be optimal in general. It merely shows that a second order polynomial satisfying (2.72) always exists, which also implies the feasibility of (2.74) in the case of $L = 2$, or larger.

An important remark is as follows: the sufficiency of second order polynomials does not extend to the complex case in general. To see this consider the following set of N complex numbers: $\lambda_n = e^{j2\pi n/(N\text{-}1)}$ for $1 \leq n \leq N\text{-}1$ and $\lambda_N = 0$. As shown in the supplementary document, *no* polynomial of order $L \leq N\text{-}2$ (possibly with complex coefficients) can satisfy $|h(\lambda_n)| < 1$ for $1 \leq n \leq N\text{-}1$ and $h(\lambda_N) = 1$. This adversarial example shows not only that second order polynomials are insufficient, but also that a polynomial of order N-1 is in fact necessary in the complex case in general. Although no guarantee can be provided, low order polynomials might still exist in the complex case depending on the values of the eigenvalues of a given operator $\mathbf{A}$.

2.6.3 Spectrum-Blind Construction of Suboptimal Polynomials

Although the solution of (2.74) provides the optimal polynomial, it requires the knowledge of all the eigenvalues of $\mathbf{A}$. Such information is not available and difficult to obtain in general. By compromising the optimality, we will discuss a way of constructing second order polynomials satisfying (2.72) *without* using the knowledge of all eigenvalues of $\mathbf{A}$, except the target eigenvalue λ_j.

First of all notice that a value for the coefficient s_j used in (2.75) can be found using only the minimum and the maximum eigenvalues of the operator. That is, the following selection

$$s_j = \max\{\lambda_{\max} - \lambda_j, \ \ \lambda_j - \lambda_{\min}\}, \tag{2.79}$$

satisfies (2.76). Therefore, the minimum, the maximum and the target eigenvalues suffice to construct the polynomial (2.75).

In fact (2.76) can be satisfied by using an appropriate upper bound for $\lambda_{\max}$ and lower bound for $\lambda_{\min}$. For example if $\mathbf{A}$ is the adjacency matrix or Laplacian, we can use the largest degree $d_{\max}$ of the graph to select s_j as follows:

The Laplacian

In this case the eigenvalues are bounded as $0 \leq \lambda_i \leq 2\, d_{\max}$. Hence,

$$s_j = d_{\max} + |\lambda_j - d_{\max}|. \tag{2.80}$$

The Adjacency

In this case the eigenvalues are bounded as $-d_{\max} \leq \lambda_i \leq d_{\max}$. Hence,

$$s_j = d_{\max} + |\lambda_j|. \tag{2.81}$$

The Normalized Laplacian

In this case the eigenvalues are bounded as $0 \leq \lambda_i \leq 2$. Hence,

$$s_j = 1 + |\lambda_j - 1|. \tag{2.82}$$

Since the selections in (2.80), (2.81), and (2.82) do not use the eigenvalues of the (corresponding) operator $\mathbf{A}$, the polynomial in (2.75) can be constructed using only the target eigenvalue λ_j.

Aforementioned constructions also provide a trade-off between the available information and the rate of convergence. This point will be elaborated in Section 2.7.

2.6.4 Inexact Spectral Information and The Use of Nonlinearity

In this subsection, we will focus on the construction of a second order polynomial when the target eigenvalue λ_j is not known exactly. In this regard, we first assume that we are given an interval $[a \;\; b]$ to which *only* the eigenvalue λ_j belongs. More precisely, we assume the following:

$$\lambda_{j\text{-}1} < a \leq \lambda_j \leq b < \lambda_{j+1}, \tag{2.83}$$

where we consider only the distinct eigenvalues indexed in ascending order, and assume that eigenvalues are real. Then, we consider the following polynomial:

$$h(\lambda) = 1 - 2\epsilon \frac{(\lambda - a)(\lambda - b)}{\big(\max\{\lambda_{\text{upp}} - b,\ a - \lambda_{\text{low}}\} + (b - a)/2\big)^2} \tag{2.84}$$

for some ϵ in $0 < \epsilon < 1$. It should be noted that when the target eigenvalue λ_j is known exactly, we can take $a = b = \lambda_j$, in which case the polynomial in (2.84) reduces to the one in (2.75) with s_j selected as in (2.79). In the case of $a \neq b$, one can observe that the polynomial in (2.84) maps the eigenvalues of the operator $\mathbf{A}$ as follows:

$$h(\lambda_j) > 1 \qquad \text{and} \qquad |h(\lambda_i)| < 1 \quad \forall\, \lambda_i \neq \lambda_j, \tag{2.85}$$

which shows that $h(\mathbf{A})$ does not have a unit eigenvalue, thus the asynchronous updates running on $h(\mathbf{A})$ do not converge. In fact, the signal would diverge due to the dominant eigenvalue, $h(\lambda_j)$, being strictly larger than 1. (See Figure 2.2). In order to prevent the signal from diverging, we consider the following *saturated* update model:

$$\left(\mathbf{x}_k\right)_i = \begin{cases} f\Big(\big(\,h(\mathbf{A})\,\mathbf{x}_{k\text{-}1}\,\big)_i\Big), & i \in \mathcal{T}_k, \\ \left(\mathbf{x}_{k\text{-}1}\right)_i, & i \notin \mathcal{T}_k, \end{cases} \tag{2.86}$$

where $f(\cdot)$ is the "saturation nonlinearity" defined as follows:

$$f(x) = \operatorname{sign}(x) \cdot \min\{|x|,\, 1\}, \tag{2.87}$$

which is visualized in Figure 2.6.

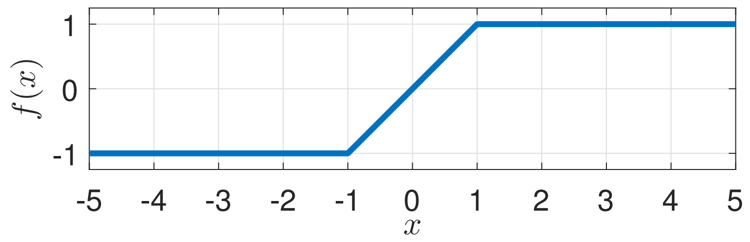


Figure 2.6: Visualization of the saturation nonlinearity defined in (2.87).

Notice that the boundedness of the function $f(\cdot)$ ensures that the signal $\mathbf{x}_k$ in (2.86) does not diverge. In fact, it is numerically observed that randomized asynchronous updates in (2.86) indeed converge to the fixed point of the model. That is, $\mathbf{x}_k$ converges to $\widehat{\mathbf{x}}$, where $\widehat{\mathbf{x}}$ satisfies the following equation:

$$f\big(h(\mathbf{A})\,\widehat{\mathbf{x}}\big) = \widehat{\mathbf{x}}, \tag{2.88}$$

where $f(\cdot)$ is assumed to operate element-wise on a vector.

The key observation regarding the solution of (2.88) is the following: when the interval $[a \;\; b]$ in (2.83) is small, then $\widehat{\mathbf{x}}$ is a good approximation of the target eigenvector $\mathbf{v}_j$. That is,

$$\widehat{\mathbf{x}} \approx \gamma\ \mathbf{v}_j \tag{2.89}$$

for some scale factor $\gamma \in \mathbb{R}$. In fact, when the target eigenvalue is known exactly, i.e., $a = b = \lambda_j$, then the approximation in (2.89) becomes equality. Therefore, we conclude that the asynchronous saturated update scheme in (2.86) allows us to find an arbitrary eigenvector approximately when the corresponding eigenvalue is not known exactly. Furthermore, as we have a better approximation of the eigenvalue, we get a better approximation of the corresponding eigenvector. Section 2.7 will make use of this observation to obtain an autonomous clustering of a network.

2.6.5 Implementation of Second Order Polynomials

In Section 2.6.2 graph signals are shown to converge to an arbitrary eigenvector of the underlying graph operator $\mathbf{A}$ through random asynchronous updates running on an appropriate second order polynomial filter. In this section, we will show that asynchronous updates on a second order polynomial can be implemented as a node-asynchronous communication protocol in which nodes follow a collect-compute-broadcast scheme independently from each other. For this purpose, we first write a second polynomial of $\mathbf{A}$ explicitly as follows:

$$h(\mathbf{A}) = h_0\,\mathbf{I} + h_1\,\mathbf{A} + h_2\,\mathbf{A}^2, \tag{2.90}$$

where the filter coefficients h_0, h_1, h_2 are assumed to be pre-determined such that (2.72) is satisfied for the eigenvalue of the target eigenvector. Then, we define an auxiliary variable $\mathbf{y}$ as:

$$\mathbf{y} = \mathbf{A}\,\mathbf{x}, \tag{2.91}$$

where $\mathbf{x}$ denotes the signal on the graph, and $\mathbf{y}$ is the "graph shifted" signal. We will assume that the i^{th} node stores x_i and y_i simultaneously. Thus, (x_i, y_i) can be considered as the state of the i^{th} node. Then, we can write the following:

$$(h(\mathbf{A})\,\mathbf{x})_i = h_0\,x_i + h_1\,y_i + h_2\,(\mathbf{A}\,\mathbf{y})_i. \tag{2.92}$$

Using (2.92), asynchronous updates with $\delta_{\mathrm{T}} = 1$ (only one node is updated per iteration) running on $h(\mathbf{A})$ can be equivalently written in the following three steps:

$$u \leftarrow (h_0 - 1)\,x_i + h_1\,y_i + h_2\,\mathbf{A}_{[i,:]}\,\mathbf{y}, \tag{2.93}$$

$$x_i \leftarrow x_i + u, \tag{2.94}$$

$$\mathbf{y} \leftarrow \mathbf{y} + \mathbf{A}_{[:,i]}\,u, \tag{2.95}$$

where $\mathbf{A}_{[i,:]}$ and $\mathbf{A}_{[:,i]}$ denote the i^{th} row and column of $\mathbf{A}$, respectively.

It is important to note that equations in (2.93)-(2.95) are in the form of a collect-compute-broadcast scheme. In (2.93), the term $\mathbf{A}_{[i,:]}\,\mathbf{y}$ requires the i^{th} node to collect y_j's from all $j \in \mathcal{N}_{\mathrm{in}}(i)$. In (2.94), the node simply updates its own signal. In (2.95), the term $\mathbf{A}_{[:,i]}\,u$ requires the i^{th} node to broadcast u to all $j \in \mathcal{N}_{\mathrm{out}}(i)$. These three steps can be converted into a node-asynchronous communication protocol as in Algorithm 1.

Algorithm 1 consists of three procedures: initialization, active stage, and passive stage. In the initialization, the i^{th} node assigns a random value to its own signal

Algorithm 1 Node-Asynchronous Communication Protocol

```
procedure INITIALIZATION($i$)
    Initialize $x_i$ randomly.
    Collect $x_j$ from nodes $j \in \mathcal{N}_{\text{in}}(i)$.
    $y_i \leftarrow \sum_{j \in \mathcal{N}_{\text{in}}(i)} A_{i,j}\; x_j$.
procedure PASSIVE STAGE($i$)
    if a broadcast $u$ is received from the node $j$ then
        $y_i \leftarrow y_i + A_{i,j}\; u$.
    if the node $j$ sends a request then
        Send $y_i$ to node $j$.
procedure ACTIVE STAGE($i$)
    Collect $y_j$ from nodes $j \in \mathcal{N}_{\text{in}}(i)$.
    $u \leftarrow (h_0 - 1)\, x_i + h_1\, y_i + h_2\, \sum_{j \in \mathcal{N}_{\text{in}}(i)} A_{i,j}\; y_j$.
    $x_i \leftarrow x_i + u$.
    Broadcast $u$ to all nodes $j \in \mathcal{N}_{\text{out}}(i)$.
```

x_i, then it constructs the auxiliary variable y_i by collecting x_j from its neighbors. Once the initialization is completed, the i^{th} node waits in the passive stage, in which the graph signal x_i is not updated. However, its neighbors can request y_i, or send a broadcast. When a broadcast is received, the i^{th} node updates only its auxiliary variable y_i. When the i^{th} node wakes up randomly, it gets into the active stage, in which it collects the auxiliary variable y_j from its neighbors, updates its signal x_i, and then broadcasts the amount of update to its neighbors. Then, the node goes back to the passive stage.

Five comments are in order: 1) The random update model in (2.9) implies that all nodes have the same probability of going into the active stage. 2) As the signal $\mathbf{x}$ converges to the target eigenvector of $\mathbf{A}$, the ratio y_i/x_i converges to the corresponding eigenvalue of $\mathbf{A}$ (assuming x_i is non-zero), thus $h(y_i/x_i)$ converges to 1 due to (2.72). 3) In the active stage, the broadcast (the step in (2.95)) is essential to ensure that $\mathbf{x}$ and $\mathbf{y}$ satisfy (2.91). 4) The amount of update for x_i is computed by the i^{th} node itself. The amount of update for y_i is dictated by the neighbors of the i^{th} node. Thus, y_i can also be considered as a buffer. 5) Since edges are allowed to be directed, a node may collect data from the j^{th} node in the active stage, but may not send data back to the j^{th} node.

As a final remark we note that Algorithm 1 assumes reliable communication between the nodes, i.e., no link failures. In this case Algorithm 1 is exactly equivalent to the model in (2.5) running on $h(\mathbf{A})$. As long as the polynomial coefficients are selected properly (see Theorem 2.4), the signal $\mathbf{x}$ in Algorithm 1 is guaranteed to converge to the eigenvector targeted by the polynomial. In the case of link failures, Algorithm 1 deviates from the model in (2.5), thus the convergence guarantees presented here are not applicable. Nevertheless, we have numerically observed that *Algorithm 1 converges even in the case of random link failures*. This case will be studied in future.

2.7 An Application: Autonomous Clustering

In this section we will consider the problem of clustering in autonomous (ad-hoc) networks [212, 13, 105]. For this purpose we will combine the well-known spectral clustering [135] with the polynomial filtering proposed in the preceding section.

Given a network, the second smallest eigenvalue of its graph Laplacian, λ_2, is known as the *algebraic connectivity* of the graph [63]. Roughly speaking graphs with larger λ_2 tend to be more "connected" than the others. Furthermore, the corresponding eigenvector $\mathbf{v}_2$, also known as the *Fiedler vector*, can be utilized to cluster the graph into two partitions. The signal $\mathbf{x}$ computed as

$$\mathbf{x} = \text{sign}(\mathbf{v}_2), \tag{2.96}$$

indicates the corresponding cluster of the nodes. Similar spectral ideas are used in [5] to obtain approximate graph coloring and in Section 6.7 to identify the hidden M-Block cyclic structure from noisy measurements under random permutations.

In the following we will consider the idea of asynchronous polynomial filtering in order to compute the eigenvector $\mathbf{v}_2$ of the Laplacian. For this purpose λ_2 will be selected as the target eigenvalue. As a result, nodes will be able to identify the cluster they belong to in an autonomous manner. Such a behavior can be considered as swarm intelligence as well: independent simple computation by individual agents (nodes) can obtain a global information regarding the whole community (graph) [25].

For the graph visualized in Figure 2.7a, the result of the spectral clustering based on (2.96) is demonstrated in Figure 2.7b where the clusters are represented with different colors. In the remaining, labels found by (2.96) will be referred to as the correct labels. The autonomous clustering on this network is simulated using the following four different polynomial filters:

1. The optimal third order filter via (2.74).

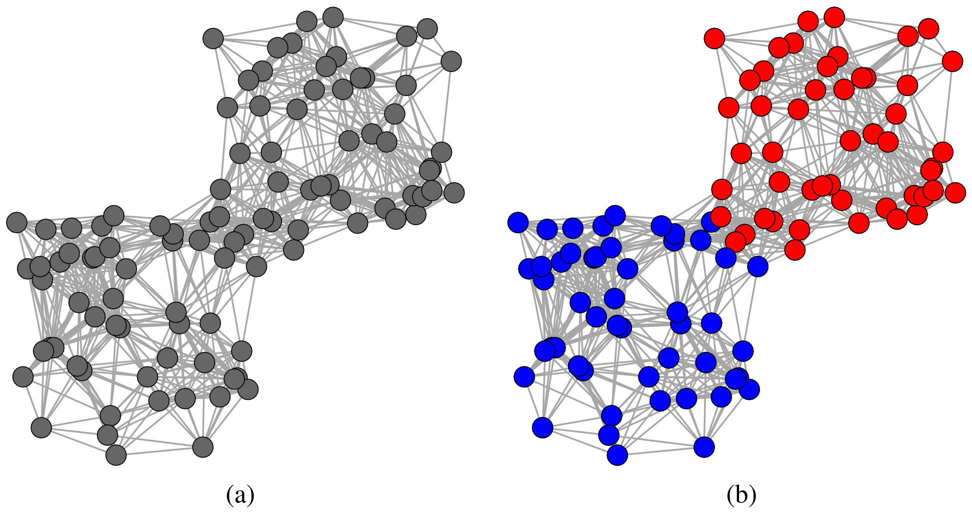


Figure 2.7: (a) A graph on $N = 100$ nodes with 2 clusters. The graph has undirected edges with binary weights. (b) The result of the spectral clustering based on (2.96) with colors representing the clusters.

2. The optimal second order filter via (2.74).

3. A second order filter of (2.75) by selecting s_2 as in (2.79) using λ_2, $\lambda_{\max}$, and setting $\lambda_{\min} = 0$.

4. A second order filter of (2.75) by selecting s_2 as in (2.80) using λ_2 and $d_{\max}$.

5. The filter in (2.84) and setting $a = 0.3$, $b = 1.5$, $\lambda_{\text{low}} = 0$ $\lambda_{\text{upp}} = 2\, d_{\max}$.

In the simulations all the nodes are initialized randomly, and the random asynchronous iterations run on the constructed polynomials of the Laplacian. We use $\delta_{\text{T}} = 1$, i.e., one node is randomly chosen and updated at every iteration. The label of a node is the sign of its most recent value as in (2.96). The average fraction of incorrect labels versus number of iterations is presented in Figure 2.8 for five filters mentioned above.

Figure 2.8 shows that the number of incorrect labels go down to zero as iterations progress, which is proven to be the case by Corollary 2.2 due to the construction of the filters in Section 2.6. For filter #5, the saturated model does not converge to $\mathbf{v}_2$ exactly as explained in Section 2.6.4. Nevertheless, the approximation in (2.89) is good enough so that the sign pattern of the fixed point of the updates, defined in (2.88), matches the correct labels.

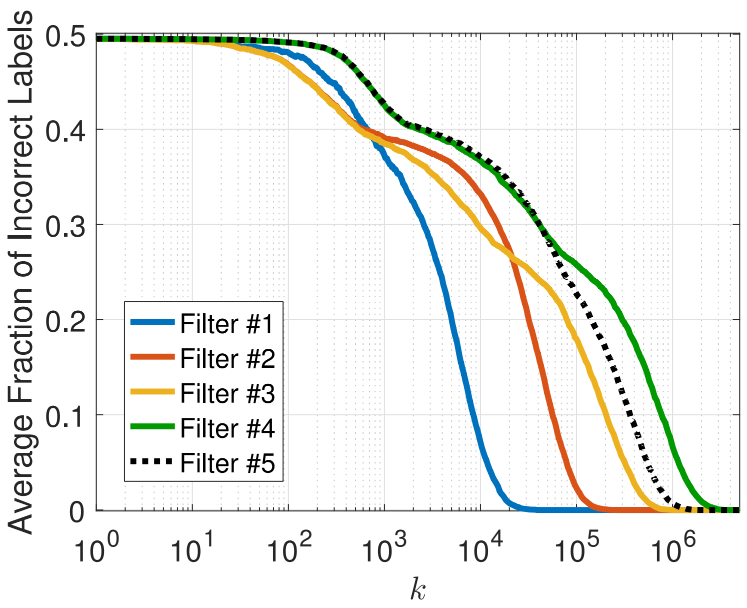


Figure 2.8: Results of the autonomous clustering experiment, which are obtained by averaging over 10^4 independent experiments.

The figure also illustrates the trade-off between the complexity, the amount of spectral information used and the rate of convergence. Although filters #1 and #2 are constructed using all the eigenvalues of the Laplacian, filter #1 yields a faster convergence due to its higher order (complexity). Filters #2, #3, and #4 have the same order, but their constructions use lesser and lesser amounts of spectral information. As a result, they yield lower and lower rates of convergence. Interestingly, filter #5 converges faster than filter #4 although it uses the least amount of spectral information. This is an interesting consequence of the fact that the filter in (2.84) results in a larger spectral gap than the one in (2.75) for the amount of spectral information used.

2.8 An Application: Randomized Computation of Singular Vectors

In most of the data related applications $\mathbf{A}$ happens to be a rectangular matrix, and procedures such as principal component analysis (PCA) require the *singular vectors* of the matrix $\mathbf{A} \in \mathbb{C}^{M\times N}$. Due to the importance of the singular vectors, many efforts have been made to develop fast algorithms, especially randomized ones [78, 155, 156, 208, 121]. Although the random component-wise update considered in (2.5) is not directly applicable to rectangular matrices, a reformulation allows us to compute the dominant singular vectors with random component-wise updates. In this regard, we start by assuming that $M \leq N$ without loss of generality, and then consider the *Hermitian dilation* of the given matrix $\mathbf{A}$, which is defined as follows:

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^{\mathrm{H}} & \mathbf{0} \end{bmatrix} \in \mathbb{C}^{(M+N)\times(M+N)}. \tag{2.97}$$

It is clear that $\bar{\mathbf{A}}$ is an Hermitian matrix, i.e., $\bar{\mathbf{A}}^{\mathrm{H}} = \bar{\mathbf{A}}$. More importantly, the eigenvalue decomposition of $\bar{\mathbf{A}}$ is closely related to the singular value decomposition of $\mathbf{A}$. More precisely, let $\mathbf{A}$ have the following reduced form singular value decomposition:

$$\mathbf{A} = \widetilde{\mathbf{U}}\,\mathbf{\Sigma}\,\widetilde{\mathbf{V}}^{\mathrm{H}} \qquad \text{where} \qquad \widetilde{\mathbf{U}} \in \mathbb{C}^{M\times M}, \quad \mathbf{\Sigma} \in \mathbb{R}^{M\times M}, \quad \widetilde{\mathbf{V}} \in \mathbb{C}^{N\times M} \tag{2.98}$$

where $\widetilde{\mathbf{U}}$ and $\widetilde{\mathbf{V}}$ are the matrices consisting of the left and right singular vectors of $\mathbf{A}$, respectively, and $\mathbf{\Sigma}$ is the diagonal matrix consisting of the singular values of $\mathbf{A}$, i.e., $\Sigma_{i,i} = \sigma_i$. We assume that the singular values are in the descending order, i.e., $\sigma_1 \geq \cdots \geq \sigma_M$. Then, the eigenvalue decomposition of $\bar{\mathbf{A}}$ is as follows:

$$\bar{\mathbf{A}} = \mathbf{Q}\,\mathbf{\Lambda}\,\mathbf{Q}^{\mathrm{H}}, \tag{2.99}$$

where

$$\mathbf{Q} = \frac{1}{\sqrt{2}} \begin{bmatrix} \widetilde{\mathbf{U}} & \widetilde{\mathbf{U}} \\ \widetilde{\mathbf{V}} & -\widetilde{\mathbf{V}} \end{bmatrix} \in \mathbb{C}^{(M+N)\times 2M}, \qquad \mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & -\mathbf{\Sigma} \end{bmatrix} \in \mathbb{R}^{2M\times 2M}. \tag{2.100}$$

If $\mathbf{A}$ is a rank-r matrix, then it is clear from (2.100) that $\bar{\mathbf{A}}$ has $2r$ non-zero eigenvalues that come in positive and negative pairs corresponding to the non-zero singular values of $\mathbf{A}$, and $\bar{\mathbf{A}}$ has $N + M - 2r$ zero eigenvalues. Furthermore, the eigenvectors of $\bar{\mathbf{A}}$ clearly reveal the left and right singular vectors of $\mathbf{A}$ by inspection. As a result, if the eigenvalue decomposition of $\bar{\mathbf{A}}$ given in (2.100) is obtained numerically, then the singular value decomposition of $\mathbf{A}$ is readily available.

In order to obtain the dominant eigenvector of $\bar{\mathbf{A}}$ (which correspond to the dominant left and right singular vectors of $\mathbf{A}$), we will consider the random component-wise updates of (2.5) running on the matrix $\bar{\mathbf{A}}$:

$$(\bar{\mathbf{x}}_k)_i = \begin{cases} (\bar{\mathbf{A}}\,\bar{\mathbf{x}}_{k\text{-}1})_i, & i \in \mathcal{T}_k, \\ (\bar{\mathbf{x}}_{k\text{-}1})_i, & i \notin \mathcal{T}_k, \end{cases} \tag{2.101}$$

where we assume that only one index is updated per iteration, i.e., $|\mathcal{T}_k| = 1$, for the sake of simplicity. The case of updating more than one index is a straightforward generalization of the approach considered here.

Let the iterand $\bar{\mathbf{x}}_k \in \mathbb{C}^{M+N}$ in (2.101) be partitioned as follows:

$$\bar{\mathbf{x}}_k = \begin{bmatrix} \mathbf{u}_k \\ \mathbf{v}_k \end{bmatrix}, \qquad \text{where} \qquad \mathbf{u}_k \in \mathbb{C}^M, \quad \mathbf{v}_k \in \mathbb{C}^N. \tag{2.102}$$

Then, the update scheme (2.101) can be expressed equivalently using the matrix $\mathbf{A}$ itself and the partitions $\mathbf{u}_k$ and $\mathbf{v}_k$. More precisely, we have the following:

$$\begin{aligned} &\text{If} \quad i \leq M, \qquad (\bar{\mathbf{x}}_k)_i = (\mathbf{u}_k)_i, \qquad \text{and} \qquad (\bar{\mathbf{A}}\,\bar{\mathbf{x}}_k)_i = \mathbf{A}_{[i,:]}\,\mathbf{v}_k, \\ &\text{If} \quad i > M, \qquad (\bar{\mathbf{x}}_k)_i = (\mathbf{v}_k)_{i\text{-}M}, \quad \text{and} \qquad (\bar{\mathbf{A}}\,\bar{\mathbf{x}}_k)_i = (\mathbf{A}_{[:,i\text{-}M]})^{\mathrm{H}}\,\mathbf{u}_k. \end{aligned} \tag{2.103}$$

Due to (2.103) and the assumption that only one index is updated per iteration, the k^{th} iteration of (2.101) can be described as follows: select an index i randomly uniformly from the set $\{1, \cdots, M{+}N\}$. If the index corresponds to a row, i.e., $i \leq M$, then the i^{th} index of $\mathbf{u}_{k\text{-}1}$ is updated as the inner product between the i^{th} row of $\mathbf{A}$ and the vector $\mathbf{v}_{k\text{-}1}$. If the index corresponds to a column, i.e., $i > M$, then the $(i - M)^{th}$ index of $\mathbf{v}_{k\text{-}1}$ is updated as the inner product between the $(i - M)^{th}$ column of $\mathbf{A}$ and the vector $\mathbf{u}_{k\text{-}1}$. These updates are described as a pseudo-code in Algorithm 2.

Algorithm 2 Dominant Singular Vector without a Normalization Step

```
1:  Assume $\|\mathbf{A}\|_2 = 1$.
2:  Initialize $\mathbf{u} \in \mathbb{R}^M$, $\mathbf{v} \in \mathbb{R}^N$.
3:  while convergence do
4:      $i \sim \mathcal{U}\{1, \cdots, M + N\}$
5:      if $i \leq M$ then
6:          $u_i \leftarrow \mathbf{A}_{[i,:]}\,\mathbf{v}$
7:      else
8:          $i \leftarrow (i - M)$
9:          $v_i \leftarrow \left(\mathbf{A}_{[:,i]}\right)^{\mathrm{H}}\,\mathbf{u}$
10: return u and v
```

Algorithm 3 Dominant r Singular Vectors with a Normalization Step

```
1:  Initialize $\mathbf{C}, \mathbf{U} \in \mathbb{R}^{M\times r}$, $\mathbf{V} \in \mathbb{R}^{N\times r}$.
2:  while convergence do
3:      $i \sim \mathcal{U}\{1, \cdots, M + N\}$
4:      if $i \leq M$ then
5:          $\mathbf{C}_{[i,:]} \leftarrow \mathbf{A}_{[i,:]}\,\mathbf{V}$
6:          Set U as $\mathbf{C} = \mathbf{U}\,\mathbf{T}$ with $T_{i,i} \geq 0$.
7:      else
8:          $i \leftarrow (i - M)$
9:          $\mathbf{V}_{[i,:]} \leftarrow \left(\mathbf{A}_{[:,i]}\right)^{\mathrm{H}}\,\mathbf{U}$
10: return U and V
```

Since $\bar{\mathbf{A}}$ is a normal matrix irrespective of the value of $\mathbf{A}$ and Algorithm 2 is equivalent to the update scheme (2.101), we can use Theorem 2.2 to study the convergence of the algorithm. The following theorem shows that Algorithm 2 indeed converges to the left and right dominant singular vectors of the matrix $\mathbf{A}$.

Theorem 2.6. *Assume that the matrix $\mathbf{A} \in \mathbb{C}^{M\times N}$ has $\|\mathbf{A}\|_2 = 1$, i.e. the maximum singular value is unity. Assume further that the eigenspace parameter ρ of the* *matrix* $\bar{\mathbf{A}}$ *satisfies* $\rho < 1$. *Then,* $\mathbf{u}$ *and* $\mathbf{v}$ *in Algorithm 2 converge to the left and right dominant singular vectors of* $\mathbf{A}$, *respectively.*

Proof. We start by noting that Algorithm 2 is equivalent to the update scheme of (2.101) due to the equivalence in (2.103). Thus, the convergence of (2.101) is equivalent to the convergence of the algorithm.

Let σ_i denote the i^{th} largest singular value of $\mathbf{A}$. Since $\|\mathbf{A}\|_2 = 1$, we have $1 = \sigma_1 \geq \sigma_2 \geq \cdots \sigma_M \geq 0$. From (2.100) we have that $\bar{\mathbf{A}}$ is an Hermitian matrix with eigenvalues $\lambda(\bar{\mathbf{A}}) = \pm\,\sigma(\mathbf{A})$ with additional $N - M$ zero eigenvalues. Thus, $\bar{\mathbf{A}}$ has a unit eigenvalue, and its non-unit eigenvalues satisfy -1 $\leq \lambda_i < 1$.

Due to random component-wise nature of the update scheme in (2.101), we have $\delta_{\mathrm{T}} > 0$. In addition, we assume that the eigenspace parameter of $\bar{\mathbf{A}}$ satisfies $\rho < 1$. Thus, Corollary 2.2 shows that the convergence region given in (2.41) contains the interval [-1 1) due to the fact that $\alpha = \delta_{\mathrm{T}}\,(\rho - 1) < 0$. Since all non-unit eigenvalues of $\bar{\mathbf{A}}$ satisfy -1 $\leq \lambda_i < 1$, Corollary 2.2 guarantees that the iterand $\bar{\mathbf{x}}_k$ in (2.101) converges to an eigenvector of $\bar{\mathbf{A}}$ with the eigenvalue 1. Then, it is clear from (2.100) that the partitions of $\bar{\mathbf{x}}_k$ defined in (2.102) converge to the left and right singular vectors of $\mathbf{A}$ corresponding to the singular value 1. That is, $\mathbf{u}_k$ converges to the left dominant singular vector $\mathbf{A}$, and $\mathbf{v}_k$ converges to the right dominant singular vector of $\mathbf{A}$. □

The convergence guarantee provided by Theorem 2.6 is based on two assumptions. The first one, $\|\mathbf{A}\|_2 = 1$, is in fact a necessary condition for the convergence of the updates in (2.101). Existence of a singular value 1 implies the existence of a fixed point of the component-wise updates. The second one, which requires the eigenspace parameter to satisfy $\rho < 1$, is a technical assumption that is needed to prove the convergence precisely. It is not a necessary condition, and it is in fact satisfied in many practical applications. It is also important to note that Theorem 2.6 does not assume realness of the matrix $\mathbf{A}$. The algorithm is guaranteed to converge even when $\mathbf{A}$ is a complex matrix, in which case dominant singular vectors are complex as well.

2.8.1 Rank-r Approximation of an Arbitrary Matrix

Although the assumption $\|\mathbf{A}\|_2 = 1$ required by Algorithm 2 can be satisfied easily by normalizing the data matrix $\mathbf{A}$ with its largest singular value, the computation of the largest singular value itself may not be very practical especially when $\mathbf{A}$ is large in dimensions. It is, in fact, possible to remove this assumption by introducing a normalization step into the algorithm. Furthermore, it is also possible to extend Algorithm 2 in such a way that it converges to the dominant r singular vectors of $\mathbf{A}$ together with the top-r singular values for an arbitrary value of r. The extended version of the algorithm is presented in Algorithm 3.

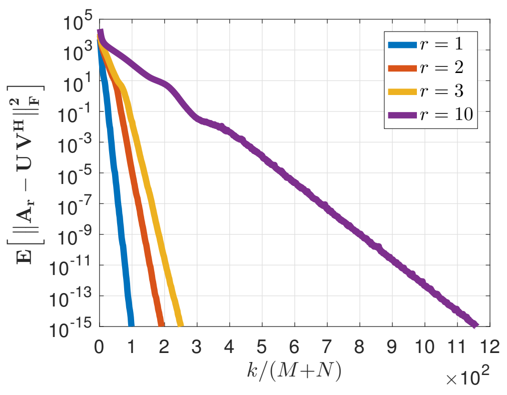
Algorithm 3 differs from Algorithm 2 in three ways: Firstly, the vector variables $\mathbf{u}$ and $\mathbf{v}$ in Algorithm 2 are extended to be matrices with r columns. Secondly, Algorithm 3 uses an auxiliary variable $\mathbf{C}$. Thirdly, and the most importantly, Algorithm 3 uses a *QR decomposition* (Line 6) that serves as the normalization step. More precisely, instead of updating the variable $\mathbf{U}$ directly, Algorithm 3 first updates the auxiliary variable $\mathbf{C}$ (Line 5), and then updates $\mathbf{U}$ as the unitary part of the QR decomposition of $\mathbf{C}$. We note that the matrix $\mathbf{T}$ in Line 6 of the algorithm denotes the upper-triangular part of the QR decomposition of $\mathbf{C}$. Without loss of generality, it is assumed that $\mathbf{T}$ has non-negative diagonal entries, and its diagonal entries are in the descending order.

Although the convergence of Algorithm 2 is ensured by Theorem 2.6, we do not provide an explicit proof for the convergence of Algorithm 3. Nevertheless, by the virtue of Theorem 2.6 we can argue for the convergence of Algorithm 3 since it is a natural extension of Algorithm 2 with an additional normalization step. We observe that the variables of Algorithm 3 converge as follows:

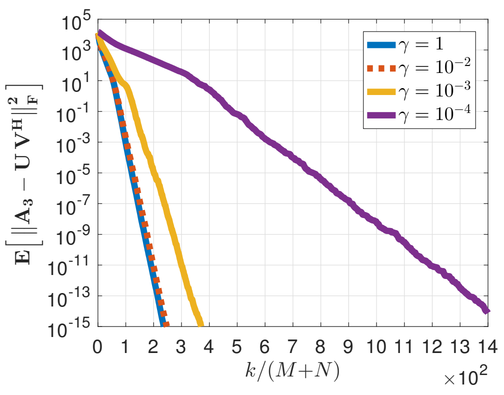
$$\mathbf{U} \to \widetilde{\mathbf{U}}_r, \qquad \mathbf{T} \to \mathbf{\Sigma}_r^2, \qquad \mathbf{V} \to \widetilde{\mathbf{V}}_r\, \mathbf{\Sigma}_r, \tag{2.104}$$

where $\widetilde{\mathbf{U}}_r$ and $\widetilde{\mathbf{V}}_r$ are the first r columns of $\widetilde{\mathbf{U}}$ and $\widetilde{\mathbf{V}}$, respectively, and $\mathbf{\Sigma}_r$ is the top-left $r \times r$ block of $\mathbf{\Sigma}$. Thus, the product $\mathbf{U}\,\mathbf{V}^{\mathrm{H}}$ converges to $\mathbf{A}_r$, which is the best rank-r approximation of $\mathbf{A}$, i.e, $\mathbf{A}_r = \widetilde{\mathbf{U}}_r\, \mathbf{\Sigma}_r\, \widetilde{\mathbf{V}}_r^{\mathrm{H}}$.

In order to verify its convergence, we simulate Algorithm 3 on the test matrix MEDLINE [188], which is a full-rank and sparse matrix of size 1033×5735. We measure the convergence of the algorithm in terms of the squared Frobenius norm of the difference between $\mathbf{A}_r$ and the product $\mathbf{U}\,\mathbf{V}^{\mathrm{H}}$. Since the update index is selected randomly (Line 3) in every iteration of Algorithm 3, the error term, i.e, $\|\mathbf{A}_r - \mathbf{U}\,\mathbf{V}^{\mathrm{H}}\|_{\mathrm{F}}^2$, is a random variable as well. So, we compute the expected error by averaging over 10^3 independent runs of the algorithm. These results are presented in Figure 2.9a for the cases of $r \in \{1, 2, 3, 10\}$, which numerically verify the convergence of the algorithm. Note that the algorithm requires more iterations to converge as the value of r gets larger.



(a)



(b)

Figure 2.9: (a) Convergence of Algorithm 3 for various different values of r. (b) Convergence of Algorithm 3 for the case $r = 3$ when the normalization step in Line 6 is executed with probability γ. Here k indicates the number of iterations.

We note that Line 5 of Algorithm 3 updates only a row of the auxiliary variable $\mathbf{C}$ in every iteration. Since the matrix $\mathbf{C}$ is not expected to change significantly during an iteration, the normalization step in Line 6 can be skipped in some iterations in order to reduce the overall computational complexity of the algorithm. In order to verify this claim, we modify the implementation of the algorithm in such a way that Line 6 is executed with probability γ. So, the modified implementation reduces to Algorithm 3 when $\gamma = 1$. For the case of $r = 3$, we compute the expected error of the modified implementation by averaging over 10^3 independent runs. These results are presented in Figure 2.9b for the values of $\gamma \in \{1,\ 10^{-2},\ 10^{-3},\ 10^{-4}\}$, which shows that the modified implementation keeps converging for wide range of values of γ. More interestingly, the rate of convergence remains visually the same even when the normalization step is executed with probability as low as $\gamma = 10^{-2}$. Moreover, the rate of convergence decreases marginally when $\gamma = 10^{-3} \approx 1/M$. This is consistent with the fact that an iteration of Algorithm 3 updates only one row of $\mathbf{C}$ that has M rows in total. Nevertheless, when γ has a very small value, e.g., $\gamma = 10^{-4}$, the algorithm indeed gets significantly slower.

Regarding the computational complexity of the algorithm, note that the cost of Line 5, Line 6 and Line 9 are $O(Nr)$, $O(Mr^2)$, and $O(Mr)$, respectively. However, the algorithm gets to Lines 5 and 6 with probability $M/(M{+}N)$, and it gets to Line 9 with probability $N/(M{+}N)$. When we further assume that Line 6 is executed with probability γ, *the average cost* of an iteration of Algorithm 3 can be found as follows:

$$\mathbb{E}[\text{computatinal cost per iteration}] = O\left(\frac{MNr + \gamma\, M^2 r^2}{M + N}\right) \approx O\left(\frac{MNr}{M + N}\right), \quad (2.105)$$

where the approximation is valid when $\gamma \leq N/(Mr)$, which is acceptable in practice as suggested by Figure 2.9b. On the other hand, the synchronous form of (2.101) requires $O(MNr)$ multiplications per iteration. In order to compensate the additional factor of $M{+}N$, the iteration index k is normalized by $M{+}N$ in both Figures 2.9a and 2.9b.

Relevance of Algorithm 3 follows from its applicability for asynchronous and distributed implementation. Since a single iteration of the algorithm requires a partial information of the matrix $\mathbf{A}$, (i.e., a single column or row) multiple processors can operate on the same matrix $\mathbf{A}$ simultaneously without requiring any ordering among them. More importantly, it is possible to extend Algorithm 3 in such a way that the data matrix $\mathbf{A}$ is partitioned into multiple smaller pieces, and each piece is stored in a different processing core as we discuss next.

2.8.2 Distributed Implementation with Partial Data Storage

In this section we will assume that the matrix $\mathbf{A} \in \mathbb{C}^{M\times N}$ represents a collection of data points where each column of the matrix is a data point in the M-dimensional feature space, and $\mathbf{A}$ has N data points in total. In some of the applications the number of the data points, N, can be too large for $\mathbf{A}$ to be stored in a single core. Thus, the data needs to be partitioned into smaller collections and stored in different cores. It could also be the case that the data is already located in different places and may not be available directly due to privacy concerns. The asynchronous (component-wise) nature of Algorithm 3 makes it suitable to compute the dominant singular vectors of $\mathbf{A}$ in these scenarios. Although Algorithm 3 is not directly applicable, it can be modified to handle these scenarios as well. For this purpose assume that the matrix $\mathbf{A}$ is partitioned into P blocks as follows:

$$\mathbf{A} = [\mathbf{A}^{(1)} \quad \mathbf{A}^{(2)} \quad \cdots \quad \mathbf{A}^{(P)}], \tag{2.106}$$

where $\mathbf{A}^{(p)} \in \mathbb{C}^{M\times N_p}$ denotes the p^{th} partition holding corresponding N_p data points, so $\sum_{p=1}^{P} N_p = N$.

When Algorithm 3 is utilized on the partitioned data, the column selection phase can be done in a straightforward manner since columns of $\mathbf{A}$ are assumed to be the individual data points. On the contrary, the row selection phase of Algorithm 3 (Line 5) requires to access the same row of *all the partitions*. Due to the distributed nature of the data it may not be possible for a processor to access all the partitions simultaneously. Nevertheless, the required inner product can be written as a sum of partial inner products, which then can be computed locally. For this purpose, we first partition the variable $\mathbf{V} \in \mathbb{C}^{N \times r}$ in Algorithm 3 with respect to the partitions given in (2.106). More precisely, assume that the p^{th} core holds a *local* variable $\mathbf{V}^{(p)} \in \mathbb{C}^{N_p \times r}$. Then, the inner product involving the rows of $\mathbf{A}$ can be written as follows:

$$\mathbf{A}_{[i,:]}\, \mathbf{V} = \sum_{p=1}^{P} \mathbf{A}^{(p)}_{[i,:]}\, \mathbf{V}^{(p)}. \tag{2.107}$$

Notice that the quantity $\mathbf{A}^{(p)}_{[i,:]}\, \mathbf{V}^{(p)}$ can be computed in the p^{th} core locally. Once it is obtained, it can be sent to a fusion center in order to update the value of the variable $\mathbf{U}$. Based on this observation, we propose Algorithm 4 for the computation of the dominant singular vectors for the case of distributed data.

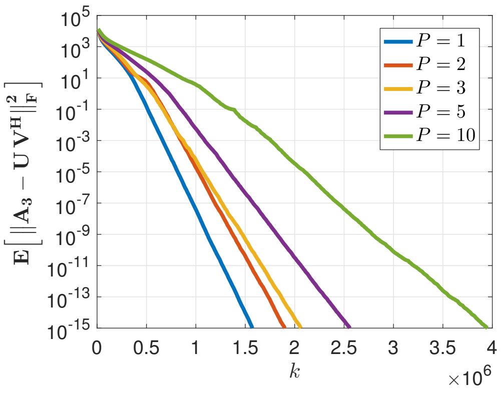
Algorithm 4 Distributed Computation of Dominant r Singular Vectors

```
 1: Initialize $\mathbf{C} \in \mathbb{R}^{M \times r \times P}$, $\mathbf{U} \in \mathbb{R}^{M \times r}$, $\mathbf{V}^{(1)} \in \mathbb{R}^{N_1 \times r}, \cdots, \mathbf{V}^{(P)} \in \mathbb{R}^{N_P \times r}$ randomly.
 2: **while** convergence **do**
 3:     $p \sim \mathcal{U}\{1, \cdots, P\}$                                   ▹ Select a partition randomly
 4:     $i \sim \mathcal{U}\{1, \cdots, M + N_p\}$                            ▹ Select an index randomly
 5:     **if** $i \le M$ **then**
 6:         $\mathbf{C}_{[i,:,p]} \leftarrow \mathbf{A}^{(p)}_{[i,:]}\, \mathbf{V}^{(p)}$                     ▹ Partial inner product
 7:         Compute $\widehat{\mathbf{C}}$ as $\widehat{C}_{i,j} = \sum_{p=1}^{P} C_{i,j,p}$              ▹ Data fusion
 8:         Set $\mathbf{U}$ as $\widehat{\mathbf{C}} = \mathbf{U}\,\mathbf{T}$ with $T_{i,i} \ge 0$.        ▹ QR decomposition
 9:     **else**
10:         $i \leftarrow (i - M)$
11:         $\mathbf{V}^{(p)}_{[i,:]} \leftarrow \left(\mathbf{A}^{(p)}_{[:,i]}\right)^{\mathrm{H}} \mathbf{U}$          ▹ Update the local variable
```

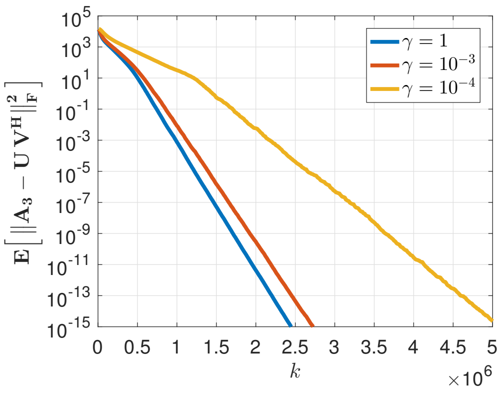
It is important to note that Algorithm 4 can be implemented in a distributed manner with the help of a data fusion center and a shared memory. In such implementation the variable $\mathbf{C}$ corresponds to the data collected by the fusion center, and the variable $\mathbf{U}$ is assumed to be in a shared memory. On the contrary, the data matrix $\mathbf{A}^{(p)}$ and the variable $\mathbf{V}^{(p)}$ are stored in the p^{th} processor locally. Then, Algorithm 4 proceeds as follows: it first selects the p^{th} partition randomly uniformly among all P partitions (Line 3). Then, the processor randomly selects and performs either one of the following two actions: 1) It uses the local variable $\mathbf{V}^{(p)}$, computes a partial inner product, and sends it to the fusion center (Line 6). 2) It uses the global shared variable $\mathbf{U}$ in order to update the local variable $\mathbf{V}^{(p)}$ (Line 11). In the mean-time, whenever the fusion center is updated, it first computes the full inner products (Line 7), and then updates the global variable $\mathbf{U}$ via QR decomposition (Line 8).

We also note that in the case of a single partition, i.e., $P = 1$, Algorithm 4 reduces to Algorithm 3.

Algorithm 4 has two major benefits. First, the p^{th} processor uses the p^{th} data partition only, thus all the computations can be done locally on a processor. More importantly, the data itself is never shared: a summary of the data (in the form an inner product) is sent to the fusion center. Only shared information is the variable $\mathbf{U}$, which converges to the dominant r left singular vectors of the whole data matrix $\mathbf{A}$. Secondly, due to the randomized nature of the algorithm, a synchronization between the processors is not required. The processors are allowed to interact with the fusion center independently at any order.



(a)



(b)

Figure 2.10: (a) Convergence of Algorithm 4 for the case of $r = 3$, i.e., rank-3 approximation, for different numbers of P data partitions. (b) Convergence of Algorithm 4 for the case $r = 3$ with $P = 5$ partitions when the normalization step in Line 8 is executed with probability γ. Here k indicates the number of iterations.

Similar to Algorithm 3, the variable $\mathbf{U}$ of Algorithm 4 converges to $\widetilde{\mathbf{U}}_r$, and the variables $\mathbf{V}^{(p)}$ converge to the corresponding partitions of $\widetilde{\mathbf{V}}_r\,\mathbf{\Sigma}_r$ similar to (2.104). In order to verify the convergence of Algorithm 4 numerically, we use the test matrix MEDLINE [188] and divide it into P blocks as in (2.106), where each partition has size (approximately) N/P. We consider the case of $r = 3$, i.e., rank-3 approximation, and the error is computed as the expected value of $\|\mathbf{A}_3 - \mathbf{U}\,\mathbf{V}^{\mathrm{H}}\|_{\mathrm{F}}^2$ where $\mathbf{V}$ denotes the matrix constructed by cascading the local variables $\mathbf{V}^{(p)}$. The expected error is computed by averaging over 10^3 independent runs of the algorithm, and the results are presented in Figure 2.10a for the cases of $P \in \{1, 2, 3, 5, 10\}$, which verify the convergence numerically.

In order to demonstrate the robustness of Algorithm 4 to stale data in the case of distributed implementation, we modify the algorithm in such a way that the normalization step in Line 8 is executed with probability γ. Thus, the processors do not always use the value of **U** that corresponds to the most recent value of **C**; rather, they use an outdated value of **U**. The modified implementation is simulated for the case of $r = 3$ with $P = 5$ partitions, and the error is computed as before by averaging over 10^3 independent runs. The simulation results are presented in Figure 2.10b for the values of $\gamma \in \{1, 10^{-3}, 10^{-4}\}$. It is clear from the figure that even when the shared memory (the variable **U**) is updated with probability as low as $\gamma = 10^{-3} \approx 1/M$, the modified implementation continues to converge as fast as Algorithm 4 itself, which indicates the robustness of the algorithm to the use of stale data.

2.9 Concluding Remarks

In this chapter, we proposed a node-asynchronous communication protocol in which nodes follow a collect-compute-broadcast scheme randomly and independently from each other. Different than the consensus, this protocol can converge to an arbitrary eigenvector of the graph operator of interest. In order to analyze the convergence behavior, we introduced a randomized asynchronous variant of the power iteration, which performs the regular power iteration (or, the graph shift) but only a random subset of the indices are updated. Assuming that the underlying operator has eigenvalue 1, we proved that repetition of such randomized updates converges to an eigenvector of the eigenvalue 1 (a fixed point) even in the case of operator having other eigenvalues with magnitudes larger than one. We also showed that not only the eigenvalue gap but also the eigenspace geometry of the operator affects the behavior of the convergence. Moreover, we showed that as the updates get more asynchronous the convergence region for the eigenvalues gets larger. We also demonstrated and discussed how the sign (or, the phase when complex) of an eigenvalue affects the rate of convergence of random component-wise updates. In order to make the updates converge to an arbitrary eigenvector, we considered polynomials of the operator. In particular, we showed that second order polynomials are sufficient to achieve such a convergence. By combining the asynchronous iterations and second order polynomials, we formally presented the node-asynchronous communication protocol. As an application, we used the proposed algorithm to compute the Fiedler vector of a network in order to achieve autonomous clustering. Simulations verified that the algorithm indeed clusters the network successfully. As another application, we reformulated the component-wise power iteration in order to compute the dominant singular vectors of a given data matrix. The proposed approach is proven to converge when computing the rank-1 approximation of a normalized data matrix, and its convergence is verified numerically for an arbitrary rank approximation of an arbitrary data matrix. The proposed algorithm is extended in order to handle large-scale distributed data with distributed asynchronous computation. The convergence of the extended algorithm is verified numerically for various different numbers of data partitions.

2.10 Appendices

2.10.1 A Useful Inequality

Lemma 2.4. *Let* $\mathbf{U} \in \mathbb{C}^{N\times M}$ *and* $\mathbf{X} \succeq \mathbf{0}$. *Then,*

$$\operatorname{tr}\left(\mathbf{U}^{\mathrm{H}}\ \operatorname{diag}(\mathbf{U}\,\mathbf{X}\,\mathbf{U}^{\mathrm{H}})\ \mathbf{U}\right) \ \leq\ \operatorname{tr}(\mathbf{X})\ \lambda_{\max}\left(\mathbf{U}^{\mathrm{H}}\ \operatorname{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\ \mathbf{U}\right), \tag{2.108}$$

and

$$\operatorname{tr}\left(\mathbf{U}^{\mathrm{H}}\ \operatorname{diag}(\mathbf{U}\,\mathbf{X}\,\mathbf{U}^{\mathrm{H}})\ \mathbf{U}\right) \ \geq\ \operatorname{tr}(\mathbf{X})\ \lambda_{\min}\left(\mathbf{U}^{\mathrm{H}}\ \operatorname{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\ \mathbf{U}\right). \tag{2.109}$$

Proof. Consider the following problem

$$\max_{\mathbf{X}\succeq\mathbf{0}}\ \operatorname{tr}\left(\mathbf{U}^{\mathrm{H}}\ \operatorname{diag}(\mathbf{U}\,\mathbf{X}\,\mathbf{U}^{\mathrm{H}})\ \mathbf{U}\right) \qquad \text{s.t.} \qquad \operatorname{tr}(\mathbf{X}) = 1. \tag{2.110}$$

Using the eigenvalue decomposition $\mathbf{X} = \mathbf{V}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V}^{\mathrm{H}}$, and the fact that $\mathbf{X}$ has unit trace, we can write the problem as

$$\max_{\mathbf{V},\,\boldsymbol{\lambda}} \qquad \operatorname{tr}\left(\mathbf{Q}^{\mathrm{H}}\operatorname{diag}(\mathbf{Q}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\right) \qquad \text{s.t.} \qquad \begin{cases} \mathbf{Q} = \mathbf{U}\,\mathbf{V}, \\ \mathbf{V}^{\mathrm{H}}\,\mathbf{V} = \mathbf{I}, \\ \mathbf{1}^{\mathrm{H}}\boldsymbol{\lambda} = 1, \\ \boldsymbol{\lambda} \geq \mathbf{0}. \end{cases} \tag{2.111}$$

Notice that the objective function can be written as

$$\operatorname{tr}\left(\mathbf{Q}^{\mathrm{H}}\operatorname{diag}(\mathbf{Q}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\right) = \mathbf{1}^{\mathrm{H}}\ \mathbf{Z}^{\mathrm{H}}\ \mathbf{Z}\ \boldsymbol{\lambda}, \tag{2.112}$$

where the matrix $\mathbf{Z} \in \mathbb{R}^{N\times M}$ is defined as $Z_{i,j} = |Q_{i,j}|^2$.

Now, consider the problem of maximization over $\boldsymbol{\lambda}$

$$\max_{\boldsymbol{\lambda}} \quad \mathbf{1}^{\mathrm{H}}\ \mathbf{Z}^{\mathrm{H}}\ \mathbf{Z}\ \boldsymbol{\lambda} \qquad \text{s.t.} \qquad \mathbf{1}^{\mathrm{H}}\boldsymbol{\lambda} = 1, \quad \boldsymbol{\lambda} \geq \mathbf{0}. \tag{2.113}$$

Since the vector $\boldsymbol{\lambda}$ is constrained to be nonnegative and sum to 1, the objective function of (2.113) is the convex combination of the elements of the vector $\mathbf{Z}^{\mathrm{H}}\,\mathbf{Z}\,\mathbf{1}$, whose elements are nonnegative as well. Since a convex combination is maximized when the largest element is selected, the solution of (2.113) is $\|\mathbf{Z}^{\mathrm{H}}\mathbf{Z}\,\mathbf{1}\|_\infty$. This can also be seen from the fact that the problem in (2.113) is in the form of the dual-norm formulation of the ℓ_1 norm.

Notice that

$$\begin{aligned}\|\mathbf{Z}^{\mathrm{H}}\mathbf{Z}\,\mathbf{1}\|_\infty &= \max_{1\le i\le M} \left(\mathbf{Q}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{Q}\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\right)_{i,i}, && (2.114)\\ &= \max_{1\le i\le M} \left(\mathbf{V}^{\mathrm{H}}\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\,\mathbf{V}\right)_{i,i}, && (2.115)\\ &= \max_{1\le i\le M} \mathbf{v}_i^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v}_i, && (2.116)\end{aligned}$$

where we use $\mathbf{Q} = \mathbf{U}\mathbf{V}$ from the conditions in (2.111), and the fact that $\mathbf{V}$ is unitary. Hence, the maximization over $\mathbf{V}$ can then be written as

$$\max_{\mathbf{v}_i} \quad \mathbf{v}_i^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v}_i \quad \text{s.t.} \quad \mathbf{v}_i^{\mathrm{H}}\,\mathbf{v}_j = \delta_{i,j}, \tag{2.117}$$

which can be simplified to

$$\max_{\mathbf{v}} \quad \mathbf{v}^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v} \quad \text{s.t.} \quad \|\mathbf{v}\|_2^2 = 1, \tag{2.118}$$

whose solution is simply the largest eigenvalue of the matrix $\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}$. Therefore for any $\mathbf{X} \succeq \mathbf{0}$ we have

$$\mathrm{tr}\left(\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{X}\mathbf{U}^{\mathrm{H}})\mathbf{U}\right) \le \mathrm{tr}(\mathbf{X})\;\lambda_{\max}\left(\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\right). \tag{2.119}$$

For the inequality in (2.109), the maximization in (2.113) is replaced with the minimization. Since the minimum of the objective function is achieved when the minimum element of $\mathbf{Z}^{\mathrm{H}}\,\mathbf{Z}\,\mathbf{1}$ is selected, maximization in (2.114) (hence the one in (2.118)) is replaced with minimization, which yields the minimum eigenvalue problem in (2.118). This validates the inequality in (2.109). □

2.10.2 Proof of Theorem 2.2

Consider the following covariance matrix:

$$\mathbf{C}_k = \mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_k\,\mathbf{x}_k^{\mathrm{H}}\,\mathbf{U}], \tag{2.120}$$

which gives $\mathrm{tr}(\mathbf{C}_k) = \mathbb{E}\left[\,\|\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_k\|_2^2\,\right] = \mathbb{E}[\,\|\mathbf{r}_k\|_2^2\,]$. We can write $\mathbf{C}_k$ as follows:

$$\mathbf{C}_k = \mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{I}\,\mathbf{x}_{k-1}\,\mathbf{x}_{k-1}^{\mathrm{H}}\,\mathbf{I}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}]. \tag{2.121}$$

Notice that we have $\mathbf{I} = \mathbf{V}_1 \mathbf{V}_1^{\mathrm{H}} + \mathbf{U}\mathbf{U}^{\mathrm{H}}$. Furthermore, $\mathbf{Q}\,\mathbf{V}_1 = \mathbf{V}_1$ is valid for *any* outcome of the random matrix $\mathbf{Q}$ in (2.7). We note that $\mathbf{V}_1^{\mathrm{H}}\,\mathbf{Q} \neq \mathbf{V}_1^{\mathrm{H}}$ in general. Further using the fact that $\mathbf{V}_1^{\mathrm{H}}\,\mathbf{U} = \mathbf{0}$, we can write (2.121) as follows:

$$\begin{aligned}
\mathbf{C}_k &= \mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_{k\text{-}1}\,\mathbf{x}_{k\text{-}1}^{\mathrm{H}}\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}], && (2.122)\\
&= \mathbb{E}\big[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_{k\text{-}1}\,\mathbf{x}_{k\text{-}1}^{\mathrm{H}}\,\mathbf{U}]\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}\big], && (2.123)\\
&= \mathbb{E}\big[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbf{C}_{k\text{-}1}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}\big], && (2.124)
\end{aligned}$$

where (2.123) follows from the fact that each update is independent of the previous ones.

In order to compute the expectation in (2.124), in the following we will first find the conditional expectation (conditioned on T) as

$$\bar{\mathbf{C}}_k = \mathbb{E}\Big[\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}\,\mathbf{U}\,\mathbf{C}_{k\text{-}1}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}^{\mathrm{H}}\,\mathbf{U}\;\big|\;\mathsf{T}\,\Big]. \qquad (2.125)$$

Then, $\mathbf{C}_k$ will be found as $\mathbf{C}_k = \mathbb{E}[\,\bar{\mathbf{C}}_k\,]$ where the expectation is with respect to T.

Since $\mathbf{A}$ is assumed to be normal matrix, its eigenspaces are orthonormal to each other. Therefore, the matrix $\mathbf{U}$ corresponds to the eigenvectors of $\mathbf{A}$ with non-unit eigenvalues. Furthermore, the normality implies also that the left and the right eigenvectors of $\mathbf{A}$ are conjugate transpose of each other. That is, $\mathbf{V}_1^{\mathrm{H}}\,(\mathbf{A}-\mathbf{I}) = \mathbf{0}$, and $\mathbf{U}^{\mathrm{H}}\,(\mathbf{A}-\mathbf{I}) = \mathbf{\Sigma}\,\mathbf{U}^{\mathrm{H}}$, where $\mathbf{\Sigma} \in \mathbb{C}^{(N\text{-}M)\times(N\text{-}M)}$ is a diagonal matrix with diagonal entries $\{\sigma_1, \cdots, \sigma_{N\text{-}M}\}$ where $\sigma_i = \lambda_i - 1$. Thus, the normality of $\mathbf{A}$ implies the following equality:

$$\mathbf{U}^{\mathrm{H}}\;\mathbf{Q}(\mathcal{T})\;\mathbf{U} = \mathbf{I} + \mathbf{U}^{\mathrm{H}}\;\mathbf{D}_{\mathcal{T}}\;\mathbf{U}\,\mathbf{\Sigma}. \qquad (2.126)$$

Using (2.126) in (2.125), the expectation can be written as:

$$\begin{aligned}
\bar{\mathbf{C}}_k &= \frac{1}{\binom{N}{\mathsf{T}}}\sum_{\mathcal{T}}\Big(\mathbf{I} + \mathbf{U}^{\mathrm{H}}\,\mathbf{D}_{\mathcal{T}}\,\mathbf{U}\,\mathbf{\Sigma}\Big)\mathbf{C}_{k\text{-}1}\Big(\mathbf{I} + \mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{D}_{\mathcal{T}}\,\mathbf{U}\Big),\\
&= \mathbf{C}_{k\text{-}1} + \frac{\mathsf{T}}{N}\Big(\mathbf{\Sigma}\,\mathbf{C}_{k\text{-}1} + \mathbf{C}_{k\text{-}1}\,\mathbf{\Sigma}^{\mathrm{H}}\Big) + \frac{\mathsf{T}(\mathsf{T}-1)}{N(N-1)}\mathbf{\Sigma}\,\mathbf{C}_{k\text{-}1}\,\mathbf{\Sigma}^{\mathrm{H}}\\
&\quad + \frac{\mathsf{T}(N-\mathsf{T})}{N(N-1)}\;\mathbf{U}^{\mathrm{H}}\;\mathrm{diag}\left(\mathbf{U}\,\mathbf{\Sigma}\,\mathbf{C}_{k\text{-}1}\,\mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\right)\mathbf{U}, \qquad (2.127)
\end{aligned}$$

where the last step follows from (2.1) and (2.2).

By taking the trace of both sides of (2.127) and using (2.108) from Lemma 2.4 we obtain the following:

$$\mathrm{tr}(\bar{\mathbf{C}}_k) \leq \mathrm{tr}\left(\mathbf{C}_{k\text{-}1}\;\left[\mathbf{I} + \frac{\mathsf{T}}{N}(\mathbf{\Sigma} + \mathbf{\Sigma}^{\mathrm{H}} + \mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{\Sigma}) + \frac{\mathsf{T}(N-\mathsf{T})}{N(N-1)}\;(\rho-1)\;\mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{\Sigma}\right]\right), \qquad (2.128)$$

where ρ is defined as in (2.30). By taking the expectation of both sides in (2.128) with respect to T, we get the following:

$$\operatorname{tr}(\mathbf{C}_k) \leq \operatorname{tr}\left(\mathbf{C}_{k\text{-}1}\left[\mathbf{I} + \frac{\mu_{\mathsf{T}}}{N}\left(\boldsymbol{\Sigma} + \boldsymbol{\Sigma}^{\mathrm{H}} + \boldsymbol{\Sigma}^{\mathrm{H}}\boldsymbol{\Sigma}\right) + \frac{\mu_{\mathsf{T}}}{N}\,\delta_{\mathsf{T}}\,(\rho - 1)\,\boldsymbol{\Sigma}^{\mathrm{H}}\boldsymbol{\Sigma}\right]\right), \tag{2.129}$$

$$= \sum_{i=1}^{N-M} (\mathbf{C}_{k\text{-}1})_{i,i}\ \Psi_i\ \leq\ \Psi\ \operatorname{tr}(\mathbf{C}_{k\text{-}1}), \tag{2.130}$$

where Ψ_i's are defined as

$$\Psi_i = 1 + \frac{\mu_{\mathsf{T}}}{N}\left(|\lambda_i|^2 - 1\right) + \frac{\mu_{\mathsf{T}}}{N}\,\delta_{\mathsf{T}}\,(\rho - 1)\,|\lambda_i - 1|^2, \tag{2.131}$$

and Ψ is defined as $\Psi = \max_i \Psi_i$. Iterative use of the inequality in (2.130) gives the upper bound in (2.38).

For the lower bound in (2.38), we consider the trace of both sides of (2.127), use (2.109) from Lemma 2.4, and take the expectation with respect to T. This will provide a lower bound for $\operatorname{tr}(\mathbf{C}_k)$ similar to (2.129) where ρ is replaced with $\bar{\rho}$ from (2.31). Hence, we get

$$\operatorname{tr}(\mathbf{C}_k) \geq \sum_{i=1}^{N-M} (\mathbf{C}_{k\text{-}1})_{i,i}\ \psi_i\ \geq\ \psi\ \operatorname{tr}(\mathbf{C}_{k\text{-}1}), \tag{2.132}$$

where ψ_i's are defined as

$$\psi_i = 1 + \frac{\mu_{\mathsf{T}}}{N}\left(|\lambda_i|^2 - 1\right) + \frac{\mu_{\mathsf{T}}}{N}\,\delta_{\mathsf{T}}\,(\bar{\rho} - 1)\,|\lambda_i - 1|^2, \tag{2.133}$$

and ψ is defied as $\psi = \min_i \psi_i$. Iterative use of the inequality in (2.132) gives the lower bound in (2.38).

2.10.3 A Counter-Example

Consider the following set of N complex numbers: $\lambda_n = e^{j2\pi n/(N\text{-}1)}$ for $1 \leq n \leq N\text{-}1$ and $\lambda_N = 0$. Assume that there exists a polynomial of order $L \leq N\text{-}2$ such that it satisfies the following mapping:

$$\left|h(\lambda_n)\right| < 1 \qquad \text{for} \qquad 1 \leq n \leq N\text{-}1, \tag{2.134}$$

$$h(\lambda_N) = 1. \tag{2.135}$$

When the polynomial is written explicitly as $h(\lambda) = \sum_{k=0}^{L} h_k\,\lambda^k$, the condition in (2.134) can be written as

$$\left|\sum_{k=0}^{L} h_k\ e^{j\,2\,\pi\,n\,k/(N\text{-}1)}\right|^2 = \sum_{k,s=0}^{L} h_k\,h_s^*\ e^{j\,2\,\pi\,n\,(k\text{-}s)/(N\text{-}1)} < 1. \tag{2.136}$$

Notice that the inequality in (2.136) holds true for all n in $1 \leq n \leq N$-1. Therefore, the following also holds true:

$$N - 1 > \sum_{n=1}^{N\text{-}1} \sum_{k,s=0}^{L} h_k\, h_s^*\, e^{j\,2\,\pi\,n\,(k\text{-}s)/(N\text{-}1)} = \sum_{k,s=0}^{L} h_k\, h_s^* \sum_{n=1}^{N\text{-}1} e^{j\,2\,\pi\,n\,(k\text{-}s)/(N\text{-}1)}. \tag{2.137}$$

Note that $0 \leq k, s \leq L \leq N$-2. As a result, the inner summation in (2.137) is nonzero if and only if $k = s$. That is,

$$\sum_{n=1}^{N\text{-}1} e^{j\,2\,\pi\,n\,(k\text{-}s)/(N\text{-}1)} = (N\text{-}1)\ \ \delta_{k,s}, \tag{2.138}$$

where $\delta_{k,s}$ stands for the Dirac delta function. Then, the inequality in (2.137) becomes

$$1 > \sum_{k,s=0}^{L} h_k\, h_s^*\, \delta_{k,s} = \sum_{k=0}^{L} |h_k|^2. \tag{2.139}$$

Notice that the condition in (2.135) implies that $h_0 = 1$. Therefore, (2.139) can be written as

$$1 > \sum_{k=0}^{L} |h_k|^2 = 1 + \sum_{k=1}^{L} |h_k|^2, \tag{2.140}$$

which implies $0 > \sum_{k=1}^{L} |h_k|^2$, which is a contradiction. Hence, no polynomial (possibly with complex coefficients) of order $L \leq N$-2 can satisfy the conditions in (2.134) and (2.135). Therefore, a polynomial of order N-1 is in fact necessary in the complex case in general.

Chapter 3

IIR FILTERING ON GRAPHS WITH RANDOM NODE-ASYNCHRONOUS UPDATES

3.1 Introduction

One important aspect of graph signal processing is the use of graph filters, which can be utilized in order to smooth out graph signals (low-pass filters), or detect anomalies (high-pass filters) [153]. Similar to the classical signal processing, graph filters can be constructed in two different forms: finite impulse response (FIR), or infinite impulse response (IIR). The FIR case corresponds to a matrix polynomial of the given graph operator [151, 160, 172]. It is well-known that a polynomial graph filter of order L is localized on the graph, that is, nodes are required to communicate only with their L-hop neighbors in order to implement the filter. For this reason it is very natural to think of polynomial graph filtering as a way of distributed signal processing, in which the low-order polynomials are favored to keep the communications localized. The papers [158, 149, 150, 154] (and references therein) made explicit connections between polynomial graph filters and distributed computation, and studied various problems including smoothing, regularization, and consensus.

In the IIR case, the graph filter is constructed with respect to a rational function rather than a polynomial. It should be noted that an IIR graph filter can be equivalently represented as an FIR graph filter (possibly with a very high order) due to the finite spectrum of the graph operator. Nevertheless, IIR filters are still useful to consider since they can provide better approximations for a given filter specifications. When extended to the case of graphs, an IIR filter of order L can be implemented via iterative procedures that preserve the locality of the communications. The studies in [157, 87, 88, 108, 109] analyzed the convergence behavior of such filters and showed successful applications on graph signals with distributed processing.

Although both polynomial and rational graph filters can be implemented in a distributed fashion, aforementioned implementations are based on successive graph shifts (multiplication with the graph operator). Although the graph shift can be implemented via data exchange with the neighboring nodes, it requires all the nodes to communicate simultaneously. That is, all the nodes should send and receive data at the same time instance, or nodes should wait until all the communications are terminated before proceeding to the next iteration (shift). Synchronization becomes an important limitation when the size of the network, N, is large, e.g. distributed large-scale graph processing frameworks [71, 163, 51, 10], or the network has autonomous behavior without a centralized control.

In order to eliminate the need for synchronization, this chapter proposes a node-asynchronous implementation of an arbitrary rational filter (including FIR) on an *arbitrary* graph. In the proposed algorithm neighboring nodes send and receive a vector variable (state vector) whose size is determined by the order of the filter, and the nodes follow a collect-compute-broadcast framework. More precisely, the algorithm consists of two main stages: passive and active. In the passive stage, a node receives and stores the data (local state vectors) sent by its incoming neighbors. When a node gets into the active stage at a *random* time instance, it completes the necessary filtering calculations (local state recursions), and then broadcasts its most recent state vector to its outgoing neighbors. Thus, nodes behave asynchronously on the network. By carefully designing the computation scheme, the proposed algorithm is proven to converge to the desired filtered signal in the mean-squared sense under mild stability conditions.

3.1.1 Relations with Asynchronous Fixed Point Iterations

In this chapter, the analysis of the algorithm will be based on the convergence properties of randomized asynchronous linear fixed point iterations (state recursions). We note that non-random asynchronous fixed point iterations are well studied problems in the literature [32, 14, 19, 20], which considered more general non-linear update models. For the linear model (which is the case in this chapter), the earliest analysis can be traced back to the study in [32] that provided the necessary and sufficient condition under which the asynchronous iterations are guaranteed to converge for any index sequence. More recently, studies in [9, 141] (and references therein) studied the randomized variations of asynchronous iterations, in which indices are assumed to be selected with equal probabilities, and they provided sufficiency conditions for the convergence. Asynchronous iterations are considered also in the context of semi-supervised learning on graphs [6, 7].

In the case considered in this chapter, the indices are allowed to be selected with non-equal probabilities during the asynchronous recursions. More importantly, the possibility of updating different number of indices in each iteration (which can be considered as partial synchrony) is also not ruled out. In fact, convergence analysis of a similar setting is studied in Chapter 2 for the case of zero-input, in which the system is assumed to have a unit eigenvalue, and the iterand is proven to converge to a point in the eigenspace of the unit eigenvalue when all the indices are updated with equal probabilities. On the contrary, the model considered here starts with the assumption that the system does *not* have a unit eigenvalue, and it further assumes that the input is a nonzero constant, so there is a unique nonzero fixed point. This chapter also considers the effect of the input noise. For this setting, we prove that the Schur diagonal stability of the system matrix (which is more relaxed than the condition given in [32, 141]) is sufficient for the convergence of the randomized asynchronous iterations in the mean-squared sense.

3.1.2 Outline and Contributions of This Chapter

This chapter consists of two main parts. The first part (Section 3.2) considers the analysis of the randomized asynchronous state recursions in arbitrary linear systems, of which synchronous non-random recursions are special-cases and all results continue to be applicable. The second part (Sections 3.3 and 3.4) focuses on the specific case of graphs and considers a node-asynchronous implementation of rational graph filters. More precisely, in Section 3.2, we introduce the randomized asynchronous model for the state recursions and present the first main result (Theorem 3.1) that provides upper and lower bounds for the mean-squared error of the randomized iterations. Based on this result, we provide a sufficient condition (Corollary 3.1) that ensures the convergence of the iterations. Then, we prove that the presented condition is more relaxed than the well-known necessary condition for the convergence of the non-random asynchronous iterations (Lemma 3.1). The special case of uniform index-selection probabilities is also considered (Corollary 3.2). In Section 3.3, we propose a node-asynchronous implementation of a graph filter (Algorithm 5) and describe its behavior. Then, in Section 3.4 we prove the convergence of the proposed algorithm to the desired filtered signal in the mean-squared sense for both synchronous and asynchronous cases (Theorems 3.2 and 3.3). Finally in Section 3.5, we simulate the proposed algorithm for various different graph filters (including rational and polynomial) and demonstrate the convergence behavior of the algorithm numerically.

We note that results presented in this chapter allow the graph to have directed edges possibly with a non-diagonalizable operator. It is also important to point out that this chapter does not consider the design of graph filters. The main focus here is a node-asynchronous implementation of a *given* graph filter.

The content of this chapter is mainly drawn from [176], and parts of it have been presented in [178, 179].

3.2 Asynchronous State Recursions

Given a matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$ and a constant input signal $\mathbf{u} \in \mathbb{C}^{N}$, we will consider the following type of recursion on the state vector $\mathbf{x}_k \in \mathbb{C}^{N}$:

$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k\text{-}1} + \mathbf{u}_{k\text{-}1}, \tag{3.1}$$

where $\mathbf{x}_0$ denotes the initial value of the state vector, and $\mathbf{u}_k$ denotes the noisy input signal. That is,

$$\mathbf{u}_k = \mathbf{u} + \mathbf{w}_k, \tag{3.2}$$

where $\mathbf{w}_k$ is the noise term with the following statistics:

$$\mathbb{E}[\mathbf{w}_k] = \mathbf{0}, \qquad \mathbb{E}\big[\mathbf{w}_k\ \mathbf{w}_s^{\mathrm{H}}\big] = \delta(k-s)\ \mathbf{\Gamma}, \tag{3.3}$$

where $\delta(\cdot)$ denotes the discrete Dirac delta function, and $\mathbf{\Gamma}$ is allowed to be non-diagonal.

In the noise-free case, i.e., $\mathbf{\Gamma} = \mathbf{0}$, the fixed point of the recursion in (3.1) is given as follows:

$$\mathbf{x}^{\star} = (\mathbf{I} - \mathbf{A})^{\text{-}1}\ \mathbf{u}, \tag{3.4}$$

which requires $\mathbf{A}$ *not* to have eigenvalue 1 so that $\mathbf{I} - \mathbf{A}$ is invertible. In order to analyze the convergence behavior, we first define the residual (error) vector as follows:

$$\mathbf{r}_k = \mathbf{x}_k - \mathbf{x}^{\star}. \tag{3.5}$$

By substituting (3.5) into the state recursion in (3.1), the residual $\mathbf{r}_k$ can be written explicitly as follows:

$$\mathbf{r}_k = \mathbf{A}^k\ \mathbf{r}_0 + \sum_{n=0}^{k-1} \mathbf{A}^n\ \mathbf{w}_{k\text{-}1\text{-}n}. \tag{3.6}$$

Due to the fact that $\mathbf{w}_k$'s are uncorrelated in different iterations and have zero-mean, the expected squared ℓ_2-norm of the residual $\mathbf{r}_k$ can be written as follows:

$$\mathbb{E}\big[\|\mathbf{r}_k\|_2^2\big] = \big\|\mathbf{A}^k\ \mathbf{r}_0\big\|_2^2 + \mathrm{tr}\left(\sum_{n=0}^{k-1} \mathbf{A}^n\ \mathbf{\Gamma}\ (\mathbf{A}^n)^{\mathrm{H}}\right). \tag{3.7}$$

It is clear from (3.7) that when $\mathbf{A}$ is a stable matrix, i.e., when the following holds true:

$$\rho(\mathbf{A}) < 1, \tag{3.8}$$

the error term in (3.7) approaches an error floor. More precisely,

$$\lim_{k\to\infty} \mathbb{E}[\mathbf{r}_k] = \mathbf{0}, \qquad \text{and} \qquad \lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{r}_k\|_2^2\big] = \mathrm{tr}(\boldsymbol{\Upsilon}\,\boldsymbol{\Gamma}), \tag{3.9}$$

where $\boldsymbol{\Upsilon}$ is given as follows:

$$\boldsymbol{\Upsilon} = \mathbf{I} + \sum_{n=1}^{\infty} (\mathbf{A}^n)^{\mathrm{H}}\, \mathbf{A}^n, \tag{3.10}$$

which converges due to (3.8). (See [95, Appendix D].)

In the noise-free case ($\boldsymbol{\Gamma} = \mathbf{0}$), the limit in (3.9) implies the convergence of $\mathbf{x}_k$ to $\mathbf{x}^{\star}$. On the other hand, in the case of an unstable transition matrix $\mathbf{A}$, i.e., $\rho(\mathbf{A}) \geq 1$, the mean-squared error is bounded away from zero even in the noise-free case. Therefore, the condition in (3.8) is both sufficient and necessary for the convergence of the state recursions in (3.1). This is, in fact, a well-known result from the linear system theory [95].

In the context of graph signal processing [152, 153, 151, 160], the matrix $\mathbf{A}$ is assumed to be a local graph operator (shift matrix) on the graph of interest. Thus, an iteration in the form of (3.1) can be implemented on the graph as a data exchange between the neighboring nodes. That is, (3.1) can be written as follows:

$$(\mathbf{x}_k)_i = \sum_j A_{i,j}\ (\mathbf{x}_{k\text{-}1})_j + (\mathbf{u}_{k\text{-}1})_i, \tag{3.11}$$

for all nodes i in $1 \leq i \leq N$. In this setting, $\mathbf{u}$ is considered as a signal defined on the graph, where the nodes will be the “domain" analogous to time. The index k will denote the round of communication, so the graph signal $\mathbf{u}$ does not depend on the iteration index k. Note that the noisy measurement $\mathbf{u}_k = \mathbf{u} + \mathbf{w}_k$ depends on k.

Although the individual nodes can perform the updates of (3.11) locally, such an implementation requires a synchronization mechanism among the nodes. That is, all the nodes should send and receive data at the same time instance, or nodes should wait until all the communications are terminated before proceeding to the next iteration. Synchronization becomes an important limitation when the size of the network, N, is large, or the network has autonomous behavior, in which case there is no centralized control over the network.

In order to overcome the need for synchronization, in this chapter we will consider a randomized asynchronous variation of the state recursion in (3.1), in which only a random subset of indices are updated simultaneously and the remaining ones stay unchanged. More precisely, we consider the following update model:

$$(\mathbf{x}_k)_i = \begin{cases} (\mathbf{A}\,\mathbf{x}_{k\text{-}1})_i + (\mathbf{u}_{k\text{-}1})_i, & i \in \mathcal{T}_k, \\ (\mathbf{x}_{k\text{-}1})_i, & i \notin \mathcal{T}_k, \end{cases} \tag{3.12}$$

where $\mathcal{T}_k$ denotes the set of indices updated at the k^{th} iteration.

For non-random variants of the model in (3.12), the study [32] assumed that only one index is updated per iteration and allowed the use of the past values of the iterant, that is, $\mathbf{x}_k$ may depend on $\{\mathbf{x}_{k\text{-}1}, \cdots, \mathbf{x}_{k\text{-}s}\}$ for some fixed s. Thus, a noise-free and non-random version of (3.12) with $|\mathcal{T}_k| = 1$ corresponds to the model considered in [32] with $s = 1$, for which the following condition is shown to be both necessary and sufficient for the convergence of the iterations (see [32, Section 5] and [14, Section 3.2]):

$$\rho(|\mathbf{A}|) < 1. \tag{3.13}$$

In words, if (3.13) is satisfied, the iterations converge for any index sequence in which no index is left out. On the contrary, if (3.13) is violated, then *there exists* an index sequence for which the iterations do not convergence.

The case of randomized index-selection was also studied more recently in [141, 9] for the solution of N linear equations with N unknowns. These studies focused also on the case of $|\mathcal{T}_k| = 1$ (updating only one index per iteration) and showed that the following condition:

$$\|\mathbf{A}\|_2 < 1, \tag{3.14}$$

is sufficient (with some additional assumptions on $\mathbf{A}$) to ensure the convergence of the iterations. We refer to [9, Lemma 3.1] and [141, Section 2.1] for the precise details.

In the randomized asynchronous model considered in this chapter, we allow the case of updating more than one index per iteration possibly with indices having non-uniform selection probabilities. In the next subsection, we will elaborate on the statistical properties of the index-selection model and define the average index-selection matrix, which will play an important role in the convergence analysis of the iterations.

3.2.1 Random Selection of the Update Sets

In the asynchronous model we consider in (3.12), the update set $\mathcal{T}_k$ is assumed to be selected randomly and independently among all possible 2^N different subsets of $\{1, \cdots, N\}$ in every iteration of (3.12). However, we would like to emphasize that the independent selection of the update sets *do not* necessarily imply independent selection of the indices. Thus, the model considered here allows correlated index-selection schemes. We also note that both the content and the size of $\mathcal{T}_k$ are random variables. We do not assume that $\mathcal{T}_k$'s have identical distributions at every iteration k. Nevertheless, we do assume that the distribution of $\mathcal{T}_k$ is first-order stationary in the following sense: expectation of the index-selection matrix $\mathbf{D}_{\mathcal{T}_k}$ does not depend on k. More precisely,

$$\mathbb{E}\big[\mathbf{D}_{\mathcal{T}_k}\big] = \mathbf{P} \qquad \forall\, k. \tag{3.15}$$

In the rest of the chapter, the matrix $\mathbf{P} \in \mathbb{R}^{N\times N}$ will be referred to as the *average index (node) selection matrix*, which is a deterministic and diagonal matrix satisfying the following:

$$\mathbf{0} \prec \mathbf{P} \preceq \mathbf{I}, \tag{3.16}$$

where the positive definiteness follows from the fact that no index is left out (on average) in the update scheme of (3.12). We also note that $\mathrm{tr}(\mathbf{P}) = \mathbb{E}[\,|\mathcal{T}_k|\,]$ corresponds to the average number of indices updated per iteration.

3.2.2 Convergence in the Mean-Squared Sense

It is easily verified that the fixed point of the randomized model (3.12) continues to be $\mathbf{x}^\star$ given in (3.4). Therefore, the vector $\mathbf{r}_k$ defined in (3.5) represents the residual for the randomized asynchronous model as well. Thus, the convergence of $\mathbf{r}_k$ to zero implies the convergence of $\mathbf{x}_k$ to the fixed point $\mathbf{x}^\star$. However, $\mathbf{r}_k$ is a random variable in the asynchronous case due to the random selection of the indices. The following theorem, whose proof is presented in Section 3.7.1, provides bounds on the mean-squared error as follows:

Theorem 3.1. *In the randomized asynchronous model* (3.12)*, the mean-squared error can be bounded as follows:*

$$\psi^k\,\|\mathbf{r}_0\|_2^2 + \frac{1-\psi^k}{1-\psi}\,\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma}) \;\le\; \mathbb{E}\big[\|\mathbf{r}_k\|_2^2\big] \;\le\; \Psi^k\,\|\mathbf{r}_0\|_2^2 + \frac{1-\Psi^k}{1-\Psi}\,\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma}), \tag{3.17}$$

where

$$\psi = \lambda_{\min}\big(\mathbf{I} + \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P}\big), \qquad \Psi = \lambda_{\max}\big(\mathbf{I} + \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P}\big). \tag{3.18}$$

Regarding the bounds in (3.17) we first note that the inequality in (3.16) implies $\psi \geq 0$, hence $\Psi \geq 0$, irrespective of the values of $\mathbf{A}$ and $\mathbf{P}$. As a result the expressions on both sides of (3.17) are positive and finite. However, Theorem 3.1 by itself does not ensure the convergence of the iterations as the values of ψ and Ψ can be larger than or equal to 1 for some values of $\mathbf{A}$ and $\mathbf{P}$. The following corollary presents a *sufficiency* condition that ensures the convergence of the randomized iterations of (3.12) in the mean-squared sense up to an error floor depending on the amount of input noise:

Corollary 3.1. *If the state transition matrix* $\mathbf{A}$ *and the average index-selection matrix* $\mathbf{P}$ *satisfy the following:*

$$\mathbf{A}^{\mathrm{H}}\, \mathbf{P}\, \mathbf{A} \prec \mathbf{P}, \tag{3.19}$$

then, the limit of the mean squared error of the asynchronous model in (3.12) *is bounded as follows:*

$$\frac{\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma})}{\lambda_{\max}\big(\mathbf{P}-\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A}\big)} \;\leq\; \lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{r}_k\|_2^2\big] \;\leq\; \frac{\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma})}{\lambda_{\min}\big(\mathbf{P}-\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A}\big)}. \tag{3.20}$$

Proof. The assumption (3.19) implies that ψ and Ψ defined in (3.18) satisfy the inequality $0 \leq \psi \leq \Psi < 1$. Then, the bounds in (3.20) follow directly from Theorem 3.1. □

A number of remarks are in order:

1) *Update probabilities and convergence:* The convergence of the iterations depends on the matrix $\mathbf{A}$ as well as the average index-selection matrix $\mathbf{P}$. Thus, the random asynchronous iterations running on a given matrix $\mathbf{A}$ may not converge for an arbitrary set of update probabilities, yet the convergence can still be achieved for specific sets of probabilities. The question of whether *there exists* a $\mathbf{P}$ satisfying (3.19) or not for a given $\mathbf{A}$ will be discussed in the next section.

2) *Error floor:* The lower bound in (3.20) reveals an error floor: no matter how many iterations are used, the expected residual error is always bounded away from zero in the presence of noise ($\mathbf{\Gamma} \neq \mathbf{0}$), which is also the case in synchronous iterations as seen in (3.9). Nevertheless, (3.20) shows that the error floor is bounded linearly by the noise covariance matrix.

3) *Convergence rate and index selection:* It should be noted from Theorem 3.1 that the rate of convergence as well as the error floor depend on the average index-selection matrix $\mathbf{P}$. That is to say, some set of index-selection probabilities may

yield a faster rate of convergence or a lower error floor, for which we provide a numerical evidence in Section 3.5 (See Figures 3.5, 3.8). However, their theoretical analysis will be considered in a later study.

4) *Sufficiency:* It is important to emphasize that the condition (3.19) is only sufficient but not necessary to ensure the convergence of the randomized asynchronous iterations. When (3.19) does not hold true, it merely means that the upper bound dictated by Theorem 3.1 diverges in the limit, which makes the theorem inconclusive regarding the convergence. The non-necessity of the condition (3.19) will be numerically verified later in Section 3.5.3. Nevertheless, the importance of the sufficient condition (3.19) follows from the fact that it does not have any additional assumption on the matrix $\mathbf{A}$: it may have complex values, may be non-Hermitian, and it may even be non-diagonalizable. When the graph operators are considered in Sections 3.3 and 3.4, this will be very important to ensure the convergence of filters on *an arbitrary directed graph.*

In the following Sections 3.2.3 and 3.2.4, we will elaborate on the condition (3.19) as well as the implications of Corollary 3.1. If desired, the reader can skip these two subsections and jump to Section 3.3 directly, where we present an asynchronous implementation of IIR graph filters.

3.2.3 On The Schur Diagonal Stability

In addition to the convergence results presented here for the randomized asynchronous state recursions, the mathematical condition (3.19) appears in various different contexts. For example, an implementation of a digital filter is guaranteed to be free from limit cycles (overflow oscillation) when the transition matrix $\mathbf{A}$ of its realization satisfies (3.19) for some $\mathbf{P}$ [120, 197]. (In fact, [197] requires $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \preceq \mathbf{P}$ only.) Moreover, the study in [100] showed that a condition in the form of (3.19) is sufficient to ensure the convergence of time-varying block asynchronous iterations.

Due to its importance in various different application, the condition (3.19) and its variations have been studied extensively in the literature. In fact, the condition was first referred to as diagonal stability in [21]. Later in [101], the term was revised as Schur diagonal stability in order to distinguish the discrete and the continuous counterparts. (See [101, Definitions 2.1.3 and 2.5.2].) More precisely:

Definition 3.1. *A matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$ is said to be Schur diagonally stable (alternatively, $\mathbf{A} \in \mathcal{D}_{\mathrm{d}}$) if and only if there exists a positive diagonal matrix $\mathbf{P}$ such that*

$\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P} \prec \mathbf{0}$.

Unlike the stability condition (3.8) that depends only on the eigenvalues of a matrix, the Schur diagonal stability of a matrix cannot be decided just by its eigenvalues in the sense that among two *similar* matrices one may be Schur diagonally stable and the other may not [197, 101]. Furthermore, Schur diagonal stability is more restrictive than stability, but more relaxed than (3.13) as shown by the following lemma (whose proof is provided in Section 3.7.2):

Lemma 3.1. *The following hold true for any* $\mathbf{A} \in \mathbb{C}^{N\times N}$*:*

$$\rho(|\mathbf{A}|) < 1 \quad \Longrightarrow \quad \mathbf{A} \in \mathcal{D}_{\mathrm{d}} \quad \Longrightarrow \quad \rho(\mathbf{A}) < 1. \tag{3.21}$$

Furthermore,

$$\|\mathbf{A}\|_2 < 1 \quad \Longrightarrow \quad \mathbf{A} \in \mathcal{D}_{\mathrm{d}}. \tag{3.22}$$

We also note that the converse of the implications in (3.21) and (3.22) do not hold in general. We refer to [101, Section 2] (and references therein) for an elaborate compilation of properties of the diagonal stability.

Two remarks are in order:

1) *Random vs Non-random iterations:* We would like to point out that Corollary 3.1 together with Lemma 3.1 does not contradict the well-known result of [32] that showed the necessity of the condition $\rho(|\mathbf{A}|) < 1$ for the convergence of non-random asynchronous iterations. The key difference between Corollary 3.1 and [32] is the notion of convergence. The study [32] ensures the convergence of the iterations for *any index sequence*, whereas Corollary 3.1 considers the convergence in the mean-squared sense. When $\rho(|\mathbf{A}|) \geq 1$, there exists an index sequence for which iterations do not converge, yet the iterations do converge in the mean-squared sense if the indices can be updated with appropriate probabilities, i.e., the condition (3.19) of Corollary 3.1 is satisfied.

2) *Numerical search:* Schur diagonal stability of a given matrix can be verified via the following semi-definite program:

$$\min_{c,\,\mathbf{p}} \quad c \qquad \text{s.t.} \qquad \begin{array}{c} c\,\mathbf{I} \succeq \mathbf{A}^{\mathrm{H}}\ \mathrm{diag}(\mathbf{p})\,\mathbf{A} - \mathrm{diag}(\mathbf{p}), \\ \mathbf{1} \geq \mathbf{p} \geq \mathbf{0}, \end{array} \tag{3.23}$$

where $\mathbf{1}$ denotes the vector with all ones. More precisely, it can be shown that the optimal value of (3.23) satisfies $c^{\star} < 0$ if and only if the matrix $\mathbf{A}$ is Schur diagonally stable. Thus, the strict negativity of the numerical solution of (3.23) for a given matrix $\mathbf{A}$ determines the Schur diagonal stability of $\mathbf{A}$.

3.2.4 The Case of Uniform Probabilities

The sufficiency condition given by Corollary 3.1 involves both the matrix $\mathbf{A}$ and the average index-selection matrix $\mathbf{P}$. In many practical scenarios, the indices (or the update sets) are selected with uniform probabilities, in which case implications of Theorem 3.1 can be simplified further as we discuss next.

When the indices are equally likely to be updated in each iteration of (3.12), the average index-selection matrix becomes a scaled identity matrix. More precisely,

$$\mathbf{P} = p\,\mathbf{I}, \qquad \text{where} \qquad 0 < p \leq 1. \tag{3.24}$$

In general, it is possible to use different stochastic models for the selection of the update sets whose average index-selection matrix is in the form of (3.24). For example, when a subset of size T is selected uniformly randomly among all possible $\binom{N}{T}$ different subsets, the average index-selection matrix becomes $\mathbf{P} = (T/N)\,\mathbf{I}$. Notice that the case of $T = 1$ corresponds to the selection of only one index uniformly randomly per iteration. It is also possible to select subsets of different sizes, which is considered in Chapter 2 of this thesis (See Section 2.2.1).

When the matrix $\mathbf{P}$ has the form in (3.24), the rate parameters in (3.18) given by Theorem 3.1 reduce to the following form:

$$\psi = p\,\sigma^2_{\min}(\mathbf{A}) + 1 - p, \qquad \Psi = p\,\sigma^2_{\max}(\mathbf{A}) + 1 - p, \tag{3.25}$$

which shows that the singular values of the matrix $\mathbf{A}$ bound the rate of convergence of the iterations of (3.12). As a result, the matrix $\mathbf{A}$ having a bounded spectral norm is sufficient to ensure the convergence of the randomized asynchronous iterations, which is formally presented in the following corollary:

Corollary 3.2. *If* $\|\mathbf{A}\|_2 < 1$ *and the indices are updated with equal probabilities in the random asynchronous model of* (3.12)*, then the limit of the mean squared error is bounded as follows:*

$$\frac{\mathrm{tr}(\mathbf{\Gamma})}{1 - \sigma^2_{\min}(\mathbf{A})} \leq \lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{r}_k\|_2^2\big] \leq \frac{\mathrm{tr}(\mathbf{\Gamma})}{1 - \sigma^2_{\max}(\mathbf{A})}. \tag{3.26}$$

Proof. If the indices are updated with equal probabilities, the average index-selection matrix $\mathbf{P}$ is in the form of (3.24), thus the condition (3.19) of Corollary 3.1 reduces to $\mathbf{A}^{\mathrm{H}}\,\mathbf{A} \prec \mathbf{I}$, which is readily satisfied due to the assumption $\|\mathbf{A}\|_2 < 1$. Then, we can apply Corollary 3.1. The use of (3.25) in Theorem 3.1 gives the bounds in (3.26). $\square$

Some remarks are in order:

1) *Convergence irrespective of the update probability:* Unlike the condition presented by Corollary 3.1, when the indices are updated with equal probabilities, the sufficiency condition given by Corollary 3.2 involves only the matrix $\mathbf{A}$. Therefore, if the condition (bounded spectral norm) is met, the convergence is ensured irrespective of the actual value of the average index-selection matrix $\mathbf{P}$. However, the rate of convergence does depend on $\mathbf{P}$ in general as suggested by (3.25), which will be verified numerically as well in Section 3.5.

2) *Noise amplification:* When the indices are equally likely to be selected in the random asynchronous iterations, there is an amplification to the input noise. This observation follows simply from the assumption $\|\mathbf{A}\|_2 < 1$ that implies $1/(1 - \sigma^2_{\min}(\mathbf{A})) \geq 1$. Thus, the lower bound in (3.26) can be further lower bounded with $\mathrm{tr}(\mathbf{\Gamma}) = \mathbb{E}[\|\mathbf{w}_k\|_2^2]$, which shows that the error floor is always larger than the amount of input noise. This behavior of the random asynchronous iterations is consistent with the synchronous counterpart. The error floor of the synchronous iterations given in (3.9) can be lower bounded as $\mathrm{tr}(\mathbf{\Upsilon}\,\mathbf{\Gamma}) \geq \mathrm{tr}(\mathbf{\Gamma})$ since the matrix $\mathbf{\Upsilon}$ in (3.10) satisfies $\mathbf{\Upsilon} \geq \mathbf{I}$.

3) *Nonstationary noise covariance:* We note that the input noise need not have a stationary distribution for Theorem 3.1, Corollary 3.1, and Corollary 3.2 to be valid. As long as the noise covariance matrix is upper bounded as $\mathbb{E}[\mathbf{w}_k\, \mathbf{w}_k^{\mathrm{H}}] \leq \mathbf{\Gamma}$ for all k, the corresponding upper bounds remain valid. Similarly, the corresponding lower bounds are valid as long as the covariance matrix is lower bounded as $\mathbb{E}[\mathbf{w}_k\, \mathbf{w}_k^{\mathrm{H}}] \geq \mathbf{\Gamma}$ for all k.

3.3 Asynchronous Rational Filters on Graphs

In this section, we will consider a node-asynchronous implementation of a rational graph filter that is specified as follows:

$$h(x) = p(x)\,/\,q(x), \tag{3.27}$$

where the polynomials $p(x)$ and $q(x)$ are of degree (at most) L, and they are assumed to be in the following form:

$$p(x) = \sum_{n=0}^{L} p_n\, x^n, \qquad q(x) = 1 + \sum_{n=1}^{L} q_n\, x^n. \tag{3.28}$$

The coefficients are allowed to be complex in general, i.e., $p_n, q_n \in \mathbb{C}$. In particular, polynomial graph filters, which corresponds to the case of $q_1 = \cdots = q_L = 0$, are not excluded.

3.3.1 Rational Graph Filters

In the following we will use $\mathbf{G} \in \mathbb{C}^{N\times N}$ to denote a graph operator for the graph with N nodes. Here $G_{i,j}$ denotes the weight of the edge from node j to node i. In particular, $G_{i,j} = 0$ when nodes i and j are not neighbors. Examples of such local graph operators include the adjacency matrix, the graph Laplacian, etc. *The graph is allowed to be directed possibly with a non-diagonalizable adjacency matrix.*

For a given graph operator $\mathbf{G} \in \mathbb{C}^{N\times N}$, the rational graph filter corresponding to (3.27) has the following form:

$$h(\mathbf{G}) = p(\mathbf{G})\; q(\mathbf{G})^{-1}, \tag{3.29}$$

where we implicitly assume that $q(\mathbf{G})$ is an invertible matrix.

When $\mathbf{u} \in \mathbb{C}^N$ is a signal on the graph, we will use $\widetilde{\mathbf{u}}$ to denote the filtered version of $\mathbf{u}$ with the filter $h(\mathbf{G})$. That is,

$$\widetilde{\mathbf{u}} = h(\mathbf{G})\, \mathbf{u}, \tag{3.30}$$

where $\mathbf{u}$ is the given signal on the graph.

A special case of rational graph filtering corresponds to Laplacian smoothing [158, 214, 8]. More precisely, given an undirected graph with the Laplacian matrix $\mathbf{L}$ and a signal $\mathbf{u}$ on the graph, the Laplacian smoothing is obtained as the solution of the following regularized least-squares problem:

$$\widetilde{\mathbf{u}} = \arg\min_{\boldsymbol{\xi}} \|\mathbf{u} - \boldsymbol{\xi}\|_2^2 + \gamma\, \boldsymbol{\xi}^{\mathrm{H}}\, \mathbf{L}\, \boldsymbol{\xi}, \qquad \gamma \geq 0, \tag{3.31}$$

whose closed form solution can be obtained as follows:

$$\widetilde{\mathbf{u}} = h(\mathbf{L})\, \mathbf{u}, \qquad \text{where} \qquad h(x) = 1\,/\,(1 + \gamma\, x). \tag{3.32}$$

Thus, a rational graph filter can be considered as an extension to the Laplacian smoothing, in which the filter can have an arbitrary response in the graph frequency rather than (3.32) [158].

3.3.2 Node-Asynchronous Implementation

Unlike the classical digital filters, a rational graph filter can be represented as an FIR (polynomial) graph filter of order at most N-1. (See [86, Theorem 6.2.9].) Thus, one way to implement (3.30) is to compute N-1 graph shifts and take an appropriate linear combination. However, for large graphs (N is large) this is not practical because of its complexity. Furthermore, as discussed in Chapter 2, the graph shift (multiplication with $\mathbf{G}$) forces all nodes to communicate at the same time, which requires a synchronization among the nodes of the network. In a large network synchronization introduces delays, or it may not be even possible in the case of autonomous networks. In order to overcome this limitation, this section will introduce a randomized node-asynchronous implementation of the rational graph filtering in (3.30).

In the proposed implementation, the i^{th} node is assumed to have the following four local variables:

- an input signal: $u_i \in \mathbb{C}$,
- a state-vector: $\mathbf{x}_i \in \mathbb{C}^L$,
- an output variable: $y_i \in \mathbb{C}$,
- a buffer of size $L\,|\mathcal{N}_{\text{in}}(i)|$,

where only the input signal u_i is constant, and the value of the remaining quantities are changing over time in a random manner. In fact, the output variable y_i will be proven to converge to the corresponding element of the filtered signal $\widetilde{u}_i$ in the mean square sense in the proposed approach under some (realistic and practical) conditions on the filter, the graph operator, and the update statistics. (See Theorem 3.3.)

An overview of our approach (which is illustrated in Figure 3.1) is as follows: while a node is not doing updates, it stays in the “passive" stage in which it only receives and stores the state vectors in its buffer sent by its incoming neighbors. See Figure 3.1b. When the node i “wakes up" at a random time instance (asynchronously with respect to other nodes), it follows a two-step update procedure (see Figure 3.1c):

1. Graph shift step: using the values held in its buffer, the state vector $\mathbf{x}_i$ is updated based on its neighbors according to the graph operator $\mathbf{G}$.

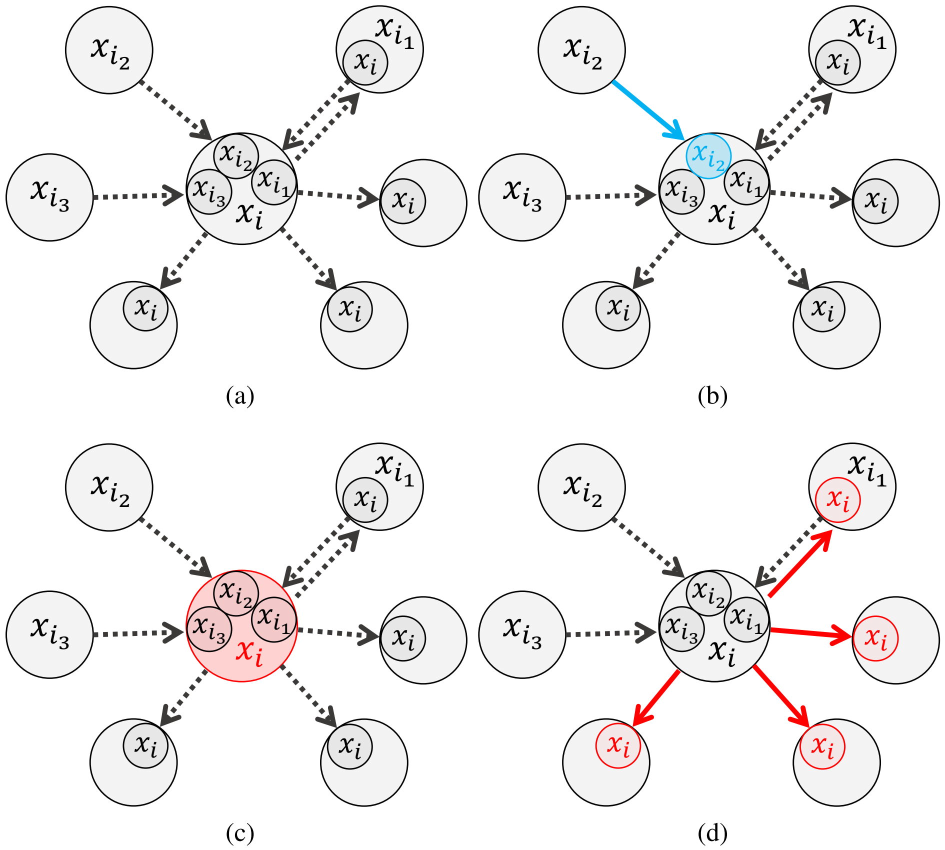


Figure 3.1: Visual illustration of the proposed asynchronous implementation of a given graph filter. Edges can be directed in the network. (a) The node i waits and listens in the passive stage. (b) When the node i receives a message, it updates its buffer. (c) When the node i gets into the active stage at a random time instance, it first updates its state vector. (d) After the update, the node i broadcasts its state vector to its outgoing neighbors.

2. Filtering step: the state vector $\mathbf{x}_i$ is updated once more using the input signal and state recursions imposed by the underlying graph filter in (3.29).

Once the graph filtering stage is completed, the node i broadcasts its most recent state vector $\mathbf{x}_i$ to its outgoing neighbors, who can use its value to update themselves at random asynchronous times in future, in a similar manner. See Figure 3.1d. In the mean time, the local output variable y_i also gets updated using the state vector $\mathbf{x}_i$ and the input signal u_i.

3.3.3 Implementation Details

In this section, we will present the precise details of the proposed asynchronous update mechanism, which was outlined in the previously section. Then, we will present the proposed method formally in Algorithm 5.

We first consider the graph shift step. In order to incorporate the underlying graph structure into the filtering operation, the graph shift step updates the local state vector as *the linear combination of the state vectors of the incoming neighbors*. More precisely, when the i^{th} node is updating its state vector, the node does the following computation first:

$$\mathbf{x}'_i \leftarrow \sum_{j \in \mathcal{N}_{\text{in}}(i)} G_{i,j}\ \mathbf{x}_j, \tag{3.33}$$

where $\mathbf{x}'_i \in \mathbb{C}^L$ denotes the "graph shifted version" of the state vector $\mathbf{x}_i$. It is important to note that the computation in (3.33) can be done locally and *asynchronously* by the i^{th} node, as the node is assumed to have all the state vectors of its incoming neighbors already available in its buffer.

In the filtering step, we use the graph shifted state vector $\mathbf{x}'_i$ to carry out a state recursion corresponding to the underlying filter. In this regard, consider the scalar IIR digital filter, $h_{\text{d}}(z)$, whose transfer function is given as follows:

$$h_{\text{d}}(z) = p(z^{-1}) \,/\, q(z^{-1}) = \sum_{n=0}^{\infty} h_n\ z^{-n}, \tag{3.34}$$

where $p(z)$ and $q(z)$ are as in (3.28), and h_n's correspond to the coefficients of the impulse response of the digital filter. Furthermore, we assume that the digital filter (3.34) has the following state-space description:

$$\mathbf{A} = \mathbf{T}^{-1}\ \widehat{\mathbf{A}}\ \mathbf{T}, \qquad \mathbf{b} = \mathbf{T}^{-1}\ \widehat{\mathbf{b}}, \qquad \mathbf{c} = \widehat{\mathbf{c}}\ \mathbf{T}, \qquad d = \widehat{d}, \tag{3.35}$$

where the quadruple $(\widehat{\mathbf{A}}, \widehat{\mathbf{b}}, \widehat{\mathbf{c}}, \widehat{d})$ corresponds to the direct form description of the filter in (3.34) (see [200, Section 13.4]):

$$\widehat{\mathbf{A}} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -q_L & -q_{L-1} & \cdots & \cdots & -q_1 \end{bmatrix}, \qquad \widehat{\mathbf{b}} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \qquad \widehat{d} = p_0,$$

$$\widehat{\mathbf{c}} = \begin{bmatrix} p_L - p_0\, q_L & p_{L-1} - p_0\, q_{L-1} & \cdots & p_1 - p_0\, q_1 \end{bmatrix}, \tag{3.36}$$

and $\mathbf{T} \in \mathbb{C}^{L\times L}$ is an arbitrary *invertible* matrix.

Although the response of the filter in (3.34) does not depend on the particular selection of the matrix $\mathbf{T}$, the convergence properties of the node-asynchronous implementation of the filter on the graph does depend on the similarity transformation. We will elaborate on this in Section 3.4.1, but the optimal choice of $\mathbf{T}$ is not known at this time.

Using the state-space description of the underlying filter in (3.35), the i^{th} node executes the following updates locally in the filtering step:

$$\begin{aligned} y_i &\leftarrow \mathbf{c}\,\mathbf{x}'_i + d\,(u_i + w_i), \\ \mathbf{x}_i &\leftarrow \mathbf{A}\,\mathbf{x}'_i + \mathbf{b}\,(u_i + w_i), \end{aligned} \tag{3.37}$$

where w_i denotes the additive input noise measured by the node i during an update. We note that the value of u_i remains the same, but the value of w_i is different in each update due to it being random. If the nodes do not take measurements, one can easily assume that the measurements are noise-free and set $w_i = 0$ so that the noise covariance is $\mathbf{\Gamma} = \mathbf{0}$.

The random node-asynchronous implementation of the IIR graph filter (3.29) is summarized in Algorithm 5. In the next section we will prove that this algorithm is indeed a valid implementation of (3.29) under some conditions to be stated.

Algorithm 5 Node-Asynchronous Rational Graph Filtering

```
1: procedure INITIALIZATION($i$)
2:     Initialize the state vector $\mathbf{x}_i \in \mathbb{C}^L$ as $\mathbf{x}_i = \mathbf{0}$.
3: procedure PASSIVE STAGE($i$)
4:     if $\mathbf{x}_j$ is received from the node $j \in \mathcal{N}_{\text{in}}(i)$ then
5:         Store the most recent value of $\mathbf{x}_j$.
6: procedure ACTIVE STAGE($i$)
7:     $\mathbf{x}'_i \leftarrow \sum_{j\in\mathcal{N}_{\text{in}}(i)} G_{i,j}\ \mathbf{x}_j$.          ▹ graph shift
8:     $v_i \leftarrow u_i + w_i$.                                                  ▹ noisy sample
9:     $y_i \leftarrow \mathbf{c}\,\mathbf{x}'_i + d\,v_i$.                          ▹ filtering
10:    $\mathbf{x}_i \leftarrow \mathbf{A}\,\mathbf{x}'_i + \mathbf{b}\,v_i$.        ▹ filtering
11:    Broadcast $\mathbf{x}_i$ to all $j \in \mathcal{N}_{\text{out}}(i)$.
```

Except the initialization stage, which is executed only once, Algorithm 5 consists of two main states: passive and active, both of which are triggered asynchronously. More precisely, the active stage is triggered randomly by a node-specific timer, or condition, and the passive stage is triggered when a node receives a state vector from its incoming neighbors. It is also assumed that a node stores only the most recent received data.

When the node i gets into the active stage at a random time instance, the node first computes its graph shifted state vector $\mathbf{x}_i'$ in Line 7 using the values that are available in its buffer. Then, the node takes a noisy measurement of the underlying graph signal u_i. When the filtering recursions in Lines 9 and 10 are completed, the node broadcasts its own local state vector to its outgoing neighbors, and gets back into the passive stage.

In the presented algorithm we emphasize that a node getting into the active stage is independent of the values in its buffer. In general, a node does *not* wait until it receives state vectors from all of its neighbors. In between two activations (i.e., in the passive stage), some values in the buffer may be updated more than once, and some may not be updated at all. Nodes use the most recent update only.

Since nodes are assumed to store the most recent data of its incoming neighbors in the presented form of the algorithm, the node i requires a buffer of size $L \cdot |\mathcal{N}_{\text{in}}(i)|$. In fact, Algorithm 5 can be implemented in such a way that each node uses a buffer of size $2L$ only. One can show that this is achieved when each node broadcasts the difference in its state vector rather than the state vector itself, and the variable $\mathbf{x}_i'$ accumulates all the received differences in the passive stage. However, such an implementation may not be robust under communication failures, whereas the current form of the algorithm is shown to be robust to communication failures. (See Section 3.5.2.) Moreover, the current form of the algorithm is easier to model and analyze mathematically as we shall elaborate in the next section.

Due to its random asynchronous nature, Algorithm 5 appears similar to filtering over time varying graphs, which is studied extensively in [88]. However, random asynchronous communications differ from randomly varying graph topologies in two ways: 1) Expected value of the signal depends on the “expected graph" in randomly varying graph topologies [88, Theorem 1], whereas the fixed point does not depend on the update probabilities in the case of asynchronous communications. 2) The graph signal converges in the mean-squared sense in the case of asynchronous communications (see Section 3.4), whereas the signal has a nonzero variance in the case of randomly varying graph topologies [88, Theorem 3].

We also note that the study in [74] proposed a similar algorithm, in which nodes retrieve and aggregate information from a subset of neighbors of fixed size selected uniformly randomly. However, the computational stage of [74] consists of a linear mapping followed by a sigmoidal function, whereas Algorithm 5 uses a linear update model. More importantly, aggregations are done synchronously in [74], that is, all nodes are required to complete the necessary computations before proceeding to the next level of aggregation. On the contrary, nodes aggregate information repetitively and *asynchronously* without waiting for each other in Algorithm 5.

3.4 Convergence of the Proposed Algorithm

For convenience of analysis we define

$$\mathbf{X}_{(k)} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_{N\text{-}1} \quad \mathbf{x}_N] \in \mathbb{C}^{L\times N},$$
$$\mathbf{y}_{(k)} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}, \qquad \mathbf{w}_{(k)} = \begin{bmatrix} w_1 \\ \vdots \\ w_N \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}. \tag{3.38}$$

Here, $\mathbf{X}_{(k)}$ will be called the augmented state variable matrix, and $\mathbf{y}_{(k)} \in \mathbb{C}^N$ is the output vector after k iterations of the algorithm. Also $\mathbf{w}_{(k)} \in \mathbb{C}^N$ is the noise vector at the k^{th} iteration, and $\mathbf{u} \in \mathbb{C}^N$ is the graph input signal as before.

We note that the index k is a global counter that we use to enumerate the iterations. In general, nodes are unaware of the value of k, which is why the augmented variables in (3.38) are indexed with k in parenthesis, but the variables corresponding to individual nodes are not indexed by k at all. Whenever a node completes the execution of the active stage of Algorithm 5, we assume that an iteration has passed. Thus, (3.38) *denotes the variables at the end of the* k^{th} *iteration.* Furthermore, we will use $\mathcal{T}_k$ to denote the set of nodes that get into the active stage simultaneously at the k^{th} iteration.

Algorithm 5 allows the nodes to update their values with different frequencies. Similar to (4.16), we will use the diagonal matrix $\mathbf{P} \in \mathbb{R}^{N\times N}$ to denote the average node (index) selection matrix in the algorithm. In particular, $\mathbf{P} = p\,\mathbf{I}$ corresponds to the case of all the nodes having the same rate of getting into the active stage.

In order to analyze the evolution of the state variables in the algorithm, we first note that the state vector of a node i at the *beginning* of the k^{th} iteration can be written as follows:

$$\mathbf{x}_i = \mathbf{X}_{(k\text{-}1)}\, \mathbf{e}_i, \qquad 1 \le i \le N. \tag{3.39}$$

Thus, if the node i gets into the active stage at the k^{th} iteration, i.e., $i \in \mathcal{T}_k$, then its graph shifted state vector (computed in Line 7 of the algorithm) can be written as follows:

$$\mathbf{x}'_i = \sum_{j \in \mathcal{N}_{\text{in}}(i)} G_{i,j}\, \mathbf{x}_j = \sum_{j} \mathbf{X}_{(k\text{-}1)}\, \mathbf{e}_j\, \mathbf{e}_j^{\mathrm{T}}\, \mathbf{G}^{\mathrm{T}}\, \mathbf{e}_i = \mathbf{X}_{(k\text{-}1)}\, \mathbf{G}^{\mathrm{T}}\, \mathbf{e}_i. \tag{3.40}$$

Therefore, the next value for its state vector is given as follows:

$$\mathbf{X}_{(k)}\, \mathbf{e}_i = \left(\mathbf{A}\, \mathbf{X}_{(k\text{-}1)} \mathbf{G}^{\mathrm{T}} + \mathbf{b}\, (\mathbf{u} + \mathbf{w}_{(k\text{-}1)})^{\mathrm{T}}\right) \mathbf{e}_i, \quad i \in \mathcal{T}_k. \tag{3.41}$$

On the other hand, if the node i does not get into the active stage at the k^{th} iteration, i.e., $i \notin \mathcal{T}_k$, its state vector remains unchanged. Thus, we can write the following:

$$\mathbf{X}_{(k)}\, \mathbf{e}_i = \mathbf{X}_{(k\text{-}1)}\, \mathbf{e}_i, \qquad i \notin \mathcal{T}_k. \tag{3.42}$$

Since both (3.41) and (3.42) are linear in the augmented state variable matrix $\mathbf{X}_{(k)}$, we can transpose, and then vectorize both equations and represent them as follows:

$$(\bar{\mathbf{x}}_k)_i = \begin{cases} (\bar{\mathbf{A}}\, \bar{\mathbf{x}}_{k\text{-}1})_i + (\bar{\mathbf{u}}_{k\text{-}1})_i, & i \in \bar{\mathcal{T}}_k, \\ (\bar{\mathbf{x}}_{k\text{-}1})_i, & i \notin \bar{\mathcal{T}}_k, \end{cases} \tag{3.43}$$

where the variables of the vectorized model are as follows:

$$\begin{aligned} \bar{\mathbf{x}}_k &= \operatorname{vec}\left(\mathbf{X}_{(k)}^{\mathrm{T}}\right), & \bar{\mathbf{A}} &= \mathbf{A} \otimes \mathbf{G}, \\ \bar{\mathbf{u}} &= \mathbf{b} \otimes \mathbf{u}, & \bar{\mathbf{w}}_k &= \mathbf{b} \otimes \mathbf{w}_{(k)}, \end{aligned} \tag{3.44}$$

and $\bar{\mathbf{u}}_k$ is defined similar to (3.2) as $\bar{\mathbf{u}}_k = \bar{\mathbf{u}} + \bar{\mathbf{w}}_k$. Furthermore, the update set $\bar{\mathcal{T}}_k$ of the vectorized model is defined as follows:

$$\bar{\mathcal{T}}_k = \left\{\, i + jN \;\mid\; i \in \mathcal{T}_k, \quad 0 \le j < L \,\right\}, \tag{3.45}$$

which follows from the fact that when a node gets into the active stage, it updates all elements of its own state vector simultaneously according to Line 10 of the algorithm.

We note that the mathematical model in (3.43) appears as a pull-like algorithm, in which nodes retrieve data from their incoming neighbors. However, with the use of a buffer, the model (3.43) can be implemented in a collect-compute-broadcast scheme as proposed in Algorithm 5. See also Figure 3.1.

When the algorithm is implemented in a synchronous manner, the state recursions of (3.43) reduce to the following form:

$$\bar{\mathbf{x}}_k = \bar{\mathbf{A}}\, \bar{\mathbf{x}}_{k\text{-}1} + \bar{\mathbf{u}}_{k\text{-}1}, \tag{3.46}$$

and the following theorem (whose proof is provided in Section 3.7.3) presents the mean-squared error of the algorithm:

Theorem 3.2. *In Algorithm 5, assume that all the nodes on the graph get into the active stage synchronously, and the matrix* $\bar{\mathbf{A}}$ *does not have an eigenvalue equal to* 1. *Then,*

$$\mathbb{E}\big[\big\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\big\|_2^2\big] = \Big\|\big(\mathbf{c}\otimes\mathbf{G}\big)\ \bar{\mathbf{A}}^{k\text{-}1}\ \big(\bar{\mathbf{x}}_0 - \bar{\mathbf{x}}^{\star}\big)\Big\|_2^2 + \sum_{n=0}^{k-1} |h_n|^2\ \mathrm{tr}\Big(\mathbf{G}^n\,\mathbf{\Gamma}\,(\mathbf{G}^n)^{\mathrm{H}}\Big), \quad (3.47)$$

where $\bar{\mathbf{x}}^{\star}$ *is the fixed point of* (3.46)*, and* h_n*'s are the coefficients of the impulse response of the digital filter as in* (3.34)*.*

In (3.47) it is clear that as long as

$$\rho(\bar{\mathbf{A}}) < 1, \quad (3.48)$$

the first term of (3.47) converges to zero irrespective of the initial vector $\bar{\mathbf{x}}_0$, as the iteration progresses. So, from Theorem 3.2 the residual error approaches an error floor:

$$\lim_{k\to\infty} \mathbb{E}\big[\big\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\big\|_2^2\big] = \mathrm{tr}(\mathbf{H}\,\mathbf{\Gamma}), \quad (3.49)$$

where

$$\mathbf{H} = \sum_{n=0}^{\infty} |h_n|^2\,(\mathbf{G}^n)^{\mathrm{H}}\,\mathbf{G}^n. \quad (3.50)$$

Thus, the error floor in the synchronous case depends on the *impulse response* of the underlying digital filter as well as the graph operator, but the similarity transform $\mathbf{T}$ does not affect the error floor. In short, the similarity transform does not affect either the convergence or the error floor in the synchronous case. Note that the stability condition in (3.48) ensures the convergence of (3.50). Note also that $\rho(\bar{\mathbf{A}}) = \rho(\mathbf{A})\,\rho(\mathbf{G})$ in view of (3.44).

Next consider the asynchronous case. The equivalent model of the algorithm in (3.43) is in the form of (3.12), thus the results presented in Section 3.2 (Corollary 3.1 in particular) can be used to study the convergence of the algorithm. In this regard, we present the following theorem, whose complete proof is given in Section 3.7.4:

Theorem 3.3. *In Algorithm 5, let* $\mathbf{P}$ *denote the average node selection matrix and* $\mathbf{\Gamma}$ *the covariance matrix of the measurement noise. If the state transition matrix* $\mathbf{A}$ *of the filter, and the operator* $\mathbf{G}$ *of the graph satisfy the following:*

$$\|\mathbf{A}\|_2^2\ \mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G} \prec \mathbf{P}, \quad (3.51)$$

then

$$\lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\|_2^2\big] \le \mathrm{tr}(\mathbf{R}\,\mathbf{\Gamma}), \quad (3.52)$$

where

$$\mathbf{R} = \frac{\|\mathbf{b}\|_2^2 \ \|\mathbf{c}\|_2^2 \ \|\mathbf{G}\|_2^2}{\lambda_{\min}\Big(\mathbf{P} - \|\mathbf{A}\|_2^2 \, \mathbf{G}^{\mathrm{H}} \, \mathbf{P} \, \mathbf{G}\Big)} \ \mathbf{P} + |d|^2 \ \mathbf{I}. \tag{3.53}$$

Theorem 3.3 presents an upper bound on the mean-squared error. In the noise-free case ($\mathbf{\Gamma} = \mathbf{0}$), the right-hand-side of (3.52) becomes zero, and the condition (3.51) ensures the convergence of the output signal to the desired filtered signal in the mean-squared sense. We note also that the right-hand-side of (3.52) is linear in the noise covariance matrix, which implies that the error floor of the algorithm increases at most linearly with the input noise. This will be numerically verified later in Section 3.5.1. (See Figure 3.4b.) In fact, it is possible to integrate stochastic averaging techniques studied in [6, 7] into Algorithm 5 in order to overcome the error due to noise at expense of a reduced convergence rate.

We conclude by noting that graph filtering implementations considered in [158, 149, 150, 154, 157, 87, 88, 108, 109] are likely to tolerate asynchronicity up to a certain degree. In fact, [109] presented numerical evidences in this regard. This is not surprising because linear asynchronous fixed-point iterations are known to converge under some conditions [32, 14]. The main difference of Algorithm 5 studied in this chapter is due to its *proven convergence* under some mild and interpretable conditions with the assumed random asynchronous model (Theorem 3.3).

3.4.1 Selection of the Similarity Transform

In addition to the dependency on the graph operator and the average node selection matrix, the sufficiency condition (3.51) depends also on the realization of the filter of interest. Thus, in the asynchronous case, both the condition for convergence and the error bound depend on the similarity transform. Since the condition becomes more relaxed as the state transition matrix $\mathbf{A}$ has a smaller spectral norm, it is important to select the similarity transform $\mathbf{T}$ in (3.35) in such a way that $\mathbf{A}$ has the minimum spectral norm.

Due to their robustness, minimum-norm realizations of digital filters have been studied extensively in signal processing [12, 120, 198]. A minimum-norm implementation corresponds to an appropriate selection of the similarity transform $\mathbf{T}$ in (3.35) due to the following inequality:

$$\|\mathbf{A}\|_2 \geq \rho(\mathbf{A}) = \rho(\widehat{\mathbf{A}}). \tag{3.54}$$

The lower bound $\rho(\widehat{\mathbf{A}})$ depends only on the coefficients of the polynomial $q(x)$ due to the definition of $\widehat{\mathbf{A}}$ in (3.36).

The lower bound in (3.54) may *not* be achieved with equality in general, and we will consider one such example in the next section. Nevertheless, it is known that the companion matrix $\widehat{\mathbf{A}}$ is diagonalizable if and only if the digital filter in (3.34) has L distinct poles [85]. That is to say, when there are L distinct nonzero z_n's such that $q(z_n^{-1}) = 0$, we can write the following eigenvalue decomposition:

$$\widehat{\mathbf{A}} = \mathbf{V}_{\widehat{\mathbf{A}}} \, \mathbf{\Lambda}_{\widehat{\mathbf{A}}} \, \mathbf{V}_{\widehat{\mathbf{A}}}^{-1}, \tag{3.55}$$

where $\mathbf{\Lambda}_{\widehat{\mathbf{A}}}$ is a diagonal matrix with z_n^{-1}'s on the diagonal, and $\mathbf{V}_{\widehat{\mathbf{A}}}$ is a Vandermonde matrix corresponding to z_n^{-1}'s. If the similarity transform $\mathbf{T}$ is selected according to (3.55), then the bound in (3.54) is indeed achieved. More precisely,

$$\mathbf{T} = \mathbf{V}_{\widehat{\mathbf{A}}} \quad \Longrightarrow \quad \mathbf{A} = \mathbf{\Lambda}_{\widehat{\mathbf{A}}} \quad \Longrightarrow \quad \|\mathbf{A}\|_2 = \rho(\widehat{\mathbf{A}}). \tag{3.56}$$

Thus, the most relaxed version of the sufficiency condition of Theorem 3.3 is obtained when the updates of Algorithm 5 are implemented using the similarity transform given in (3.56).

When the filter (3.34) has repeated poles, the companion matrix $\widehat{\mathbf{A}}$ is not diagonalizable, hence an implementation achieving the bound (3.54) does not exist [12]. Nevertheless, the study [12] discussed that for any $\epsilon > 0$, there exists a realization with a state transition matrix $\mathbf{A}$ such that

$$\|\mathbf{A}\|_2 \leq \rho(\widehat{\mathbf{A}}) + \epsilon. \tag{3.57}$$

Therefore, it is always possible to obtain “almost minimum" realizations with the spectral norm arbitrarily close to the lower bound in (3.54). As a particular example, the case of FIR graph filters will be considered in the next section.

3.4.2 The Case of Polynomial Filters

Polynomial (FIR) graph filters can be considered as a special case of the rational graph filter (3.29), in which the denominator is selected as $q(x) = 1$ so that $q(\mathbf{G}) = \mathbf{I}$, and the filtered signal in (3.30) reduces to $\widetilde{\mathbf{u}} = p(\mathbf{G})\,\mathbf{u}$. In this case, the companion matrix $\widehat{\mathbf{A}}$ (direct form implementation) has the following form:

$$\widehat{\mathbf{A}} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{L\times L}, \tag{3.58}$$

which has all eigenvalues equal to zero, so that $\rho(\widehat{\mathbf{A}}) = 0$. As a result, no realization of a polynomial filter can achieve the lower bound (3.54) since $\|\mathbf{A}\|_2 = 0$ implies $\mathbf{A} = \mathbf{0}$. However, the spectral norm of a realization can be made arbitrarily small. In particular, consider the following similarity transform:

$$\mathbf{T} = \text{diag}\left([1 \quad \epsilon \quad \epsilon^2 \quad \cdots \quad \epsilon^{L\text{-}1}] \right), \tag{3.59}$$

where ϵ is an arbitrary nonzero complex number. Then, the corresponding realization $\mathbf{A}$ can be found as follows:

$$\mathbf{A} = \mathbf{T}^{\text{-}1}\, \widehat{\mathbf{A}}\, \mathbf{T} = \epsilon\, \widehat{\mathbf{A}} \qquad \Longrightarrow \qquad \|\mathbf{A}\|_2 = |\epsilon|. \tag{3.60}$$

Thus, it is possible to select a value for ϵ (with a sufficiently small magnitude) in order to satisfy the condition (3.51). (See [101, Fact 2.5.4].) Such a selection is not unique in general, and one can easily find a value for ϵ satisfying the following:

$$|\epsilon| < \left(\|\mathbf{G}\|_2 \sqrt{\|\mathbf{P}\|_2 \, \|\mathbf{P}^{\text{-}1}\|_2} \right)^{\text{-}1}, \tag{3.61}$$

which ensures that the condition (3.51) is met.

As a result, for *any* graph operator $\mathbf{G}$ and average node selection matrix $\mathbf{P}$, it is always possible to implement *any* polynomial filter in a random node-asynchronous manner that is guaranteed to converge in the mean-squared sense. However, we note that $\mathbf{T}$ given in (3.59) may not be the optimal similarity transform in general.

We also note that when a polynomial filter is implemented in a synchronous manner, Theorem 3.2 shows that the algorithm reaches the error floor after L iterations since $\widehat{\mathbf{A}}$ in (3.58) is a nil-potent matrix and $\widehat{\mathbf{A}}^n = \mathbf{A}^n = \mathbf{0}$ for $n \geq L$. This convergence behavior will be verified numerically later in Section 3.5.4. The error bound still depends on $\mathbf{T}$ because of $\|\mathbf{A}\|_2$.

3.5 Numerical Simulations

We now simulate the proposed algorithm on the graph visualized in Figure 3.2. This is a random geometric graph on $N = 150$ nodes, in which nodes are distributed over the region $[0\ 1] \times [0\ 1]$ uniformly at random. Two nodes are connected to each other if the distance between them is less than 0.15, and the graph is undirected. The graph operator, the matrix $\mathbf{G} \in \mathbb{R}^{N\times N}$, is selected as the Laplacian matrix whose eigenvalues can be sorted as follows:

$$0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N = \rho(\mathbf{G}) = \|\mathbf{G}\|_2 = 16.8891, \tag{3.62}$$

where the spectral norm of $\mathbf{G}$ is computed numerically, and the equality between the spectral radius and the spectral norm follows from the fact that $\mathbf{G}$ is a real symmetric matrix.

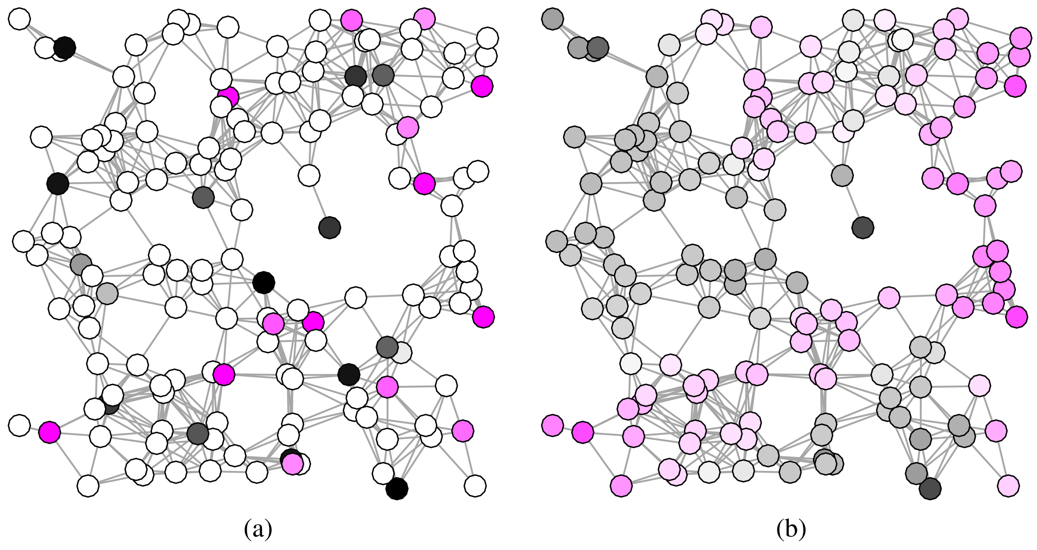


Figure 3.2: Visualization of the signals on the graph. Colors black and pink represent positive and negative values, respectively. Intensity of a color represents the magnitude. (a) The graph signal $\mathbf{u}$ that has nonzero values on 30 nodes. (b) The filtered signal $\widetilde{\mathbf{u}}$ on the graph with the filter in (3.67).

For the numerical simulations we consider the following smoothing problem: assume that we are given the graph signal $\mathbf{u} \in \mathbb{R}^N$ that has only 30 nonzero entries, which is visualized in Figure 3.2a. It is clear that the signal $\mathbf{u}$ is not smooth on the graph. In order to obtain a smoothed version of $\mathbf{u}$, which will be denoted by $\widetilde{\mathbf{u}} \in \mathbb{R}^N$, we will apply a low-pass graph filter to the signal $\mathbf{u}$. In this regard, we will consider examples of rational (IIR) graph filters in Sections 3.5.1, 3.5.2, and 3.5.3, and consider a polynomial (FIR) filter in Section 3.5.4.

Throughout the simulations we will consider a particular stochastic model for the selection of the nodes. That is, in each iteration of Algorithm 5 we will select a subset of size μ uniformly randomly among all subsets of size μ. For this particular model, the average node selection matrix becomes:

$$\mathbf{P} = \frac{\mu}{N}\,\mathbf{I}. \tag{3.63}$$

We note that the case of $\mu = N$ corresponds to the synchronous implementation of the algorithm. With $\mathbf{P}$ as in (3.63), we note that the sufficiency condition (3.51), which ensures the convergence of Algorithm 5, reduces to the following form:

$$\|\mathbf{A}\|_2 \;\|\mathbf{G}\|_2 < 1, \tag{3.64}$$

which does not depend on μ. Furthermore, the bound on the noise floor given by (3.52) reduces to the following form:

$$\lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\|_2^2\big] \leq \operatorname{tr}(\mathbf{\Gamma}) \left(\frac{\|\mathbf{b}\|_2^2 \;\|\mathbf{c}\|_2^2 \;\|\mathbf{G}\|_2^2}{1 - \|\mathbf{A}\|_2^2 \;\|\mathbf{G}\|_2^2} + |d|^2 \right). \tag{3.65}$$

For the sake of simplicity we will assume that the covariance matrix of the measurement noise is as follows:

$$\mathbf{\Gamma} = \sigma^2\,\mathbf{I}, \tag{3.66}$$

where σ^2 will denote the variance of input noise.

3.5.1 An Example of a Rational Graph Filter

In this section we will consider a rational filter (3.27) constructed with the following polynomials of order $L = 3$:

$$p(x) = (1-\gamma\,x)^3, \quad q(x) = 1 + \sum_{n=1}^{3} \gamma^n\,x^n, \quad \gamma = 0.055, \tag{3.67}$$

where the value of γ is selected in such a way that it normalizes the spectrum of $\mathbf{G}$, that is, $|\gamma|\;\|\mathbf{G}\|_2 < 1$ is satisfied.

The frequency response of the filter in (3.67) on the graph is visualized in Figure 3.3a, which shows that the filter has low-pass characteristics on the graph. When compared with the input signal $\mathbf{u}$, the filtered signal $\widetilde{\mathbf{u}}$ has a lower amount of projection on the eigenvectors with larger eigenvalues as shown in Figure 3.3b. Since $\widetilde{\mathbf{u}}$ mainly contains low frequency components (eigenvectors with small eigenvalues [160]), $\widetilde{\mathbf{u}}$ is smoother on the graph as visualized in Figure 3.2b.

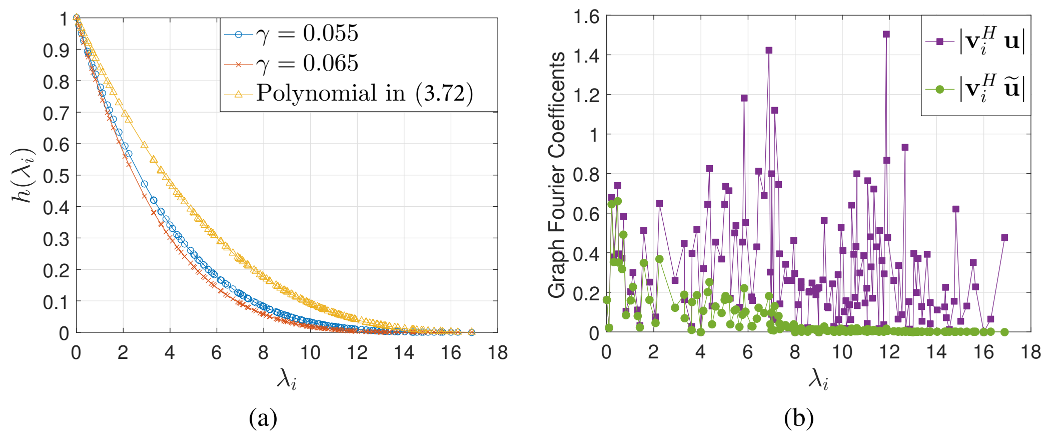


Figure 3.3: (a) Response of the rational filter $h(\lambda)$ constructed with (3.67). (b) Magnitude of the graph Fourier transforms of $\mathbf{u}$ and $\widetilde{\mathbf{u}}$ where $(\lambda_i, \mathbf{v}_i)$ denotes an eigenpair of $\mathbf{G}$.

We now consider the implementation of the filter (3.67) using Algorithm 5. In this regard, we first construct the direct form implementation of the corresponding digital filter as in (3.36):

$$\widehat{\mathbf{A}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\gamma^3 & -\gamma^2 & -\gamma \end{bmatrix}, \qquad \widehat{\mathbf{b}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \widehat{\mathbf{c}}^{\mathrm{T}} = \begin{bmatrix} -2\gamma^3 \\ 2\gamma^2 \\ -4\gamma \end{bmatrix}, \qquad \widehat{d} = 1. \quad (3.68)$$

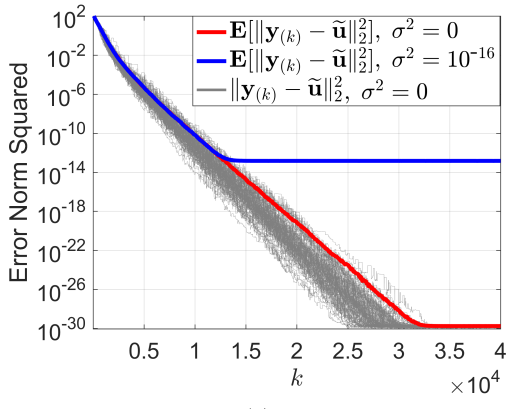
It is readily verified that the matrix $\widehat{\mathbf{A}}$ in (3.68) has $L = 3$ distinct eigenvalues that are given as $\{-\gamma,\ j\gamma,\ -j\gamma\}$. Thus, the similarity transform $\mathbf{T}$ can be selected as the eigenvectors of $\widehat{\mathbf{A}}$ as in (3.56), which corresponds to the Vandermonde matrix constructed with $\{-\gamma,\ j\gamma,\ -j\gamma\}$. As a result, the corresponding realization of the filter according to (3.35) is given as follows:

$$\mathbf{A} = \gamma \begin{bmatrix} -1 & 0 & 0 \\ 0 & j & 0 \\ 0 & 0 & -j \end{bmatrix}, \qquad \mathbf{b} = \frac{-1}{4\gamma^2} \begin{bmatrix} -2 \\ 1+j \\ 1-j \end{bmatrix}, \qquad \mathbf{c}^{\mathrm{T}} = \gamma^3 \begin{bmatrix} -8 \\ 2+2j \\ 2-2j \end{bmatrix}. \quad (3.69)$$

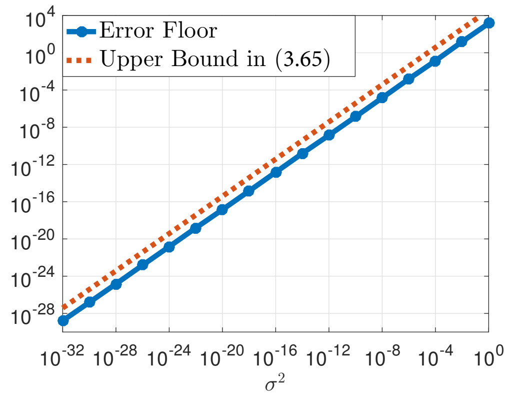
Since $\|\mathbf{A}\|_2 = \rho(\mathbf{A}) = |\gamma|$, we note that (3.64) is satisfied for the value of γ in (3.67), thus Algorithm 5 converges in the mean-squared sense when no input noise is present, and when there is noise, it reaches an error floor upper bounded as in (3.65).

In the first set of simulations of Algorithm 5 we consider the case of $\mu = 1$, i.e., only one randomly selected node is updated per iteration. In order to verify the convergence numerically, we simulated independent runs of Algorithm 5 with the filter realization in (3.69) and computed the mean-squared error by averaging over 10^4 independent runs. In order to present the effect of the measurement noise, we consider the case of $\sigma^2 = 10^{-16}$ as well as the noise-free case. Figure 3.4a presents the corresponding mean-squared errors together with the error in the noise-free case for 100 different realizations. Due to the random selection of the nodes, the residual itself is a random quantity, which does not decrease monotonically as seen in Figure 3.4a. Nevertheless, the *expectation* of the error norm decreases monotonically until it reaches the error floor. We note that the error floor in the noise-free case corresponds to the numerical precision of the numerical environment (MATLAB).

In order to present the effect of the noise variance on the error floor, we run Algorithm 5 for different values of σ^2 for $k_{\max} = 4 \cdot 10^4$ iterations (which ensures that the algorithm reaches an error floor as seen in Figure 3.4a) while selecting only $\mu = 1$ node per iteration. The error floor corresponding to different values of σ^2 together with the upper bound in (3.65) are presented in Figure 3.4b. In addition to the upper bound (3.65) scaling linearly with the noise variance, Figure 3.4b shows that the error floor itself scales almost linearly with the noise variance as well.



(a)



(b)

Figure 3.4: (a) Squared error norm in 100 different independent realizations together with the mean squared error of Algorithm 5 with the implementation in (3.69). (b) Error floor of the algorithm as a function of the input noise together with the bound in (3.65).

We note that the filter realization in (3.69) ensures the convergence of the algorithm irrespective of the value of μ. However, the *convergence rate* of the algorithm does depend on the value of μ in general. This point will be demonstrated in the following set of simulations, in which we use the filter realization in (3.69) and set the noise variance as $\sigma^2 = 10^{-16}$. In order to obtain a fair comparison between different values of μ, we fix the total number of updates to be 25000, so the algorithm gets $\lceil 25000/\mu \rceil$ iterations. We run the algorithm independently 10^5 times for each value of $\mu = \{1, \cdots, N\}$ and present the corresponding mean-squared errors as a function of the number of updated nodes in Figure 3.5.

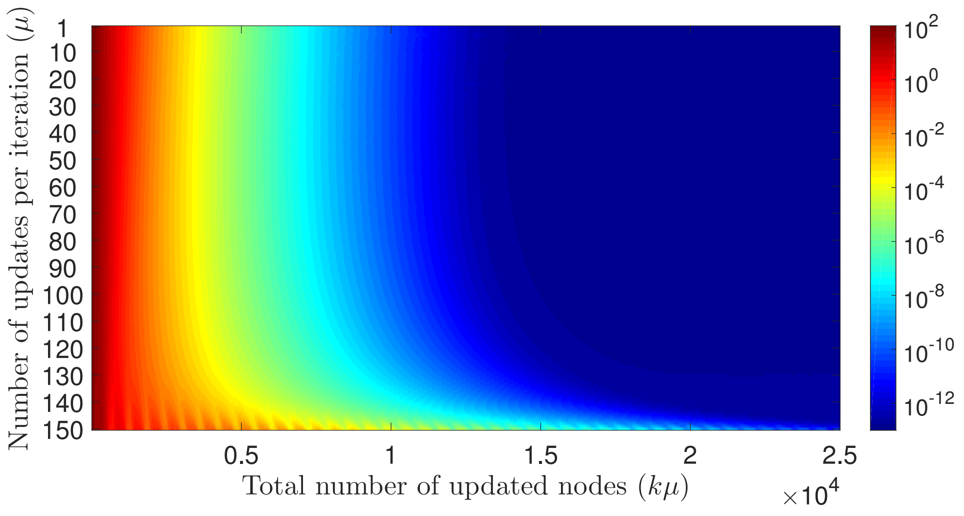


Figure 3.5: The mean squared error of Algorithm 5 when more than one node is updated simultaneously with noise variance $\sigma^2 = 10^{-16}$. The first row in the figure corresponds to Figure 3.4a.

We first point out that Figure 3.5 verifies the convergence of the algorithm for all possible values of μ. More interestingly, the figure shows also that *the algorithm gets faster as it gets more asynchronous (small μ).* Equivalently, for a given fixed amount of computational budget (total number of nodes to be updated), having nodes updated randomly and asynchronously results in a smaller error than having synchronous updates. However, it is important to emphasize that the behavior shown in Figure 3.5 is *not* typical for the algorithm; rather, it depends on the underlying filter. Indeed we will find a similar behavior in Section 3.5.3, but an opposite behavior later in Section 3.5.4. We also note that for the case of zero-input, Section 2.3.3 of this thesis theoretically discussed the conditions under which randomized asynchronicity results in a faster convergence.

3.5.2 Updates with Failing Broadcasts

Algorithm 5 assumes that when a node broadcasts the most recent value of its state vector to its outgoing neighbors (Line 11 of Algorithm 5), all the recipient nodes reliably receive the message. However, in a more realistic scenario the broadcasted message may not be received by some of the recipients due to unreliable communication between the nodes. In the case of such communication failures, the theoretical analysis presented in Section 3.4 (Theorems 3.2 and 3.3) becomes inconclusive regarding the convergence of the algorithm. Nevertheless, in this section we will numerically verify that the proposed algorithm is robust to such communication failures.

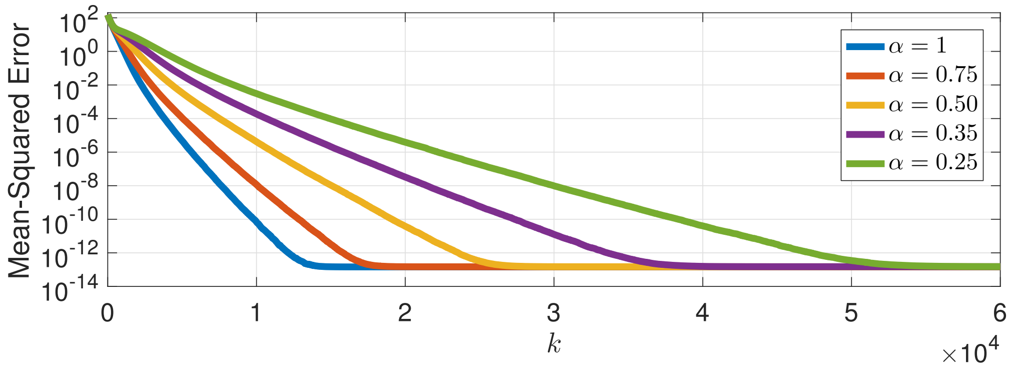
Similar to the previous section, in this set of simulations we use the filter realization in (3.69) (with γ selected as in (3.67)) and set the noise variance as $\sigma^2 = 10^{-16}$. However, we modify the implementation in such a way that when a node broadcasts its state vector, a recipient node is assumed to receive the message with probability α independent of the other nodes. Thus, $\alpha = 1$ corresponds to the case where convergence is guaranteed by Theorem 3.3.

We consider two cases, namely $\mu = 1$ (one node is activated randomly in each iteration) and $\mu = N$ (all nodes are activated synchronously). In both cases the broadcasted messages are delivered with probability α. The mean squared errors for these two cases are given in Figures 3.6a and 3.6b, respectively.

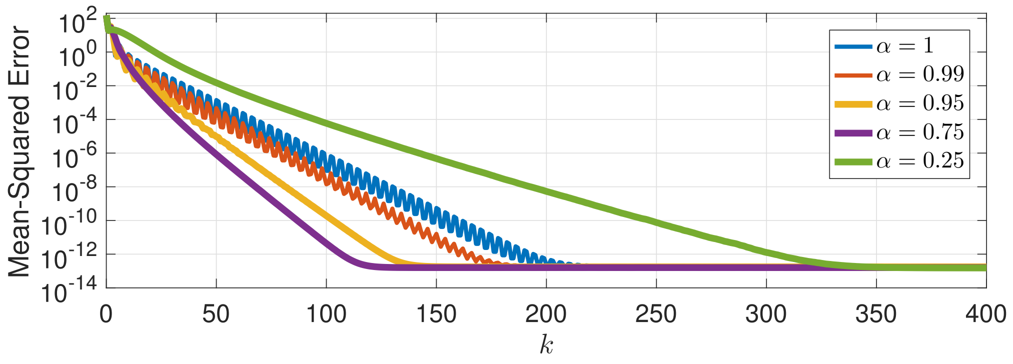
Both Figures 3.6a and 3.6b verify the convergence of the algorithm even in the case of unreliable communication. In the case of $\mu = 1$, Figure 3.6a suggests that the convergence rate of the algorithm decreases as the communications become more unreliable (the value of α gets smaller). However, for the case of $\mu = N$, Figure 3.6b presents an unexpected behavior. The case of reliable communications ($\alpha = 1$) does *not* result in the fastest rate of convergence. When the communications fail with some probability, the algorithm may converge *faster*. While the behavior is surprising, it is consistent with Figure 3.5 in the sense that fully synchronous iterations are slower than asynchronous counterparts for the specific filter in (3.69). Even when the nodes get updated synchronously, failed broadcasts break the overall synchrony over the network, hence the algorithm converges faster. However, when the communications fail with high probability (e.g., the case of $\alpha = 0.25$ in Figure 3.6b), the convergence is indeed slower. We also note that the behaviors demonstrated in Figure 3.6a and Figure 3.6b remain the same even for the noise-free ($\sigma^2 = 0$) case.

3.5.3 A Case of Convergence Only with Asynchronous Iterations

The results in Section 3.5.1 (namely Figure 3.5) showed that the proposed algorithm may converge faster as the iterations get more asynchronous (i.e., the value of μ gets



(a)



(b)

Figure 3.6: The mean-squared error of the algorithm with (a) $\mu = 1$, (b) $\mu = N$, when the broadcasted messages are delivered successfully with probability α.

smaller). In this section we will demonstrate an even more interesting behavior, where the algorithm converges *only if the iterations are sufficiently asynchronous (μ is smaller than a threshold).*

In this subsection, we will use the same filter realization as in (3.69), but use the following value for the parameter γ:

$$\gamma = 0.065, \tag{3.70}$$

which results in a slight change in the response of the filter as presented in Figure 3.3a. More importantly, for the value of γ in (3.70), the sufficiency condition in (3.64) is not satisfied, thus Theorem 3.3 is *inconclusive* regarding the convergence of the algorithm with asynchronous iterations. In fact, Theorem 3.2 tells that the algorithm *diverges* in the synchronous case since the matrix $\bar{\mathbf{A}}$ is unstable for the value of γ in (3.70):

$$\rho(\bar{\mathbf{A}}) = \rho(\mathbf{A} \otimes \mathbf{G}) = \rho(\mathbf{A})\, \rho(\mathbf{G}) = |\gamma|\, \rho(\mathbf{G}) \approx 1.0978. \tag{3.71}$$

In order to examine the convergence behavior of the algorithm, we repeat the simulations done in Section 3.5.1 with the value of γ set as in (3.70). That is, the noise variance is set to be $\sigma^2 = 10^{-16}$, and the algorithm is simulated independently 10^4 times for each value of $\mu = \{1, \cdots, N\}$. The corresponding mean-squared errors are presented in Figure 3.7.

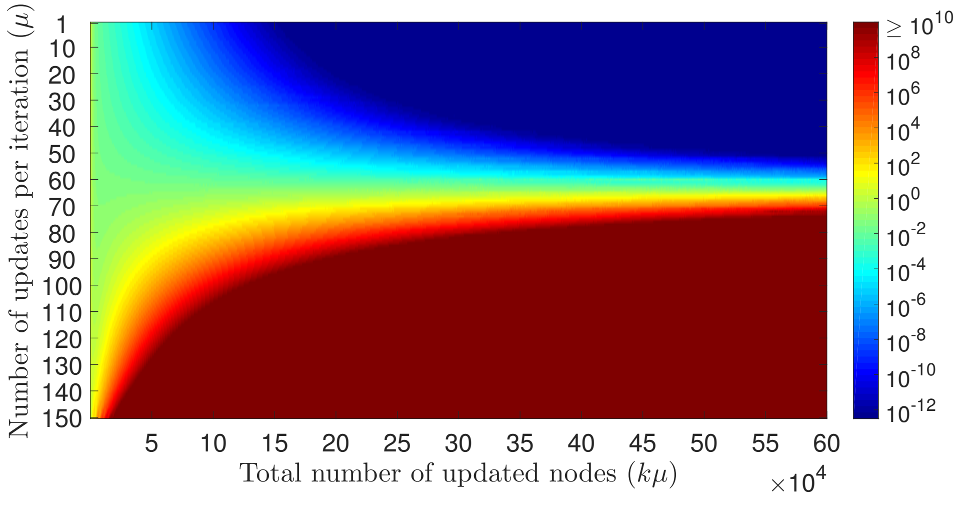


Figure 3.7: The mean-squared error of Algorithm 5 for a case of an unstable augmented state transition matrix $\bar{\mathbf{A}}$. Input noise variance is $\sigma^2 = 10^{-16}$.

For the specific filter considered in this simulations, Figure 3.7 shows that the convergence of the algorithm displays an obvious phase-transition in terms of the amount of asynchronicity. That is to say, the algorithm convergences only if the number of simultaneously updated nodes satisfies $\mu \leq 66$, and the algorithm diverges otherwise. Therefore, a specific amount of asynchronicity is in fact required for the convergence in this example.

Although the theoretical analysis of the algorithm presented in Section 3.4 does not explain the phenomena observed here, for the zero-input case Chapter 2 of this thesis proved that the convergence can be achieved for some unstable systems as long as the iterations are sufficiently asynchronous. Simulation results presented in Figure 3.7 shows that a similar behavior exists even when the input is nonzero. See Chapter 4 for further details.

3.5.4 An Example of a Polynomial Graph Filter

Now consider the implementation of a polynomial (FIR) graph filter with the proposed algorithm. In particular, we consider the following filter of order $L = 3$:

$$p(x) = (1 - \gamma x)^3, \qquad q(x) = 1, \qquad \gamma = 0.055, \tag{3.72}$$

which has low-pass characteristics on the graph as visualized in Figure 3.3a. In the implementation of the filter we use the following similarity transformation $\mathbf{T} = \operatorname{diag}([1 \;\; \gamma \;\; \gamma^2])$ so that the realization of the filter has the following form:

$$\mathbf{A} = \gamma \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{b} = \frac{1}{\gamma^2} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathbf{c}^{\mathrm{T}} = \gamma^3 \begin{bmatrix} -1 \\ 3 \\ -3 \end{bmatrix}, \qquad d = 1, \tag{3.73}$$

which satisfies $\|\mathbf{A}\|_2 \, \|\mathbf{G}\|_2 = |\gamma| \, \|\mathbf{G}\|_2 < 1$ for the value of γ in (3.72), thus Theorem 3.3 ensures the convergence of the algorithm irrespective of the value of μ. For the particular case of synchronous iterations, $\mu = N = 150$, Theorem 3.2 shows that the algorithm converges after $L = 3$ iterations, i.e., the algorithm reaches the error floor in (3.49) when $k\mu \geq 600$.

In order to examine the convergence behavior of the algorithm with a polynomial filter, we repeat the simulations done in Sections 3.5.1 and 3.5.3 with the filter realization in (3.73). That is, the noise variance is set to be $\sigma^2 = 10^{-16}$, and the algorithm is simulated for each value of $\mu = \{1, \cdots, N\}$. The corresponding mean-squared errors are presented in Figure 3.8.

We note that the convergence behavior of the algorithm with the polynomial filter in (3.73) differs from that of the filter considered in Section 3.5.1. In particular, the algorithm reaches the error floor after $k\mu \geq 600$ in the synchronous case (which is proven by Theorem 3.2), and *the algorithm gets slower as it gets more asynchronous.* When the results presented in Figure 3.5 and Figure 3.8 are considered together, a definite conclusion cannot be drawn regarding the effect of the asynchronicity on the convergence rate. Depending on the underlying graph filter, the asynchronicity may result in a faster or a slower convergence of the proposed algorithm.

3.6 Conclusions

In this chapter, we proposed a node-asynchronous implementation of rational graph filters, in which nodes on the graph follow a collect-compute-broadcast scheme in a randomized manner: in the passive stage a node only collects data, and when it gets activated randomly it runs a local state recursion for the filter, and then broadcasts

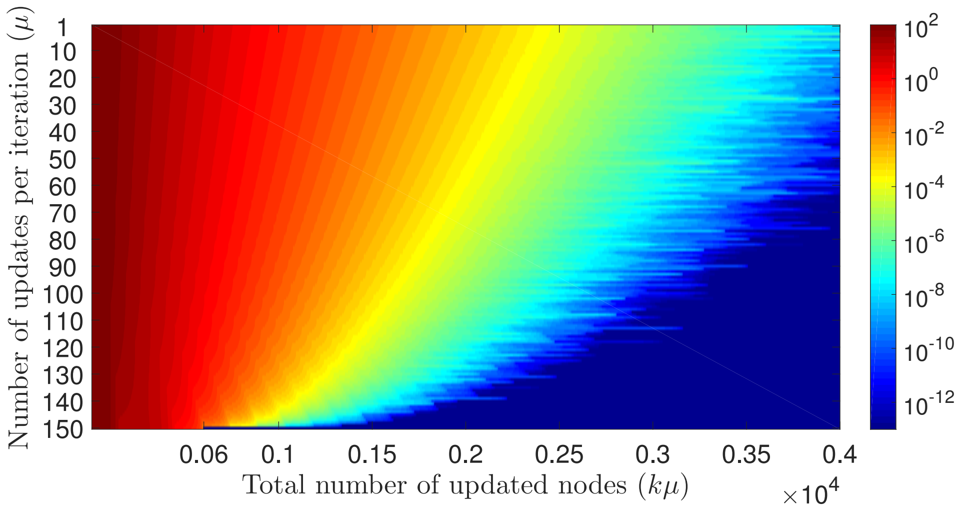


Figure 3.8: The mean-squared error of Algorithm 5 for the case of the polynomial (FIR) filter described in (3.72) with the input noise variance $\sigma^2 = 10^{-16}$.

its most recent value. In order to analyze the proposed method, we first studied a more general case of randomized asynchronous state recursions and presented a sufficiency condition that ensures the convergence in the mean-squared sense. Based on these results, we proved the convergence of the proposed algorithm in the mean-squared sense when the graph operator, the average update rate of the nodes and the filter of interest satisfy a certain condition. We simulated the proposed algorithm under different conditions and verified its convergence numerically.

Simulation results indicated that the presented sufficient condition is not necessary for the convergence of the algorithm. Moreover, the algorithm was observed to be robust to the communication failures between the nodes. It was observed also that the asynchronicity may increase the rate of convergence. Furthermore, simulations revealed that the proposed algorithm can converge even with an unstable filter if the nodes behave sufficiently asynchronously. Deeper theoretical analysis of some of these experimental observations is left for the future. For future studies, it would be interesting to consider the randomized asynchronous scenario in which nodes get updated depending on the values they have.

3.7 Appendices

3.7.1 Proof of Theorem 3.1

The update model (3.12) can be written as follows:

$$\mathbf{x}_k = \sum_{i \notin \mathcal{T}_k} \mathbf{e}_i \, \mathbf{e}_i^{\mathrm{H}} \, \mathbf{x}_{k\text{-}1} + \sum_{i \in \mathcal{T}_k} \mathbf{e}_i \, \mathbf{e}_i^{\mathrm{H}} \left(\mathbf{A} \, \mathbf{x}_{k\text{-}1} + \mathbf{u} + \mathbf{w}_{k\text{-}1} \right), \tag{3.74}$$

$$= \mathbf{x}_{k\text{-}1} + \mathbf{D}_{\mathcal{T}_k} \left(\, (\mathbf{A} - \mathbf{I}) \, \mathbf{x}_{k\text{-}1} + \mathbf{u} + \mathbf{w}_{k\text{-}1} \, \right), \tag{3.75}$$

which can be re-written in terms of the residual vector $\mathbf{r}_k$ defined in (3.5) as follows:

$$\mathbf{r}_k = \left(\mathbf{I} + \mathbf{D}_{\mathcal{T}_k} \, (\mathbf{A} - \mathbf{I}) \, \right) \, \mathbf{r}_{k\text{-}1} + \mathbf{D}_{\mathcal{T}_k} \, \mathbf{w}_{k\text{-}1}. \tag{3.76}$$

Using the assumption that the residual vector $\mathbf{r}_{k\text{-}1}$, the index-selection matrix $\mathbf{D}_{\mathcal{T}_k}$ and the noise term $\mathbf{w}_{k\text{-}1}$ are uncorrelated with each other, and the assumption that $\mathbf{w}_{k\text{-}1}$ has a zero mean, the expected residual norm $\mathbf{r}_k$ conditioned on the previous residual $\mathbf{r}_{k\text{-}1}$ can be written as follows:

$$\mathbb{E}\left[\|\mathbf{r}_k\|_2^2 \mid \mathbf{r}_{k\text{-}1} \right] = \mathbb{E}\left[\left\| \left(\mathbf{I} + \mathbf{D}_{\mathcal{T}_k} (\mathbf{A} - \mathbf{I}) \right) \mathbf{r}_{k\text{-}1} \right\|_2^2 \right] + \mathbb{E}\left[\|\mathbf{D}_{\mathcal{T}_k} \, \mathbf{w}_{k\text{-}1}\|_2^2 \right]. \tag{3.77}$$

The first term on the right-hand-side of (3.77) can be written as follows:

$$\mathbb{E}\left[\mathbf{r}_{k\text{-}1}^{\mathrm{H}} \left(\mathbf{I} + (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{D}_{\mathcal{T}_k} \right) \left(\mathbf{I} + \mathbf{D}_{\mathcal{T}_k} (\mathbf{A} - \mathbf{I}) \right) \mathbf{r}_{k\text{-}1} \right] = \mathbf{r}_{k\text{-}1}^{\mathrm{H}} \left(\mathbf{I} + \mathbf{A}^{\mathrm{H}} \, \mathbf{P} \, \mathbf{A} - \mathbf{P} \right) \mathbf{r}_{k\text{-}1}, \tag{3.78}$$

which can be upper and lower bounded as $\Psi \, \|\mathbf{r}_{k\text{-}1}\|_2^2$ and $\psi \, \|\mathbf{r}_{k\text{-}1}\|_2^2$, respectively, where Ψ and ψ are defined as in (3.18).

The second term on the right-hand-side of (3.77) can be written as follows:

$$\mathbb{E}\left[\mathbf{w}_{k\text{-}1}^{\mathrm{H}} \, \mathbf{D}_{\mathcal{T}_k} \, \mathbf{w}_{k\text{-}1} \right] = \mathrm{tr}\left(\mathbb{E}\left[\mathbf{D}_{\mathcal{T}_k} \, \mathbf{w}_{k\text{-}1} \, \mathbf{w}_{k\text{-}1}^{\mathrm{H}} \right] \right) = \mathrm{tr}(\mathbf{P} \, \mathbf{\Gamma}), \tag{3.79}$$

where we use the fact that $\mathbf{D}_{\mathcal{T}_k}^{\mathrm{H}} \, \mathbf{D}_{\mathcal{T}_k} = \mathbf{D}_{\mathcal{T}_k}$, and the assumption that the noise and the index-selection are uncorrelated.

Thus, the conditional expected residual norm can be upper and lower bounded as follows:

$$\psi \, \|\mathbf{r}_{k\text{-}1}\|_2^2 \leq \mathbb{E}\left[\|\mathbf{r}_k\|_2^2 \mid \mathbf{r}_{k\text{-}1} \right] - \mathrm{tr}(\mathbf{P} \, \mathbf{\Gamma}) \leq \Psi \, \|\mathbf{r}_{k\text{-}1}\|_2^2. \tag{3.80}$$

By taking expectation of (3.80) with respect to the previous residual $\mathbf{r}_{k\text{-}1}$, we obtain the following:

$$\psi \, \mathbb{E}\left[\|\mathbf{r}_{k\text{-}1}\|_2^2 \right] \leq \mathbb{E}\left[\|\mathbf{r}_k\|_2^2 \right] - \mathrm{tr}(\mathbf{P} \, \mathbf{\Gamma}) \leq \Psi \, \mathbb{E}\left[\|\mathbf{r}_{k\text{-}1}\|_2^2 \right]. \tag{3.81}$$

The iterative use of the inequalities in (3.81) yields the results given in (3.17).

3.7.2 Proof of Lemma 3.1

We first note that by left and right multiplying with $\mathbf{P}^{-1/2}$, the condition (3.19) can be equivalently written as:

$$\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P} \quad \Longleftrightarrow \quad \big\|\mathbf{P}^{1/2}\,\mathbf{A}\,\mathbf{P}^{-1/2}\big\|_2 < 1, \tag{3.82}$$

which proves the implication in (3.22). (Consider $\mathbf{P} = p\,\mathbf{I}$.)

We now prove the first implication of (3.21): Lemma 2.7.25 of [101] shows that $\rho(|\mathbf{A}|) < 1$ if and only if $|\mathbf{A}| \in \mathcal{D}_{\mathrm{d}}$. Since $|\mathbf{A}|$ is Schur diagonally stable, there exits a positive diagonal $\mathbf{P}$ such that $\|\mathbf{P}^{1/2}|\mathbf{A}|\mathbf{P}^{-1/2}\|_2 < 1$ due to (3.82). Then,

$$\big\|\mathbf{P}^{1/2}|\mathbf{A}|\mathbf{P}^{-1/2}\big\|_2 = \big\|\,|\mathbf{P}^{1/2}\mathbf{A}\mathbf{P}^{-1/2}|\,\big\|_2 \geq \big\|\mathbf{P}^{1/2}\mathbf{A}\mathbf{P}^{-1/2}\big\|_2, \tag{3.83}$$

where the equality follows from the fact that $\mathbf{P}$ is a positive diagonal matrix, and the inequality follows from the fact that $\|\,|\mathbf{X}|\,\|_2 \geq \|\mathbf{X}\|_2$ holds true for any matrix $\mathbf{X}$ [116]. Then, we have that $\|\mathbf{P}^{1/2}\mathbf{A}\mathbf{P}^{-1/2}\|_2 < 1$, which implies $\mathbf{A} \in \mathcal{D}_{\mathrm{d}}$ due to the equivalence in (3.82).

We now prove the second implication of (3.21): assume that $\mathbf{A} \in \mathcal{D}_{\mathrm{d}}$ and further assume that there exists an eigenpair $(\lambda, \mathbf{v})$ of $\mathbf{A}$ such that $|\lambda| \geq 1$. Then,

$$\mathbf{v}^{\mathrm{H}}\left(\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P}\right)\mathbf{v} = (|\lambda|^2 - 1)\;\mathbf{v}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{v} \geq 0, \tag{3.84}$$

which contradicts with the assumption that $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P}$ is negative definite. Thus, $\rho(\mathbf{A}) < 1$ must hold true.

3.7.3 Proof of Theorem 3.2

In what follows $\mathbf{I}_m$ denotes the $m \times m$ identity matrix. Since the state variables evolves according to (3.46) in the synchronous case, we can write the following due to (3.6):

$$\mathbb{E}\big[\bar{\mathbf{r}}_k\,\bar{\mathbf{r}}_k^{\mathrm{H}}\big] = \bar{\mathbf{A}}^k\;\bar{\mathbf{r}}_0\;\bar{\mathbf{r}}_0^{\mathrm{H}}\;(\bar{\mathbf{A}}^k)^{\mathrm{H}} + \sum_{n=0}^{k-1} \bar{\mathbf{A}}^n\;\bar{\mathbf{\Gamma}}\;(\bar{\mathbf{A}}^n)^{\mathrm{H}}, \tag{3.85}$$

where we define $\bar{\mathbf{r}}_k = \bar{\mathbf{x}}_k - \bar{\mathbf{x}}^\star$ similar to (3.5). Here $\bar{\mathbf{x}}^\star$ denotes the fixed point of the vectorized model in (3.46) (which exists since $\bar{\mathbf{A}}$ is assumed to not have eigenvalue 1), and it can be written as follows:

$$\begin{aligned}
\bar{\mathbf{x}}^\star &= \big(\mathbf{I}_{LN} - \bar{\mathbf{A}}\big)^{-1}\;\bar{\mathbf{u}} = \big(\mathbf{I}_{LN} - (\mathbf{A}\otimes\mathbf{G})\big)^{-1}\,(\mathbf{b}\otimes\mathbf{u}) \\
&= (\mathbf{T}^{-1}\otimes\mathbf{I}_N)\;(\mathbf{I}_{LN} - \widehat{\mathbf{A}}\otimes\mathbf{G})^{-1}\;(\widehat{\mathbf{b}}\otimes\mathbf{u}) \qquad (3.86) \\
&= (\mathbf{T}^{-1}\otimes\mathbf{I}_N)\;\mathrm{vec}\Big(\big[\mathbf{G}^{L-1}\mathbf{z} \quad \cdots \quad \mathbf{G}\,\mathbf{z} \quad \mathbf{z}\big]\Big), \qquad (3.87)
\end{aligned}$$

where the vector $\mathbf{z} \in \mathbb{C}^N$ is defined as follows:

$$\mathbf{z} = q(\mathbf{G})^{-1}\,\mathbf{u}. \tag{3.88}$$

We note that the equivalence between (3.86) and (3.87) follows from the following identity:

$$\begin{bmatrix} \mathbf{I} & -\mathbf{G} & 0 & \cdots & 0 \\ 0 & \mathbf{I} & -\mathbf{G} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\mathbf{G} \\ q_L\mathbf{G} & q_{L-1}\mathbf{G} & \cdots & q_2\mathbf{G} & \mathbf{I}+q_1\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{G}^{L-1}\,\mathbf{z} \\ \mathbf{G}^{L-2}\,\mathbf{z} \\ \vdots \\ \mathbf{G}\,\mathbf{z} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ q(\mathbf{G})\,\mathbf{z} \end{bmatrix} \tag{3.89}$$

which can be written as follows:

$$(\mathbf{I}_{LN} - \widehat{\mathbf{A}} \otimes \mathbf{G})\,\mathrm{vec}\Big(\big[\mathbf{G}^{L-1}\,\mathbf{z} \quad \cdots \quad \mathbf{G}\,\mathbf{z} \quad \mathbf{z}\big]\Big) = \widehat{\mathbf{b}} \otimes \mathbf{u}, \tag{3.90}$$

where we use the fact that $q(\mathbf{G})\,\mathbf{z} = \mathbf{u}$.

Line 9 of the algorithm together with the result of (3.40) shows that the output vector $\mathbf{y}_{(k)}$ defined in (3.38) is related to the vectorized state variables as follows:

$$\mathbf{y}_{(k)} = \mathbf{G}\,\mathbf{X}^{\mathrm{T}}_{(k-1)}\mathbf{c}^{\mathrm{T}} + d\,\mathbf{u}_{(k)} = (\mathbf{c} \otimes \mathbf{G})\,\bar{\mathbf{x}}_{k-1} + d\,\mathbf{u} + d\,\mathbf{w}_{(k)}. \tag{3.91}$$

Furthermore, the fixed point of the vectorized model given in (3.87) satisfies the following equality:

$$\begin{aligned} (\mathbf{c} \otimes \mathbf{G})\,\bar{\mathbf{x}}^{\star} + d\,\mathbf{u} &= (\widehat{\mathbf{c}} \otimes \mathbf{G}) \begin{bmatrix} \mathbf{G}^{L-1}\,\mathbf{z} \\ \vdots \\ \mathbf{G}\,\mathbf{z} \\ \mathbf{z} \end{bmatrix} + d\,\mathbf{u} \\ &= \sum_{n=1}^{L} (p_n - p_0\,q_n)\,\mathbf{G}^n\,\mathbf{z} + p_0\,q(\mathbf{G})\,\mathbf{z} = \sum_{n=0}^{L} p_n\,\mathbf{G}^n\,\mathbf{z} \\ &= p(\mathbf{G})\,\mathbf{z} = p(\mathbf{G})\,q(\mathbf{G})^{-1}\,\mathbf{u} = \widetilde{\mathbf{u}}. \end{aligned} \tag{3.92}$$

Combining (3.91) and (3.92), we can write the following:

$$\mathbf{y}_{(k)} - \widetilde{\mathbf{u}} = (\mathbf{c} \otimes \mathbf{G})(\bar{\mathbf{x}}_{k-1} - \bar{\mathbf{x}}^{\star}) + d\,\mathbf{w}_{(k)}. \tag{3.93}$$

Then,

$$\mathbb{E}\left[\|\mathbf{y}_{(k)}-\widetilde{\mathbf{u}}\|_2^2\right] = \operatorname{tr}\left((\mathbf{c}\otimes\mathbf{G})\,\mathbb{E}\left[\bar{\mathbf{r}}_{k\text{-}1}\,\bar{\mathbf{r}}_{k\text{-}1}^{\mathrm{H}}\right](\mathbf{c}\otimes\mathbf{G})^{\mathrm{H}}\right) + |d|^2\,\operatorname{tr}(\boldsymbol{\Gamma}), \tag{3.94}$$
$$= \left\|(\mathbf{c}\otimes\mathbf{G})\,\bar{\mathbf{A}}^{k\text{-}1}\,\bar{\mathbf{r}}_0\right\|_2^2 + |d|^2\,\operatorname{tr}(\boldsymbol{\Gamma}) + \sum_{n=0}^{k-2}\operatorname{tr}\left((\mathbf{c}\otimes\mathbf{G})\bar{\mathbf{A}}^n\,\bar{\boldsymbol{\Gamma}}\,(\bar{\mathbf{A}}^n)^{\mathrm{H}}\,(\mathbf{c}\otimes\mathbf{G})^{\mathrm{H}}\right),$$

where we use the result in (3.85).

Due to (3.44), we note that $\bar{\mathbf{A}}^n = (\mathbf{A}\otimes\mathbf{G})^n = \mathbf{A}^n\otimes\mathbf{G}^n$. Furthermore, the augmented noise covariance matrix $\bar{\boldsymbol{\Gamma}}$ can be written explicitly from (3.44) as follows:

$$\bar{\boldsymbol{\Gamma}} = \mathbb{E}\left[\bar{\mathbf{w}}_k\,\bar{\mathbf{w}}_k^{\mathrm{H}}\right] = \mathbb{E}\left[(\mathbf{b}\otimes\mathbf{w}_k)\,(\mathbf{b}\otimes\mathbf{w}_k)^{\mathrm{H}}\right] = \mathbf{b}\mathbf{b}^{\mathrm{H}}\otimes\boldsymbol{\Gamma}. \tag{3.95}$$

Thus,

$$(\mathbf{c}\otimes\mathbf{G})\bar{\mathbf{A}}^n\,\bar{\boldsymbol{\Gamma}}\,(\bar{\mathbf{A}}^n)^{\mathrm{H}}(\mathbf{c}\otimes\mathbf{G})^{\mathrm{H}} = |\mathbf{c}\,\mathbf{A}^n\,\mathbf{b}|^2\,\mathbf{G}^{n+1}\,\boldsymbol{\Gamma}\,(\mathbf{G}^{n+1})^{\mathrm{H}}. \tag{3.96}$$

We also note that the coefficients of the impulse response of the underlying digital filter is given as follows [200, Eq. (13.4.13)]:

$$h_n = \begin{cases} d, & n = 0,\\ \mathbf{c}\;\mathbf{A}^{n\text{-}1}\;\mathbf{b}, & n > 0. \end{cases} \tag{3.97}$$

Using (3.97) in (3.96), we can re-write (3.94) as follows:

$$\mathbb{E}\left[\|\mathbf{y}_{(k)}-\widetilde{\mathbf{u}}\|_2^2\right] = \left\|(\mathbf{c}\otimes\mathbf{G})\,\bar{\mathbf{A}}^{k\text{-}1}\,\bar{\mathbf{r}}_0\right\|_2^2 + |h_0|^2\,\operatorname{tr}(\boldsymbol{\Gamma}) + \sum_{n=0}^{k\text{-}2}|h_{n+1}|^2\operatorname{tr}\left(\mathbf{G}^{n+1}\,\boldsymbol{\Gamma}\,(\mathbf{G}^{n+1})^{\mathrm{H}}\right), \tag{3.98}$$

which is equivalent to the result in (3.47).

3.7.4 Proof of Theorem 3.3

Let $\bar{\mathbf{P}} \in \mathbb{R}^{LN\times LN}$ denote the average index-selection matrix for the vectorized model in (3.43). Then, the structure of the update sets in (3.45) imply the following:

$$\bar{\mathbf{P}} = \mathbf{I}_L\otimes\mathbf{P}, \tag{3.99}$$

where $\mathbf{I}_L$ denotes the identity matrix of size L.

We now show that the following holds true:

$$\bar{\mathbf{A}}^{\mathrm{H}}\;\bar{\mathbf{P}}\;\bar{\mathbf{A}} \prec \bar{\mathbf{P}}, \tag{3.100}$$

which also ensures that $\bar{\mathbf{A}}$ is a stable matrix, hence the fixed point $\bar{\mathbf{x}}^\star$ computed in (3.87) exists. In this regard, using (3.44), (3.99), and the mixed-product property in (1.4), we can write (3.100) explicitly as follows:

$$(\mathbf{A}^{\mathrm{H}}\,\mathbf{A}) \otimes (\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G}) \prec (\mathbf{I}_L \otimes \mathbf{P}). \tag{3.101}$$

Let $\mathbf{V}$ denote the eigenvectors of $\mathbf{A}^{\mathrm{H}}\mathbf{A}$. By left-multiplying with $\mathbf{V}^{\mathrm{H}} \otimes \mathbf{I}_N$ and right-multiplying with $\mathbf{V} \otimes \mathbf{I}_N$, the condition (3.101) can be written equivalently as follows:

$$\boldsymbol{\Sigma}_{\mathbf{A}}^2 \otimes (\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G}) \prec (\mathbf{I}_L \otimes \mathbf{P}), \tag{3.102}$$

where $\boldsymbol{\Sigma}_{\mathbf{A}}$ is a diagonal matrix consisting of the singular values of the matrix $\mathbf{A}$. Since both sides of (3.102) are block diagonal matrices, the condition (3.102) holds true if and only if all of the individual blocks satisfy the inequality, that is,

$$\sigma_i^2(\mathbf{A})\;\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G} \prec \mathbf{P} \tag{3.103}$$

for all singular values, $\sigma_i(\mathbf{A})$, of the matrix $\mathbf{A}$, which can be expressed explicitly as follows:

$$\sigma_{\min}^2(\mathbf{A})\;\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G} \;\;\leq\;\; \cdots \;\;\leq\;\; \sigma_{\max}^2(\mathbf{A})\;\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G} \;\;\prec\;\; \mathbf{P}. \tag{3.104}$$

Since (3.51) is both necessary and sufficient to satisfy (3.104), we conclude that (3.51) is equivalent to the condition in (3.100).

Since the assumption (3.51) ensures that (3.100) is satisfied, we can apply Corollary 3.1 to the model in (3.43) and conclude that

$$\lim_{k\to\infty} \mathbb{E}\big[\, \|\bar{\mathbf{x}}_k - \bar{\mathbf{x}}^\star\|_2^2 \,\big] \leq \frac{\operatorname{tr}(\bar{\mathbf{P}}\,\bar{\boldsymbol{\Gamma}})}{\lambda_{\min}\Big(\bar{\mathbf{P}} - \bar{\mathbf{A}}^{\mathrm{H}}\,\bar{\mathbf{P}}\,\bar{\mathbf{A}}\Big)}. \tag{3.105}$$

Using the equality (3.93) from Section 3.7.3:

$$\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\|_2^2 \leq \|\mathbf{c}\|_2^2\;\|\mathbf{G}\|_2^2\;\|\bar{\mathbf{x}}_{k\text{-}1} - \bar{\mathbf{x}}^\star\|_2^2 + \|d\,\mathbf{w}_{(k)}\|_2^2, \tag{3.106}$$

which results in the following:

$$\lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{y}_{(k)} - \widetilde{\mathbf{u}}\|_2^2\big] \leq \frac{\|\mathbf{c}\|_2^2\;\|\mathbf{G}\|_2^2\;\operatorname{tr}(\bar{\mathbf{P}}\,\bar{\boldsymbol{\Gamma}})}{\lambda_{\min}\Big(\bar{\mathbf{P}} - \bar{\mathbf{A}}^{\mathrm{H}}\,\bar{\mathbf{P}}\,\bar{\mathbf{A}}\Big)} + |d|^2\;\operatorname{tr}(\boldsymbol{\Gamma}). \tag{3.107}$$

We have the following due to (3.95) of Section 3.7.3 and (3.99):

$$\operatorname{tr}(\bar{\mathbf{P}}\,\bar{\boldsymbol{\Gamma}}) = \operatorname{tr}\Big(\mathbf{b}\mathbf{b}^{\mathrm{H}} \otimes \mathbf{P}\boldsymbol{\Gamma}\Big) = \|\mathbf{b}\|_2^2\;\operatorname{tr}(\mathbf{P}\,\boldsymbol{\Gamma}). \tag{3.108}$$

We also note that (3.101) and (3.104) implies the following:

$$\lambda_{\min}\left(\bar{\mathbf{P}} - \bar{\mathbf{A}}^{\mathrm{H}}\,\bar{\mathbf{P}}\,\bar{\mathbf{A}}\right) = \lambda_{\min}\left(\mathbf{P} - \|\mathbf{A}\|_2^2\,\mathbf{G}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{G}\right). \tag{3.109}$$

The use of (3.108) and (3.109) in (3.107) gives the desired result.

Chapter 4

RANDOM ASYNCHRONOUS LINEAR SYSTEMS: FREQUENCY RESPONSE AND MEAN-SQUARED STABILITY

4.1 Introduction

Linear time-invariant discrete-time systems are well studied mathematical models that find applications in wide range of different areas ranging from mathematical finance to implementation of digital filters [94, 33, 213, 200, 95]. Although such models are especially useful for analyzing (and controlling) dynamical systems that evolve in time, the mathematical models are useful in numerical linear algebra problems as well. An example is the "power method" that can extract the dominant eigenvector of the transition matrix, whose notable application is the PageRank algorithm used in search engines for ranking web pages [139].

State-space models are studied also in the field of graph signal processing [160, 151], in which the state transition matrix is assumed to model the underlying graph structure and the state variables are interpreted as the signals held by the nodes of the graph. In this setting an iteration of the state-space model is equivalent to the nodes communicating with their neighbors. With this formalism, state recursions are utilized for distributed implementation of polynomial (FIR) graph filters [158, 149, 150, 154, 172, 173] as well as rational (IIR) graph filters [157, 87, 88, 108].

4.1.1 Contributions of the Chapter

Different from Chapters 2 and 3, this chapter explores the behavior of random asynchronous state-space models when the input is nonzero and time dependent, such as exponentials and sinusoids. This is done without any a priori assumptions on the state transition matrix (such as symmetry, normalcy, and so forth), or on the update probabilities. In particular, this chapter investigates the notion of "frequency response" by showing that exponentials continue to be eigenfunctions of linear random asynchronous systems in a *statistical sense*. So, a random asynchronous system can be treated as a time-invariant system in an average sense, despite its randomly time-varying behavior. The mean-squared stability of random asynchronous systems is also studied in detail, and it is shown that an unstable system (in the synchronous world) may get stable with randomized asynchronicity. However, randomized asynchronicity introduces an error into the state variables depending on the amount of variation in the input signal as well as the update probabilities. This chapter also draws parallels between random asynchronous systems and Markov jump linear systems, which were studied rigorously in [42].

This chapter studies random asynchronous recursions from a linear system theory viewpoint. Although the focus here is not on any specific application, random asynchronous recursions have a number of applications, especially in the area of graph signal processing. For example, in the implementation of graph filters (which requires nodes to communicate only with their neighbors), nodes can also interact with their neighbors asynchronously and still obtain convergence to the desired filtering output under mild assumptions (see Section 3.3). Another network related application studies asynchronous state recursions for the computation of the Fiedler vector of the graph in order to obtain spectral clustering with asynchronous computations (see Section 2.7). Since the results presented here consider time-varying input signals, it is possible to construct asynchronous graph filters that achieve spatial (over the network) filtering and time-domain filtering simultaneously. Finally, random asynchronous recursions can also be viewed as a randomized component-wise power method, which can be re-formulated to compute the singular vectors of a given data matrix (see Section 2.8).

4.1.2 Outline

Section 4.2.1 introduces the randomized model considered in this chapter, and Section 4.2.2 presents the expected behavior of the randomized system (Theorem 4.1). The "frequency response or transfer function in the mean" is introduced here. Section 4.3.1 describes how the second order statistics of the state vector evolve (Theorem 4.2), and Section 4.3.2 provides the necessary and sufficient condition for the mean-squared stability of the randomized recursions (Corollary 4.2). Such stability is essential in order for the "frequency response in the mean" to be useful in practice. Section 4.3.3 considers the random asynchronous updates from the viewpoint of Markov jump systems, and Section 4.3.4 provides a simple sufficiency condition for the mean-squared stability (Corollary 4.3). Section 4.4 shows that stability conditions in the synchronous and asynchronous settings do not imply each other in general (Lemma 4.4), and the stability in the asynchronous case cannot be determined from the eigenvalues of the state transition matrix alone (Lemma 4.5). Section 4.5 compares the conditions for the convergence of the random and non-random asynchronous fixed-point iterations with constant input. Further insights and discussions are given in Section 4.6.

The content of this chapter is mainly drawn from [180], and parts of it have been presented in [182].

4.2 Asynchronous Linear Systems with Exponential Inputs

Consider a discrete time-invariant system with R inputs, P outputs, and N state variables, whose state-space description is given as follows:

$$\mathbf{x}_{k+1} = \mathbf{A}\,\mathbf{x}_k + \mathbf{B}\,\mathbf{u}_k + \mathbf{w}_k, \tag{4.1}$$

$$\mathbf{y}_k = \mathbf{C}\,\mathbf{x}_k + \mathbf{D}\,\mathbf{u}_k, \tag{4.2}$$

where $\mathbf{x}_0 \in \mathbb{C}^N$ is the initial state vector (initial condition), and $\mathbf{w}_k \in \mathbb{C}^N$ is the noise term with the following statistics:

$$\mathbb{E}[\mathbf{w}_k] = \mathbf{0}, \qquad \mathbb{E}\big[\mathbf{w}_k\ \mathbf{w}_s^{\mathrm{H}}\,\big] = \delta(k-s)\ \mathbf{\Gamma}, \tag{4.3}$$

where $\delta(\cdot)$ denotes the discrete Dirac delta function, and $\mathbf{\Gamma}$ is allowed to be non-diagonal.

The matrices in the state-space model in (4.1) have the following dimensions:

$$\mathbf{A} \in \mathbb{C}^{N\times N}, \qquad \mathbf{B} \in \mathbb{C}^{N\times R}, \qquad \mathbf{C} \in \mathbb{C}^{P\times N}, \qquad \mathbf{D} \in \mathbb{C}^{P\times R}, \tag{4.4}$$

where $\mathbf{A}$ is referred to as the *state-transition matrix*, and the columns of the matrices $\mathbf{B}$ and $\mathbf{D}$ will be denoted as follows:

$$\mathbf{B} = [\mathbf{B}_1 \ \cdots \ \mathbf{B}_R], \qquad \mathbf{D} = [\mathbf{D}_1 \ \cdots \ \mathbf{D}_R]. \tag{4.5}$$

We further assume that the input signal $\mathbf{u}_k$ consists of R exponential signals in the following form:

$$\mathbf{u}_k = \begin{bmatrix}\alpha_1^k & \cdots & \alpha_R^k\end{bmatrix}^{\mathrm{T}}, \tag{4.6}$$

where α_i's are assumed to be distinct without loss of generality. Furthermore, we always assume that

$$|\alpha_i| \leq 1, \qquad \forall\, 1 \leq i \leq R, \tag{4.7}$$

so that $\mathbf{u}_k$ stays bounded throughout the iterations. While exponential inputs may seem restrictive, they form the basis for more general practical signals, making this study useful.

In the noise free case, i.e., $\mathbf{\Gamma} = \mathbf{0}$, it is well-known from linear system theory that the output vector $\mathbf{y}_k \in \mathbb{C}^P$ in (4.2) can be written as follows:

$$\mathbf{y}_k = \mathbf{y}_k^{\mathrm{ss}} + \mathbf{y}_k^{\mathrm{tr}}, \tag{4.8}$$

where $\mathbf{y}_k^{\text{ss}}$ denotes the steady-state component, and $\mathbf{y}_k^{\text{tr}}$ denotes the transient component that are given as follows:

$$\mathbf{y}_k^{\text{ss}} = \sum_{i=1}^{R} \mathbf{H}_i(\alpha_i)\ \alpha_i^k, \qquad \mathbf{y}_k^{\text{tr}} = \mathbf{C}\,\mathbf{A}^k\left(\mathbf{x}_0 - \mathbf{x}_0^{\text{ss}}\right). \tag{4.9}$$

where $\mathbf{H}_i(z) \in \mathbb{C}^P$ denotes the transfer function that relates the i^{th} input to the output, which is given as follows:

$$\mathbf{H}_i(z) = \mathbf{D}_i + \mathbf{C}\left(z\,\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B}_i. \tag{4.10}$$

We also note that the term $\mathbf{x}_0^{\text{ss}}$ in (4.9) is given as follows:

$$\mathbf{x}_0^{\text{ss}} = \sum_{i=1}^{R}\left(\alpha_i\,\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B}_i. \tag{4.11}$$

It is clear from (4.9) that when the state-transition matrix $\mathbf{A}$ is a stable matrix, i.e., the following holds true:

$$\rho(\mathbf{A}) < 1, \tag{4.12}$$

then the transient component $\mathbf{y}_k^{\text{tr}}$ converges to zero as the iterations progress leaving only the steady-state component $\mathbf{y}_k^{\text{ss}}$ in the output signal. In fact, stability of $\mathbf{A}$ is also necessary for the transient part to converge to zero, which is a well-known result from linear system theory [95, 200].

In this chapter we consider the following randomized model:

$$(\mathbf{x}_{k+1})_i = \begin{cases} \left(\mathbf{A}\,\mathbf{x}_k + \mathbf{B}\,\mathbf{u}_k + \mathbf{w}_k\right)_i, & i \in \mathcal{T}_{k+1}, \\ (\mathbf{x}_k)_i, & i \notin \mathcal{T}_{k+1}, \end{cases} \tag{4.13}$$

$$\mathbf{y}_k = \mathbf{C}\,\mathbf{x}_k + \mathbf{D}\,\mathbf{u}_k, \tag{4.14}$$

where $\mathcal{T}_k$ denotes the set of indices updated at the k^{th} iteration. The model (4.13) is very similar to the standard synchronous recursions in (4.2) except the fact that only a random subset of indices (denoted by $\mathcal{T}_k$) are updated in every iteration, and the remaining indices stay the same. The specific stochastic model regarding the selection of the indices will be elaborated next.

4.2.1 Random Selection of the Update Sets

In the asynchronous model considered in (4.13), we assume that the i^{th} index (state variable) is updated independently with probability p_i in every iteration. That is,

$$\mathbb{P}[i \in \mathcal{T}_k] = p_i \qquad \forall k, \tag{4.15}$$

where p_i will be referred to as *the update probability of the i^{th} index*. We will use $\mathbf{P}$ to denote the diagonal matrix consisting of the index selection probabilities. More precisely,

$$\mathbf{P} = \text{diag}\Big([p_1 \quad p_2 \quad \cdots \quad p_N]\Big). \tag{4.16}$$

It is assumed that $\mathbf{P}$ satisfies $\mathbf{0} \prec \mathbf{P} \preceq \mathbf{I}$, where the positive definiteness follows from the fact that no index should be left out permanently during the updates of (4.13). See Section 4.4 for further details. Additionally, $\text{tr}(\mathbf{P}) = \mathbb{E}[|\mathcal{T}_k|]$ denotes the number of indices updated per iteration on average.

4.2.2 Frequency Response in the Mean

Due to the random updates of the state variables it is clear from (4.13) that the state vector $\mathbf{x}_k$ is a random vector. So, the state vector will not have an exponential behavior exactly even when the input is a simple exponential ($R = 1$ in (4.6)). Nevertheless, we will show that $\mathbf{x}_k$ still behaves like a sum of exponential signals *in a statistical sense*:

Theorem 4.1. *Assume that the randomized asynchronous state recursions in* (4.13) *are initialized independently and randomly. Then, the expectation of the state vector in* (4.13) *is as follows:*

$$\mathbb{E}[\mathbf{x}_k] = \mathbf{x}_k^{\text{ss}} + \mathbf{x}_k^{\text{tr}}, \tag{4.17}$$

where

$$\mathbf{x}_k^{\text{ss}} = \sum_{i=1}^{R} (\alpha_i \, \mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{B}}_i \, \alpha_i^k, \qquad \mathbf{x}_k^{\text{tr}} = \bar{\mathbf{A}}^k \big(\mathbb{E}[\mathbf{x}_0] - \mathbf{x}_0^{\text{ss}}\big), \tag{4.18}$$

and

$$\bar{\mathbf{A}} = \mathbf{I} + \mathbf{P}\,(\mathbf{A} - \mathbf{I}), \qquad \bar{\mathbf{B}}_i = \mathbf{P}\,\mathbf{B}_i. \tag{4.19}$$

Proof. See Section 4.8.2. □

Here, $\bar{\mathbf{A}}$ will be referred to as *the average state-transition matrix*, and $\bar{\mathbf{B}}$ *the average input matrix*. We can also represent $\mathbf{x}_k^{\text{ss}}$ in (4.18) as follows:

$$\mathbf{x}_k^{\text{ss}} = \sum_{i=1}^{R} \mathbf{x}_{0,i}^{\text{ss}} \, \alpha_i^k \quad \text{where} \quad \mathbf{x}_{0,i}^{\text{ss}} = (\alpha_i \, \mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{B}}_i, \tag{4.20}$$

which will be useful later in the chapter.

As an immediate corollary to Theorem 4.1, we present the following result regarding the expected behavior of the output:

Corollary 4.1. *Assume that the randomized asynchronous state recursions in* (4.13) *are initialized independently and randomly. Then, the expectation of the output in* (4.14) *is as follows:*

$$\mathbb{E}[\mathbf{y}_k] = \mathbf{y}_k^{\text{ss}} + \mathbf{y}_k^{\text{tr}}, \tag{4.21}$$

where

$$\mathbf{y}_k^{\text{ss}} = \sum_{i=1}^{R} \bar{\mathbf{H}}_i(\alpha_i)\ \alpha_i^k, \qquad \mathbf{y}_k^{\text{tr}} = \mathbf{C}\,\bar{\mathbf{A}}^k\ \big(\mathbb{E}[\mathbf{x}_0] - \mathbf{x}_0^{\text{ss}}\big), \tag{4.22}$$

and

$$\bar{\mathbf{H}}_i(z) = \mathbf{D}_i + \mathbf{C}\ \big(z\,\mathbf{I} - \bar{\mathbf{A}}\big)^{-1}\ \bar{\mathbf{B}}_i. \tag{4.23}$$

This can, therefore, be regarded as the "transfer function in the mean" from the i^{th} input node to the output.

Proof. Due to (4.6), (4.14), and Theorem 4.1, the expectation of $\mathbf{y}_k$ can be written as follows:

$$\mathbb{E}[\mathbf{y}_k] = \mathbf{C}\,\mathbb{E}[\mathbf{x}_k] + \sum_{i=1}^{R} \mathbf{D}_i\,\alpha_i^k = \mathbf{y}_k^{\text{ss}} + \mathbf{y}_k^{\text{tr}}, \tag{4.24}$$

where $\mathbf{y}_k^{\text{ss}}$ and $\mathbf{y}_k^{\text{tr}}$ are as in (4.22). □

Regarding the form in (4.21) we first note that the terms $\mathbf{y}_k^{\text{ss}}$ and $\mathbf{y}_k^{\text{tr}}$ are deterministic quantities, and the expectation is with respect to the random selection of the indices, the input noise, and the random selection of the initial condition.

Corollary 4.1 shows that the random output vector $\mathbf{y}_k$ behaves the same as its deterministic counterpart (4.8) in expectation. That is, $\mathbb{E}[\mathbf{y}_k]$ can be decomposed into steady-state and transient parts similar to (4.8). Therefore, the quantity $\bar{\mathbf{H}}_i(z)$ given in (4.23) can be regarded as the "transfer function" from the i^{th} input to the output in the *expectation sense*.

It is clear from (4.22) that as long as the average state transition matrix $\bar{\mathbf{A}}$ is stable, i.e., the following holds true:

$$\rho(\bar{\mathbf{A}}) < 1, \tag{4.25}$$

the component $\mathbf{y}_k^{\text{tr}}$ converges to zero irrespective of the observation matrix $\mathbf{C}$ and the statistics of the initial condition. Thus, the condition (4.25) is both necessary and sufficient for $\mathbb{E}[\mathbf{y}_k]$ to behave like a sum of exponentials, that is,

$$\lim_{k\to\infty} \mathbb{E}\big[\mathbf{y}_k - \mathbf{y}_k^{\text{ss}}\big] = \mathbf{0}. \tag{4.26}$$

This shows that when (4.25) is satisfied an exponential input results in an exponential output *in expectation* even with the randomized asynchronous state recursions.

4.2.3 A Running Numerical Example

In order to demonstrate the behavior of the random vector $\mathbf{y}_k$, we consider the following state-space model with $N = 4$ state variables, $R = 1$ input, and $P = 1$ output:

$$\mathbf{A} = \frac{1}{10}\begin{bmatrix} -4 & -1 & 2 & -6 \\ 4 & -6 & -5 & 3 \\ 2 & -2 & 7 & 2 \\ 5 & 9 & -3 & 1 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 \\ 4 \\ 2 \\ 3 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}^{\mathrm{T}}, \qquad \mathbf{D} = 0, \tag{4.27}$$

where we point out that the matrix $\mathbf{A}$ is *not* stable since $\rho(\mathbf{A}) \approx 1.0441$. As we shall discuss later in Section 4.3, stability of the randomized asynchronous state recursions does not require stability of the state-transition matrix in general.

In the following numerical example we assume that $\mathbf{P} = p\,\mathbf{I}$, i.e., all nodes have the update probability p and assume that $\boldsymbol{\Gamma} = \mathbf{0}$. Furthermore, we assume that the input signal has the following complex exponential form:

$$\alpha = e^{j2\pi/100} \qquad \Longrightarrow \qquad \mathbf{u}_k = e^{j2\pi k/100}. \tag{4.28}$$

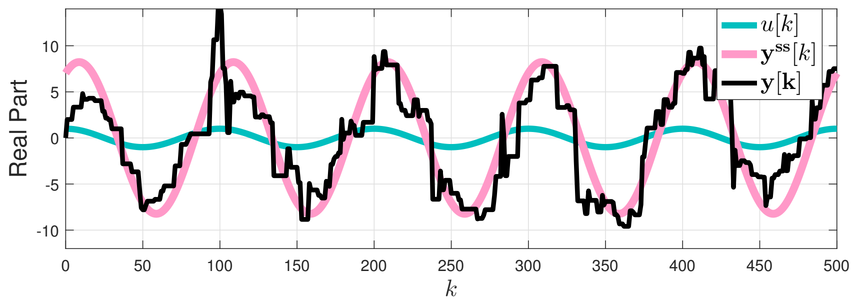
In Figure 4.1 we visualize *a realization* of the output signal $\mathbf{y}_k$ together with the steady-state component $\mathbf{y}_k^{\mathrm{ss}}$ as well as the input signal u_k for three different update probabilities, namely, $p \in \{0.1,\ 0.3,\ 0.6\}$. The figure shows only the real part of the signals for convenience.

From Figure 4.1 it is clear that the random vector $\mathbf{y}_k$ is not a complex exponential in a strict sense, yet it "behaves like" one. We also note that $\mathbf{y}_k$ has the same "frequency" as the input signal irrespective of the update probabilities.

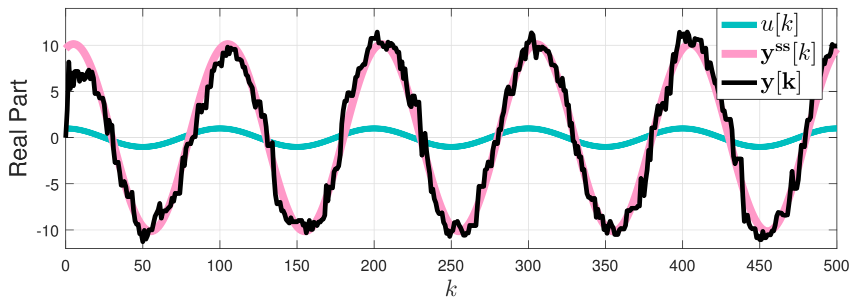
Figure 4.1 shows also that the response of the random asynchronous system depends on the update probabilities, which is also apparent from the expression in (4.23). In fact, the response of the random asynchronous updates running on a system denoted with $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ can be represented in an average sense as the response of a synchronous system $(\bar{\mathbf{A}}, \bar{\mathbf{B}}, \mathbf{C}, \mathbf{D})$. As a result, even when a single state variable updates its value with a different probability, the response of the overall system changes. In short, different update probabilities result in different "frequency responses" while leaving the output frequency unchanged.

4.3 Second Order Stability of the State Variables

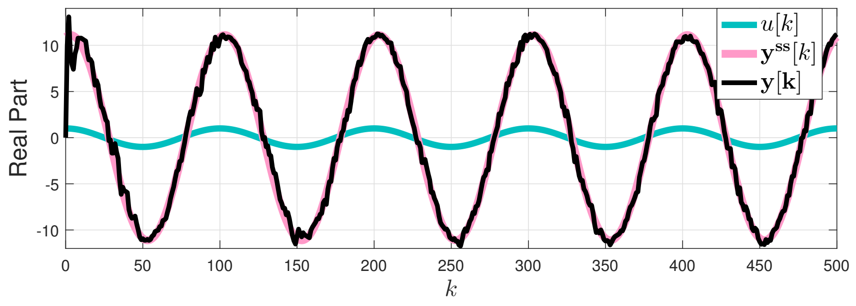
Corollary 4.1 together with Figure 4.1 show that the random vector $\mathbf{y}_k$ behaves like $\mathbf{y}_k^{\mathrm{ss}}$ in expectation. However, in order to interpret the response of the randomized



(a)



(b)



(c)

Figure 4.1: A realization of the output signal with the state-space model in (4.27), the frequency in (4.28), and the probabilities (a) $p = 0.1$, (b) $p = 0.3$, and (c) $p = 0.6$.

asynchronous system meaningfully, the random variable $\mathbf{y}_k$ must be ensured to have a “finite amount of deviation” from $\mathbf{y}_k^{\text{ss}}$. In this regard we consider the following quantities:

$$\mathbf{r}_k = \mathbf{y}_k - \mathbf{y}_k^{\text{ss}}, \qquad \mathbf{q}_k = \mathbf{x}_k - \mathbf{x}_k^{\text{ss}}, \tag{4.29}$$

where $\mathbf{r}_k$ will be referred to as the error in the output, and $\mathbf{q}_k$ will be referred to as the error in the state variables. It is readily verified that the error terms in (4.29) are related with each other through the output matrix $\mathbf{C}$ as follows:

$$\mathbf{r}_k = \mathbf{C}\,\mathbf{q}_k. \tag{4.30}$$

In the rest of this section we will focus on the term $\mathbf{q}_k$, i.e., study the internal stability of the random asynchronous system. More precisely, we consider *the error (auto) correlation matrix of the state variables* defined as follows:

$$\mathbf{Q}_k = \mathbb{E}\big[\mathbf{q}_k\,\mathbf{q}_k^{\text{H}}\big] \in \mathbb{C}^{N\times N}. \tag{4.31}$$

At this point it is very important to emphasize that the error correlation matrix $\mathbf{Q}_k$ does not converge to zero in general even when no noise present in the system. More interestingly, $\mathbf{Q}_k$ may not converge to a point at all; rather, it shows an oscillatory behavior. This is an inherent side effect of the randomized asynchronicity, which will be discussed in detail. As a result, we will consider the conditions under which $\mathbf{Q}_k$ stays bounded (or, equivalently $\mathbf{q}_k$ stays bounded in the mean-squared sense). Here, “second order stability” is synonymous to “boundedness of the matrix $\mathbf{Q}_k$.”

We also note that stability of the matrix $\bar{\mathbf{A}}$ merely ensures the first order stability of the error term. That is,

$$\rho(\bar{\mathbf{A}}) < 1 \qquad \Longleftrightarrow \qquad \lim_{k\to\infty} \mathbb{E}[\mathbf{q}_k] = \mathbf{0}. \tag{4.32}$$

On the other hand, stability of $\bar{\mathbf{A}}$ is not sufficient to ensure the second order stability of $\mathbf{q}_k$. (See Lemma 4.3 in Section 4.4.1).

In what follows, we will first study how the error correlation matrix $\mathbf{Q}_k$ evolves throughout the iterations (Theorem 4.2). Based on this result, we will consider the necessary and sufficient condition for the boundedness of the error correlation matrix (Corollary 4.2). Later, we will consider the updates from the Markovian jump system view point, and then we will provide a sufficiency condition for the second order stability.

4.3.1 Evolution of the Error Correlation Matrix

We start by consider the following matrix valued function:

$$\varphi(\mathbf{X}) = \bar{\mathbf{A}}\,\mathbf{X}\,\bar{\mathbf{A}}^{\mathrm{H}} + \Big((\mathbf{A}-\mathbf{I})\,\mathbf{X}\,(\mathbf{A}^{\mathrm{H}}-\mathbf{I})\Big) \odot \big(\mathbf{P}-\mathbf{P}^2\big), \tag{4.33}$$

where $\bar{\mathbf{A}}$ is as in (4.19), and $\odot$ denotes the Hadamard product.

Note that the function $\varphi(\cdot)$ is defined through the state-transition matrix $\mathbf{A}$ as well as the update probabilities $\mathbf{P}$. More importantly, $\varphi(\cdot)$ is a positive linear map. So, the following is readily verified to hold true (for any $\mathbf{A}$ and $\mathbf{P}$):

$$\mathbf{X} \geq \mathbf{0} \qquad \Longrightarrow \qquad \varphi(\mathbf{X}) \geq \mathbf{0}. \tag{4.34}$$

The importance of the function $\varphi(\cdot)$ follows from the fact that it governs the evolution of the error correlation matrix through the iterations. This is presented precisely as follows:

Theorem 4.2. *The error correlation matrix of the state variables evolves according to the following recursion:*

$$\mathbf{Q}_{k+1} = \varphi(\mathbf{Q}_k) + \mathbf{P}\,\boldsymbol{\Gamma}\,\mathbf{P} + \boldsymbol{\Gamma}\odot\big(\mathbf{P}-\mathbf{P}^2\big) + \Re\Big\{\big(\boldsymbol{\delta}_k + 2\,(\bar{\mathbf{A}}-\mathbf{I})\,\mathbf{x}_k^{\mathrm{tr}}\big)\,\boldsymbol{\delta}_k^{\mathrm{H}}\Big\} \odot \big(\mathbf{P}^{-1}-\mathbf{I}\big), \tag{4.35}$$

where $\Re\{\cdot\}$ denotes the real part of its argument, and the deterministic vector $\boldsymbol{\delta}_k \in \mathbb{C}^N$ is defined as follows:

$$\boldsymbol{\delta}_k = \mathbf{x}_{k+1}^{\mathrm{ss}} - \mathbf{x}_k^{\mathrm{ss}} = \sum_{i=1}^{R} (\alpha_i\,\mathbf{I} - \bar{\mathbf{A}})^{-1}\,\bar{\mathbf{B}}_i\,\alpha_i^k\,(\alpha_i - 1). \tag{4.36}$$

Proof. See Section 4.8.3. □

In order to study the behavior of the recursion in (4.35), we first represent the linear map $\varphi(\cdot)$ as a matrix-vector product by vectorizing (4.33). That is,

$$\varphi(\mathbf{X}) = \mathrm{vec}^{-1}\Big(\boldsymbol{\Phi}\ \mathrm{vec}(\mathbf{X})\Big), \tag{4.37}$$

and the matrix $\boldsymbol{\Phi} \in \mathbb{C}^{N^2\times N^2}$ is as follows:

$$\boldsymbol{\Phi} = \bar{\mathbf{A}}^* \otimes \bar{\mathbf{A}} + \Big((\mathbf{I}-\mathbf{P})\otimes\mathbf{P}\Big)\,\mathbf{J}\,\Big((\mathbf{A}^*-\mathbf{I})\otimes(\mathbf{A}-\mathbf{I})\Big), \tag{4.38}$$

where $\mathbf{A}^*$ denotes the element-wise conjugate (not conjugate transpose) of the matrix, and $\mathbf{J}$ is a diagonal matrix as follows:

$$\mathbf{J} = \sum_{i=1}^{N} (\mathbf{e}_i\,\mathbf{e}_i^{\mathrm{H}}) \otimes (\mathbf{e}_i\,\mathbf{e}_i^{\mathrm{H}}) \in \mathbb{R}^{N^2\times N^2}. \tag{4.39}$$

Since the error correlation matrix evolves according to $\varphi(\cdot)$ and $\mathbf{\Phi}$ is the matrix representation of the linear map, spectral properties of the matrix $\mathbf{\Phi}$ are very important in the behavior of the error correlation matrix $\mathbf{Q}_k$. In this regard, we first note that $\mathbf{\Phi}$ has complex eigenvalues in general. Secondly, the matrix $\mathbf{\Phi}$ always has a real nonnegative eigenvalue that is equal to its spectral radius, and the corresponding eigenvector is the vectorized version of a positive semi-define matrix. More precisely, it is always possible to find a nonzero $\mathbf{X} \geq \mathbf{0}$ such that the following holds true:

$$\varphi(\mathbf{X}) = \rho(\mathbf{\Phi})\, \mathbf{X}, \tag{4.40}$$

which follows from the extensions of the Perron-Frobenius theorem to positive maps in more general settings. We refer to [99, Theorem 5], or [61, Theorem 2.5] for the precise details.

4.3.2 The Necessary and Sufficient Condition

As a corollary to Theorem 4.2, we present the following result regarding the long term behavior of the error correlation matrix:

Corollary 4.2. *If the following holds true:*

$$\rho(\mathbf{\Phi}) < 1, \tag{4.41}$$

where the matrix $\mathbf{\Phi}$ *is as in* (4.38)*, then the following holds true regarding the error correlation matrix of the state variables:*

$$\lim_{k\to\infty} \left(\mathbf{Q}_k - \mathbf{Q}^{\mathrm{n}} - \mathbf{Q}^{\mathrm{r}}_k\right) = \mathbf{0}, \tag{4.42}$$

where the matrices $\mathbf{Q}^{\mathrm{n}},\ \mathbf{Q}^{\mathrm{r}}_k \in \mathbb{C}^{N\times N}$ *are given as the solution of the following linear matrix equations:*

$$\mathbf{Q}^{\mathrm{n}} = \varphi(\mathbf{Q}^{\mathrm{n}}) + \mathbf{P}\,\mathbf{\Gamma}\,\mathbf{P} + \mathbf{\Gamma} \odot \left(\mathbf{P} - \mathbf{P}^2\right). \tag{4.43}$$

$$\mathbf{Q}^{\mathrm{r}}_{k+1} = \varphi(\mathbf{Q}^{\mathrm{r}}_k) + \left(\boldsymbol{\delta}_k\, \boldsymbol{\delta}_k^{\mathrm{H}}\right) \odot \left(\mathbf{P}^{-1} - \mathbf{I}\right). \tag{4.44}$$

Conversely, if the condition (4.41) *is violated, then* $\mathbf{Q}_k$ *increases unboundedly as k goes to infinity.*

Proof. See Section 4.8.4. □

Regarding the limit in (4.42) it is important to note that the error correlation matrix $\mathbf{Q}_k$ does not converge to a point in general. So, $\lim_{k\to\infty} \mathbf{Q}_k$ may not exist. However, as long as the stability condition (4.41) is met, Corollary 4.2 shows that $\mathbf{Q}_k$

approaches the sum $\mathbf{Q}^{\mathrm{n}} + \mathbf{Q}^{\mathrm{r}}_k$, where $\mathbf{Q}^{\mathrm{n}}$ is the error due to the input noise and $\mathbf{Q}^{\mathrm{r}}_k$ is the error due to the randomized asynchronicity. In what follows we will discuss these terms.

Error due to the noise

The input noise affects the error correlation matrix through the term $\mathbf{Q}^{\mathrm{n}}$ defined by the equation (4.43). By vectorizing both sides of (4.43), a numerical solution to $\mathbf{Q}^{\mathrm{n}}$ can be obtained as follows:

$$\mathrm{vec}(\mathbf{Q}^{\mathrm{n}}) = \big(\mathbf{I} - \mathbf{\Phi}\big)^{-1} \mathrm{vec}\left(\mathbf{P}\,\mathbf{\Gamma}\,\mathbf{P} + \mathbf{\Gamma} \odot \big(\mathbf{P} - \mathbf{P}^2\big)\right). \tag{4.45}$$

We point out that $\mathbf{Q}^{\mathrm{n}}$ satisfies $\mathbf{Q}^{\mathrm{n}} \succ \mathbf{0}$ as long as $\mathbf{\Gamma} \succ \mathbf{0}$ (see Corollary 4.2 in Section 4.3.3), and it does not have any dependency on the iteration index k. So, the effect of the input noise remains the same throughout the iterations (which is *not* the case for $\mathbf{Q}^{\mathrm{r}}_k$). Furthermore, it is clear from (4.45) that $\mathbf{Q}^{\mathrm{n}}$ depends linearly on the noise covariance matrix $\mathbf{\Gamma}$. However, the error due to the noise is always larger than the noise itself, which is stated more precisely in the following lemma:

Lemma 4.1. *For any given* $\mathbf{A}$ *and* $\mathbf{P}$ *satisfying the stability condition* (4.41)*, the following holds true:*

$$(\mathbf{Q}^{\mathrm{n}})_{i,i} \geq (\mathbf{\Gamma})_{i,i} \qquad \forall\, 1 \leq i \leq N, \tag{4.46}$$

where $\mathbf{Q}^{\mathrm{n}}$ *is the solution of the linear matrix equation in* (4.43).

Proof. See Section 4.8.5. □

In words, Lemma 4.1 states that the error variance due to the noise in a state variable is always larger than the variance of the noise at the input term of the state recursion. The relation between $\mathbf{Q}^{\mathrm{n}}$ and the matrices $\mathbf{A}$ and $\mathbf{P}$ are quiet intricate. In fact, one can search for a set of probabilities that minimize $\mathrm{tr}(\mathbf{Q}^{\mathrm{n}})$ for a given $\mathbf{A}$ and $\mathbf{\Gamma}$. However, the optimal choice for $\mathbf{P}$ is not known at this time.

Randomization Error

Due to the randomized nature of the updates in (4.13) there is an inherent error in the state vector that is given precisely by the term $\mathbf{Q}^{\mathrm{r}}_k$ in (4.44). An important observation is that $\mathbf{Q}^{\mathrm{r}}_k$ *does depend* on the iteration index k in general (unlike the error due to the noise). More precisely, the solution to (4.44) can be written explicitly as follows:

$$\mathbf{Q}_k^{\mathrm{r}} = \sum_{i=1}^{R}\sum_{j=1}^{R} \mathbf{Q}_{i,j}\ \left(\alpha_i\,\alpha_j^*\right)^k, \tag{4.47}$$

where α_i's denote the base of the exponential input signals as in (4.6), and the matrices $\mathbf{Q}_{i,j}$'s are as follows:

$$\mathrm{vec}(\mathbf{Q}_{i,j}) = (1-\alpha_i)\,(1-\alpha_j^*)\,\left(\alpha_i\,\alpha_j^*\,\mathbf{I}-\mathbf{\Phi}\right)^{-1}\mathrm{vec}\left((\mathbf{x}_{0,i}^{\mathrm{ss}}\,(\mathbf{x}_{0,j}^{\mathrm{ss}})^{\mathrm{H}})\odot(\mathbf{P}^{-1}-\mathbf{I})\right), \tag{4.48}$$

and $\mathbf{x}_{0,i}^{\mathrm{ss}}$ is as in (4.20).

The solution in (4.47) shows that decaying components of the input signal, i.e.,$|\alpha_i| < 1$, affect the error correlation matrix initially only. Their effect fade away as the iterations progress. On the other hand, the components with $|\alpha_i| = 1$ (i.e., complex sinusoids) have a sustaining effect on the error correlation.

The long term behavior

The expression in (4.47) shows that the randomization error (hence, the error correlation itself) has an oscillatory behavior in general. We refer to Figure 4.3 for a numerical example. Furthermore, when ω_i denotes the frequency of the i^{th} complex exponential input, (4.47) shows that the error correlation matrix has a component with frequency $\omega_i - \omega_j$ for all $1 \leq i, j \leq R$. So, difference frequencies are also important in the behavior of the error correlation matrix. We will re-visit this point later in Section 4.6.1.

The case of single input

When the input signal consists of only one complex exponential, i.e., $R = 1$ and $u_k = e^{j\omega k}$, the expression for the randomization error in (4.47) reduces to the following simpler form:

$$\mathrm{vec}(\mathbf{Q}_k^{\mathrm{r}}) = 4\sin^2(\omega/2)\left(\mathbf{I}-\mathbf{\Phi}\right)^{-1}\mathrm{vec}\left((\mathbf{x}_0^{\mathrm{ss}}(\mathbf{x}_0^{\mathrm{ss}})^{\mathrm{H}})\odot(\mathbf{P}^{-1}-\mathbf{I})\right), \tag{4.49}$$

where it is important to note that the right-hand side is free from the iteration index k. So, when there is only one complex exponential input, the randomization error $\mathbf{Q}_k^{\mathrm{r}}$ does not vary throughout the iterations despite the time-varying nature of the input signal. As a result, the error correlation matrix $\mathbf{Q}_k$ converges to a point as long as the condition (4.41) is met.

Although the randomization error stays the same throughout the iterations in the case of single complex exponential input, the actual value of the error depends on the input frequency ω as well as the update probabilities $\mathbf{P}$. Generally speaking, the randomization error tends to be smaller when the input signal varies slowly. It is also generally true that randomization error tends to be larger as the update probabilities gets smaller. We will elaborate more on this topic later in Section 4.6.2.

4.3.3 Markov Jump Linear System Viewpoint

We would like to point out that the random asynchronous model in (4.13) can be viewed as a particular instance of a Markov jump linear system, which has the following model:

$$\mathbf{x}_{k+1} = \mathbf{A}_{i_k}\, \mathbf{x}_k + \mathbf{B}_{i_k}\, \mathbf{u}_k, \tag{4.50}$$

where i_k denotes the state of the underlying Markov chain at the iteration k, and the Markov chain has finite number of states. So, the state vector $\mathbf{x}_k$ is updated with a different state transition matrix in every iteration (as determined by the state of the Markov chain). This is a well-studied model, and we refer to [42] for a rigorous treatment of the topic.

It is possible to represent the random asynchronous model (4.13) as a Markov jump system with the underlying Markov chain having 2^N states since there are 2^N different ways of selecting an update set in every iteration of the model. So, a direct application of [42, Corollary 3.26] to the random asynchronous model (4.13) gives the following result:

Lemma 4.2. *The following statements are equivalent:*

- *Random asynchronous model in* (4.13) *is stable in the mean-squared sense,*
- $\rho(\mathbf{\Phi}) < 1$,
- *There exists* $\mathbf{X} \succ \mathbf{0}$ *such that* $\mathbf{X} \succ \varphi(\mathbf{X})$*,*
- *For any given* $\mathbf{Y} \succ \mathbf{0}$*, there exists a unique* $\mathbf{X} \succ \mathbf{0}$ *such that* $\mathbf{X} = \varphi(\mathbf{X}) + \mathbf{Y}$*,*

where $\varphi(\cdot)$ *is as in* (4.33)*, and the matrix* $\mathbf{\Phi}$ *is as in* (4.38).

Proof. See Section 4.8.6. □

Although Lemma 4.2 endorses the importance of the matrix $\mathbf{\Phi}$ in the stability of the random asynchronous model, it is important to note that the viewpoint of switching systems provides a more general framework for randomized linear techniques (see Chapter 4 of this thesis). In the case of random asynchronous updates, underlying state transition matrices (i.e., states of the Markov chain) are related to the matrix $\mathbf{A}$ in a very specific way. So, the analysis presented in this chapter is tailored for the model in (4.13), and its mean-squared stability is shown to be determined precisely by the spectral radius of the matrix $\mathbf{\Phi}$.

4.3.4 A Sufficient Condition

In addition to the necessary and sufficient condition given by Corollary 4.2, it is also possible to ensure the stability of the recursions with a stronger condition based on a simple linear matrix inequality. In this regard, we present the following result as a corollary to Theorem 4.2:

Corollary 4.3. *If the state-transition matrix* $\mathbf{A}$ *and the update probabilities* $\mathbf{P}$ *satisfy the following:*

$$\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}, \tag{4.51}$$

then, the trace of the error correlation matrix of the state variables can be bounded as follows:

$$\limsup_{k\to\infty}\ \mathrm{tr}(\mathbf{Q}_k) \ \leq\ \frac{\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma}) + \mathbf{\Delta}^2\ \|\mathbf{P}^{-1} - \mathbf{I}\|_2}{\lambda_{\min}(\mathbf{P} - \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A})}, \tag{4.52}$$

where $\mathbf{\Delta}$ *is an arbitrary number satisfying the following:*

$$\|\boldsymbol{\delta}_k\|_2 \leq \mathbf{\Delta} \qquad \forall\, k. \tag{4.53}$$

Proof. See Section 4.8.7. □

A number of remarks regarding Corollary 4.3 are in order:

1) *Convergence of the error correlation:* When the input signal consists of multiple exponential signal, i.e., $R > 1$, the error correlation matrix shows an oscillatory behavior as described in (4.47). As a result, $\mathrm{tr}(\mathbf{Q}_k)$ does not converge in general. Nevertheless, (4.52) provides an upper bound on the error term as the iterations progress (k goes to infinity).

2) *Difference in the state variables:* As long as the state variables have a finite steady-state component, it is always possible to select a finite value for $\mathbf{\Delta}$. Generally speaking, when the input signal $\mathbf{u}_k$ varies slowly, the vector $\boldsymbol{\delta}_k$ tends to be smaller.

So, the upper bound (as well as the error term itself) gets smaller. We will elaborate more on this topic later in Section 4.6.2.

3) *Equal probabilities:* When all the indices are updated with equal probabilities, i.e., $\mathbf{P} = p\,\mathbf{I}$ for some p, the condition (4.51) reduces to $\|\mathbf{A}\|_2 < 1$. So, the error correlation matrix stays bounded as long as the state variables are updated with equal probabilities (no matter what the probability is) and $\mathbf{A}$ has a bounded spectral norm. However, the probability does affect the actual value of the error correlation matrix.

4.4 Synchronous v.s. Asynchronous Stability: Comparisons

In previous sections we studied stability of the randomized state recursions from two different perspectives, namely expected behavior and the second order statistics of the error term. In this section, we will discuss the relations between stability of the state-transition matrix $\mathbf{A}$ and the matrices $\bar{\mathbf{A}}$ and $\mathbf{\Phi}$. In particular, we will show that stability in the synchronous world and stability in the asynchronous world do not imply each other in general. Furthermore, stability in the asynchronous world depends on the update probabilities $\mathbf{P}$ as well as the eigenvector structure of the matrix $\mathbf{A}$.

Recall from Section 4.3 that when a randomized asynchronous system is said to be stable, it means that the error correlation matrix $\mathbf{Q}_k$ stays bounded as the randomized iterations progress. In general, the error correlation matrix does not converge to zero even when no noise is present in the system. In fact, the amount of error depends on the amount of variation in the input signal as well as the update probabilities.

Two remarks are in order:

The Synchronous Case

Results given by Theorem 4.1 and Corollary 4.2 are consistent with the synchronous case. That is,

$$\mathbf{P} = \mathbf{I} \qquad \Longrightarrow \qquad \bar{\mathbf{A}} = \mathbf{A}, \qquad \mathbf{\Phi} = \mathbf{A}^* \otimes \mathbf{A}, \tag{4.54}$$

which shows that stability of $\mathbf{A}$, $\bar{\mathbf{A}}$ and $\mathbf{\Phi}$ are equivalent to each other in the case of synchronous updates. In this case we also note that the randomization error (4.44) becomes $\mathbf{Q}_k^{\mathrm{r}} = \mathbf{0}$. So, as long as $\mathbf{A}$ is a stable matrix, the error correlation matrix converges to $\mathbf{Q}^{\mathrm{n}}$ whose value depends only on the noise statistics $\mathbf{\Gamma}$ and the matrix $\mathbf{A}$.

Strict Positivity of the Update Probabilities

Stability of the matrix $\mathbf{\Phi}$ implicitly ensures the strict positivity of the update probabilities. More precisely,

$$\rho(\mathbf{\Phi}) < 1 \qquad \Longrightarrow \qquad \mathbf{P} > \mathbf{0}, \tag{4.55}$$

which can be verified by observing that when there exists an index i such that $p_i = 0$ the matrix $\mathbf{\Phi}$ has a left eigenvector $\mathbf{e}_i \otimes \mathbf{e}_i$ with eigenvalue 1. So, the stability condition requires no state variable to be left out *permanently* during the updates.

4.4.1 Mean v.s. Mean-Squared Error

Since having a finite variance is more restrictive than having a finite mean for a random variable, it is reasonable to expect that stability of the matrix $\mathbf{\Phi}$ is more restrictive than stability of the matrix $\bar{\mathbf{A}}$. This is, indeed, the case:

Lemma 4.3. *Stability of the matrix $\mathbf{\Phi}$ implies stability of the matrix $\bar{\mathbf{A}}$, that is,*

$$\rho(\mathbf{\Phi}) < 1 \qquad \Longrightarrow \qquad \rho(\bar{\mathbf{A}}) < 1. \tag{4.56}$$

Proof. From the definition of $\varphi(\cdot)$ in (4.33), we have $\varphi(\mathbf{X}) \geq \bar{\mathbf{A}}\,\mathbf{X}\,\bar{\mathbf{A}}^{\mathrm{H}}$ for any $\mathbf{X} \geq \mathbf{0}$.

From Lemma 4.2, the condition $\rho(\mathbf{\Phi}) < 1$ implies that there exist $\mathbf{X} > \mathbf{0}$ such that the following holds true:

$$\mathbf{X} > \varphi(\mathbf{X}) \geq \bar{\mathbf{A}}\,\mathbf{X}\,\bar{\mathbf{A}}^{\mathrm{H}} \qquad \Longrightarrow \qquad \mathbf{X} > \bar{\mathbf{A}}\,\mathbf{X}\,\bar{\mathbf{A}}^{\mathrm{H}}, \tag{4.57}$$

which implies that $\rho(\bar{\mathbf{A}}) < 1$ due to the stability properties of the discrete Lyapunov equation. $\square$

The importance of Lemma 4.3 follows from the fact that there is no need to consider matrices $\bar{\mathbf{A}}$ and $\mathbf{\Phi}$ separately. As long as $\mathbf{\Phi}$ is stable, it is guaranteed that $\mathbb{E}[\mathbf{q}_k]$ converges to zero and $\mathbb{E}[\|\mathbf{q}_k\|_2^2]$ remains bounded as the iterations progress. Since the converse of (4.56) does not hold true in general (see Figure 4.2), we will focus on stability of $\mathbf{\Phi}$ in the rest of this section.

4.4.2 Stability in Synchronous v.s. Asynchronous Case

The most important observation regarding stability of the randomized asynchronous state recursions is that a stable synchronous system may get unstable with randomized asynchronicity, and conversely an unstable system (in the synchronous world) may be stabilized simply by the use of randomized asynchronicity. This is a remarkable property of the randomized asynchronous updates, which is observed also in Chapter 2 for the case of $\mathbf{A}$ being a normal matrix. We state this observation formally with the following lemma:

Lemma 4.4. *In general, stability of* $\mathbf{A}$ *is neither necessary nor sufficient for stability of* $\bar{\mathbf{A}}$ *and stability of* $\mathbf{\Phi}$*.*

Proof. Consider the following examples of size $N = 2$:

$$\mathbf{A}_1 = \begin{bmatrix} \text{-0.9} & 0.8 \\ 0.8 & \text{-0.3} \end{bmatrix}, \qquad \mathbf{A}_2 = \begin{bmatrix} 1.25 & 0.25 \\ \text{-6.25} & \text{-1.25} \end{bmatrix}, \tag{4.58}$$

which can be verified to satisfy $\rho(\mathbf{A}_1) > 1$ and $\rho(\mathbf{A}_2) = 0$. Then, we construct the matrices $\bar{\mathbf{A}}$ and $\mathbf{\Phi}$ as in (4.19) and (4.38), respectively for both $\mathbf{A}_1$ and $\mathbf{A}_2$ for all possible values of $\mathbf{P} = \text{diag}([p_1 \;\; p_2])$. Figure 4.2 presents the regions of $\mathbf{P}$ for which $\bar{\mathbf{A}}$ is stable, or $\mathbf{\Phi}$ is stable. This proves the claim. $\square$

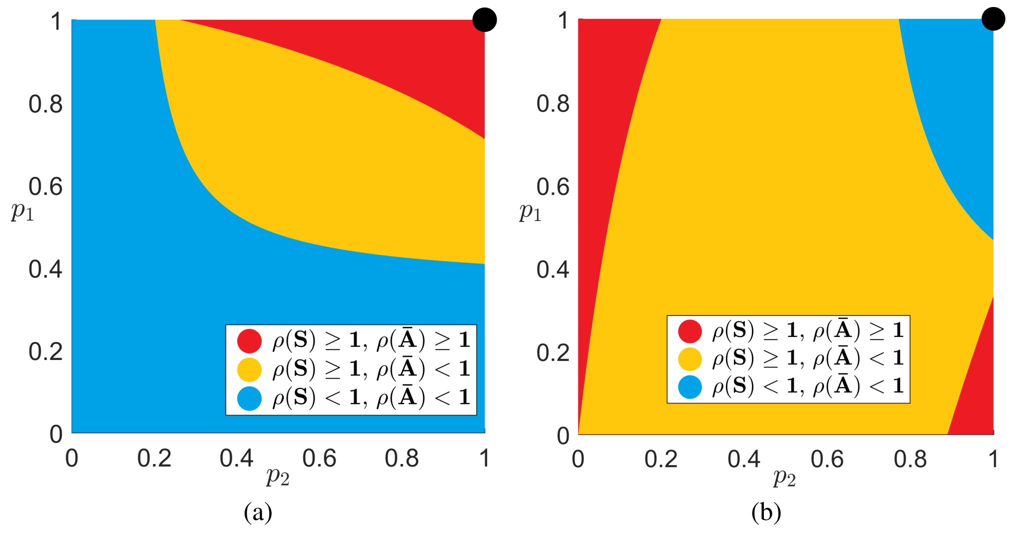


Figure 4.2: The set of probabilities that ensures the stability of the randomized asynchronous updates for the matrices (a) $\mathbf{A}_1$, (b) $\mathbf{A}_2$ defined in (4.58). The top-right corner indicates $\mathbf{P} = \mathbf{I}$, which corresponds to the synchronous case.

We would like to note that the sufficiency condition given by Corollary 4.3 does require $\mathbf{A}$ to be a stable matrix. So, Corollary 4.3 fails to explain why unstable synchronous systems may get stable with the randomized asynchronicity. The importance of Corollary 4.3 follows from the fact that condition (4.51) is easy to check, or satisfy, in practical applications (see Theorem 3.3 in Chapter 3 of this thesis).

4.4.3 The Set of Probabilities Ensuring Stability

Although Figure 4.2a shows that some unstable systems (in the synchronous world) can get stable with the use of randomized asynchronicity, it should be noted that *not every unstable synchronous system can get stable with randomized asynchronicity.* Therefore, it is important to check whether or not there exists a set of update probabilities for which randomized asynchronous recursions are stable. In this regard, we consider the following definition:

Definition 4.1 (The stability set)**.** *For a given matrix* $\mathbf{A}$*, we will use* $\mathcal{S}(\mathbf{A})$ *to denote the set of probabilities such that the matrix* $\mathbf{\Phi}$ *is stable. More precisely,*

$$\mathcal{S}(\mathbf{A}) = \big\{\mathbf{P} \mid \rho(\mathbf{\Phi}) < 1, \;\; \mathbf{0} \leq \mathbf{P} \leq \mathbf{I}\big\}, \tag{4.59}$$

where $\mathbf{P}$ *is diagonal, and the matrix* $\mathbf{\Phi}$ *is as in* (4.38)*.*

As a numerical example, consider the matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ in (4.58). The blue regions in Figures 4.2a and 4.2b denote the stability sets $\mathcal{S}(\mathbf{A}_1)$ and $\mathcal{S}(\mathbf{A}_2)$, respectively.

Some remarks are in order regarding Definition 4.1:

1) If the set $\mathcal{S}(\mathbf{A})$ is empty for a given matrix $\mathbf{A}$, then the system cannot be stable whether the updates are synchronous, or asynchronous.

2) When the matrix $\mathbf{A}$ itself is stable, i.e., $\rho(\mathbf{A}) < 1$, the stability set $\mathcal{S}(\mathbf{A})$ is not empty since $\mathbf{I} \in \mathcal{S}(\mathbf{A})$.

3) The stability set $\mathcal{S}(\mathbf{A})$ is not convex in general. That is, a random asynchronous system can be stable with probabilities $\mathbf{P}_1$ or $\mathbf{P}_2$, but it may get unstable with $\tau\,\mathbf{P}_1 + (1-\tau)\,\mathbf{P}_2$ where $0 < \tau < 1$. See Figure 4.2a as an example, in which the stability set (indicated with color blue) is visibly non-convex.

4) The sufficiency condition given by Corollary 4.3 describes *a convex subset* of the stability set. More precisely,

$$\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P} \quad \Longrightarrow \quad \mathbf{P} \in \mathcal{S}(\mathbf{A}), \tag{4.60}$$

and it is readily verified that the set of probabilities satisfying the condition (4.51) is convex. However, it should be noted that when $\mathbf{A}$ is not a stable matrix, the set of probabilities described by the condition $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}$ is empty, whereas the stability set $\mathcal{S}(\mathbf{A})$ may or may not be empty.

4.4.4 Importance of the Similarity Transform

In discrete-time, linear, time-invariant systems (i.e., the case of synchronous state recursions), it is well-known that stability is determined by the poles of the system (i.e., eigenvalues of $\mathbf{A}$), and stability is not affected by a similarity transform. More precisely, for a given *invertible* matrix $\mathbf{T} \in \mathbb{C}^{N\times N}$, we have

$$\rho(\mathbf{A}) = \rho\big(\mathbf{T}^{-1}\,\mathbf{A}\,\mathbf{T}\big). \tag{4.61}$$

So, as long as all the eigenvalues of $\mathbf{A}$ are inside the open unit disk the system is stable irrespective of the similarity transform $\mathbf{T}$. Nevertheless, we point out that similarity transforms play a role in some other aspects, such as the suppression of the overflow oscillations in fixed point digital filters [120, 197, 123].

On the contrary, eigenvectors of the matrix $\mathbf{A}$ (in addition to eigenvalues) affect the mean-squared stability of random asynchronous recursions, and in general,

$$\mathcal{S}(\mathbf{A}) \neq \mathcal{S}\big(\mathbf{T}^{-1}\,\mathbf{A}\,\mathbf{T}\big). \tag{4.62}$$

In order to point out the importance of a similarity transform more rigorously, we present the following lemma:

Lemma 4.5. *Let* $\mathbf{A} \in \mathbb{C}^{N\times N}$ *be a triangular matrix with* $A_{i,i}$ *denoting the* i^{th} *diagonal element of* $\mathbf{A}$. *Then,*

$$\rho(\mathbf{\Phi}) = 1 + \max_{1\leq i\leq N} p_i\,\big(\,|A_{i,i}|^2 - 1\,\big). \tag{4.63}$$

In addition, the following holds true for a triangular matrix $\mathbf{A}$ *irrespective of the index update probabilities:*

$$\rho(\mathbf{A}) < 1 \qquad \Longleftrightarrow \qquad \rho(\mathbf{\Phi}) < 1. \tag{4.64}$$

Proof. See Section 4.8.8. □

The significance of Lemma 4.5 follows from the fact *any* square matrix $\mathbf{A}$ is unitarily triangularizable (i.e., the Schur decomposition [85, Section 2.3]):

$$\mathbf{A} = \mathbf{T}\,\mathbf{U}\,\mathbf{T}^{-1}, \tag{4.65}$$

where $\mathbf{U}$ is upper triangular and $\mathbf{T}$ is *unitary*. Thus, a linear system with stable poles can always be realized in a triangular form that is guaranteed to be stable under the random asynchronous model *irrespective of the update probabilities*. In the non-triangular case, a realization of a system with stable poles is not guaranteed to remain stable under the randomized asynchronous model in general. (See Lemma 4.4.)

An interesting special case of the discussion above is the finite impulse response (FIR) system. In this regard, consider the example $\mathbf{A}_2$ from (4.58) together with the direct form description of an FIR system of size $N = 2$:

$$\mathbf{A}_2 = \begin{bmatrix} 1.25 & 0.25 \\ -6.25 & -1.25 \end{bmatrix}, \qquad \mathbf{A}_3 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \tag{4.66}$$

where $\mathbf{A}_2$ and $\mathbf{A}_3$ are similar to each other (i.e., there exists a $\mathbf{T}$ such that $\mathbf{A}_2 = \mathbf{T}^{-1}\mathbf{A}_3\mathbf{T}$). Although a randomized system with $\mathbf{A}_3$ is stable for any set of probabilities (Lemma 4.5), a randomized system with $\mathbf{A}_2$ may get unstable for some set of probabilities as shown in Figure 4.2b.

Finally, we note that the optimal similarity transform is not known at this point, and it is left as a future research direction.

4.5 The Constant Input: Fixed-Point Iterations

In this section we will consider the random asynchronous model with a constant input, that is, the input signal will be assumed to be in the form of $\mathbf{u}_k = \mathbf{u}$ for all k. Although this is a special case of what we studied in the previous sections, the constant input case gives more perspective from the viewpoint of fixed-point iterations. It allows us to relate these results to some known classical results for asynchronous updates [32, 14, 19, 20]. It also reveals connections to eigenspace estimation, singular vector estimation, and principal component analysis in the random asynchronous context.

We start by noting that when the input is constant, it suffices to consider the case of $R = 1$ with $\alpha = 1$ in the model (4.6) since a linear combination of constant inputs can be equivalently considered as a single constant input. In particular, we will consider the following type of updates:

$$(\mathbf{x}_{k+1})_i = \begin{cases} (\mathbf{A}\,\mathbf{x}_k)_i + (\mathbf{B} + \mathbf{w}_k)_i, & i \in \mathcal{T}_{k+1}, \\ (\mathbf{x}_k)_i, & i \notin \mathcal{T}_{k+1}, \end{cases} \tag{4.67}$$

where the noise term $\mathbf{w}_k$ follows the statistics in (4.3) as before.

Regarding the expected value of the state variables in (4.67), Theorem 4.1 gives the following

$$\mathbf{x}_k^{\mathrm{ss}} = \left(\mathbf{I} - \bar{\mathbf{A}}\right)^{-1} \bar{\mathbf{B}} = \left(\mathbf{I} - \mathbf{A}\right)^{-1} \mathbf{B} = \mathbf{x}^{\star}, \tag{4.68}$$

which shows that the steady-state component $\mathbf{x}_k^{\mathrm{ss}}$ depends neither on the update probabilities nor on the iteration index k. In fact, $\mathbf{x}_k^{\mathrm{ss}} = \mathbf{x}^\star$ corresponds to *the fixed-point* of the asynchronous iterations in (4.67), i.e., $\mathbf{x}^\star = \mathbf{A}\,\mathbf{x}^\star + \mathbf{B}$.

Although the fixed-point is determined solely by the pair $(\mathbf{A},\mathbf{B})$, the convergence of the random vector $\mathbf{x}_k$ to the fixed-point $\mathbf{x}^\star$ is still affected by the update probabilities $\mathbf{P}$ (as well as the matrix $\mathbf{A}$). In particular, Corollary 4.2 shows that the condition (4.41), i.e., stability of the matrix $\mathbf{\Phi}$, is both necessary and sufficient for the convergence of $\mathbf{x}_k$ to $\mathbf{x}^\star$ in the mean-squared sense when no noise is present. When there is noise, the error correlation matrix converges to $\mathbf{Q}^{\mathrm{n}}$ as given in (4.43).

Additionally, in the case of a constant input it is clear that the vector defined in (4.36) becomes $\boldsymbol{\delta}_k = \mathbf{0}$. So, we can select $\mathbf{\Delta} = 0$ in (4.53), and Corollary 4.3 shows that whenever $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}$ is satisfied, we have the following:

$$\lim_{k\to\infty} \mathbb{E}\big[\|\mathbf{x}_k - \mathbf{x}^\star\|_2^2\big] \le \frac{\mathrm{tr}(\mathbf{P}\,\mathbf{\Gamma})}{\lambda_{\min}(\mathbf{P} - \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A})}, \tag{4.69}$$

where limit supremum from (4.52) is replaced with limit since the error correlation matrix indeed converges to $\mathbf{Q}^{\mathrm{n}}$. We note that the bound in (4.69) was first presented in Corollary 3.1 together with a lower bound on the limit of the error term.

4.5.1 Comparison with the Classical Results

Asynchronous (non-random) fixed-point iterations are well studied problems in the literature. Theoretical analysis of the linear case can be traced back to the studies in [32, 14], which assume that only one index is updated per iteration and allow the use of the past values of the iterant. The study [32] showed that the following condition is both necessary and sufficient for the convergence of the asynchronous updates:

$$\rho(|\mathbf{A}|) < 1, \tag{4.70}$$

where $|\mathbf{A}|$ is the matrix obtained by replacing the elements of $\mathbf{A}$ by their absolute values. In the non-random setting considered in [32, 14], the condition (4.70) is necessary in the sense that when (4.70) is violated there exists an index sequence for which asynchronous iterations do not converge.

It can be shown that the condition (4.70) is more restrictive than the stability of the matrix $\mathbf{A}$. It is even more restrictive than the sufficient condition given by Corollary 4.3. That is, the following holds true (Lemma 3.1):

$$\rho(|\mathbf{A}|) < 1 \quad \Longrightarrow \quad \exists\,\mathbf{P} \quad \text{s.t.} \quad \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}. \tag{4.71}$$

So, if $\mathbf{A}$ is unstable, there exists an index sequence for which iterations do not converge. On the other hand, Corollary 4.2 shows that the convergence may be achieved even when $\mathbf{A}$ is unstable. Although these results appear to be contradictory, the difference is the notion of convergence: *the condition* (4.70) *is necessary and sufficient for the convergence of any index sequence (as in sure convergence), whereas the condition* (4.41) *is necessary and sufficient for the mean-squared convergence.* In addition, the result in [42, Corollary 3.46] implies that the condition (4.41) is *sufficient* for almost sure convergence as well.

As a result, we conclude that the asynchronous case is more restrictive than the synchronous case when the worst case behavior is considered. On the other hand, the asynchronous case may be less restrictive than the synchronous case when the statistical behavior is considered.

4.5.2 The Case of Zero Input

The asynchronous model (4.13) shows an interesting behavior when the input signal is identically zero, i.e., $\mathbf{u}_k = \mathbf{0}$ for all k, which can be equivalently represented as taking $\mathbf{B} = \mathbf{0}$. In this case the state recursions reduce to the following form:

$$(\mathbf{x}_{k+1})_i = \begin{cases} (\mathbf{A}\,\mathbf{x}_k)_i + (\mathbf{w}_k)_i, & i \in \mathcal{T}_{k+1}, \\ (\mathbf{x}_k)_i, & i \notin \mathcal{T}_{k+1}, \end{cases} \tag{4.72}$$

and the fixed-point becomes $\mathbf{x}^\star = \mathbf{0}$. So, under the stability condition (4.41) the state variables converge to zero in the mean-squared sense (or, reach an error floor determined by $\mathbf{Q}^{\mathrm{n}}$).

It is important to note that the existence of the fixed-point $\mathbf{x}^\star$ in (4.68) requires $\mathbf{A}$ not to have an eigenvalue equal to 1 so that $\mathbf{I} - \mathbf{A}$ is invertible. This requirement is satisfied implicitly by the stability condition (4.41) since $\mathbf{A}$ having an eigenvalue equal to 1 implies $\rho(\bar{\mathbf{A}}) \geq 1$, thus $\rho(\mathbf{\Phi}) \geq 1$.

When the matrix $\mathbf{A}$ *has an eigenvalue equal to* 1, there are infinitely many fixed-points (as opposed to the unique one in (4.68)), and they correspond to the eigenspace of $\mathbf{A}$ with the eigenvalue 1. Nevertheless, recursions in (4.72) can still be stable, and the random vector $\mathbf{x}_k$ can convergence to a point in the eigenspace (an eigenvector). This convergence behavior is studied in Chapter 2, where random asynchronous recursions are used for obtaining spectral clustering in autonomous networks. Furthermore, Section 2.8 used the model (4.72) for distributed asynchronous computation of dominant singular vectors of a given data matrix.

4.6 Further Discussions

In this section we make further observations regarding the behavior of random asynchronous recursions. These bring further insights into the behavior of such systems.

4.6.1 Real Sinusoidal v.s. Complex Exponential Input

First we will compare the behavior of the error correlation matrix for the following two cases of inputs:

$$u_k = e^{j\omega k}, \qquad\qquad u_k = \cos(\omega k). \tag{4.73}$$

Both input signals oscillate at frequency ω, however the error correlation matrix $\mathbf{Q}_k$ converges to a point in the case of $e^{j\omega k}$, but $\mathbf{Q}_k$ oscillates with frequency 2ω in the case of $\cos(\omega k)$. This is due to $\cos(\omega k) = (e^{j\omega k} + e^{-j\omega k})/2$, so the difference frequencies are $\{-2\omega,\ 0,\ 2\omega\}$. Thus, the randomization error in (4.47) has terms oscillating at frequency 2ω, and so does the error correlation $\mathbf{Q}_k$ as k goes to infinity.

We demonstrate this behavior in Figure 4.3, in which we use the numerical example in (4.27). When the second rows of Figure 4.3a and Figure 4.3b are compared, it is clear that error correlation shows an oscillatory behavior in the case of a real sinusoid, whereas it converges to a constant value in the case of a complex exponential as given in (4.49).

4.6.2 Signal-to-Randomization Error Ratio (SRR)

Corollary 4.2 shows that state variables always contain a randomization error even when no noise is present in the input. So, the state vector can be decomposed as follows:

$$\mathbf{x}_k = \mathbf{x}_k^{\mathrm{ss}} + \mathbf{q}_k, \tag{4.74}$$

where $\mathbf{x}_k^{\mathrm{ss}}$ denotes the expected behavior of the state variables as the iterations progress (see Theorem 4.1), and $\mathbf{q}_k$ denotes the randomization error that has a zero mean and a finite variance under the stability condition given by Corollary 4.2. Therefore, it is important to consider a *signal-to-randomization error ratio (SRR)* when studying the behavior of the state vector $\mathbf{x}_k$ in a random asynchronous system. In this regard, we consider the following quantity:

$$\mathrm{SRR}_k = \|\mathbf{x}_k^{\mathrm{ss}}\|_2^2 \,/\, \mathrm{tr}(\mathbf{Q}_k^{\mathrm{r}}). \tag{4.75}$$

In general, the quantity in (4.75) varies with k since both $\mathbf{x}_k^{\mathrm{ss}}$ and $\mathrm{tr}(\mathbf{Q}_k^{\mathrm{r}})$ are functions of k as described in Theorem 4.1 and Corollary 4.2, respectively. For the sake of

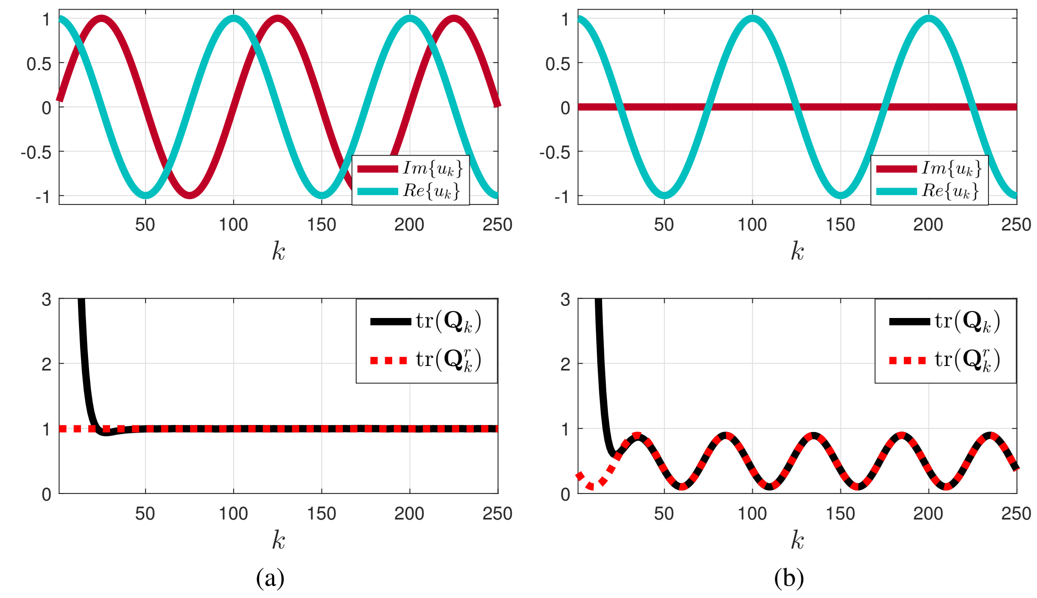


Figure 4.3: Comparison between inputs (a) $u_k = e^{j\omega k}$, and (b) $u_k = \cos(\omega k)$ with frequency $\omega = 2\pi/100$. The first row shows the input signal, and the second row shows the trace of the error correlation matrix, which is obtained by averaging over 10^6 independent runs.

simplicity, in the rest of this section we assume that the input signal is $u_k = e^{j\omega k}$. So, we have $\|\mathbf{x}_k^{\text{ss}}\| = \|(e^{j\omega}\,\mathbf{I} - \bar{\mathbf{A}})^{-1}\,\bar{\mathbf{B}}\|$, and the randomization error is given in (4.49). Thus, SRR_k remains constant as a function of the iterations. However, the value of SRR_k depends on the input frequency ω as well as the update probabilities $\mathbf{P}$. In order to demonstrate this point, we compute SRR numerically for the system in (4.27) for different values of ω and p. These results are presented in Figure 4.4.

Generally speaking, SRR tends to be larger as the input signal varies more or the state variables are updated less frequently.

We first note that the matrix $\mathbf{A}$ given in (4.27) has $\rho(\mathbf{A}) > 1$. However, for the case of $\mathbf{P} = p\,\mathbf{I}$, we have numerically verified that the stability condition (4.41) is satisfied for $0 < p \leq 0.9542$. This is why the case of $p > 0.9542$ is excluded in Figure 4.4.

Heuristically speaking, Figure 4.4 shows that state variables in the randomized case are trying to “keep up with the input signal.” As the input signal varies faster (larger values of ω), or the state variables are updated slower (small values of p), the randomization error tends to be larger. However, SRR_k is *not* monotonic in terms of the input frequency ω and the update probabilities $\mathbf{P}$ in general. This complicated behavior follows from the fact that signal power and randomization error change

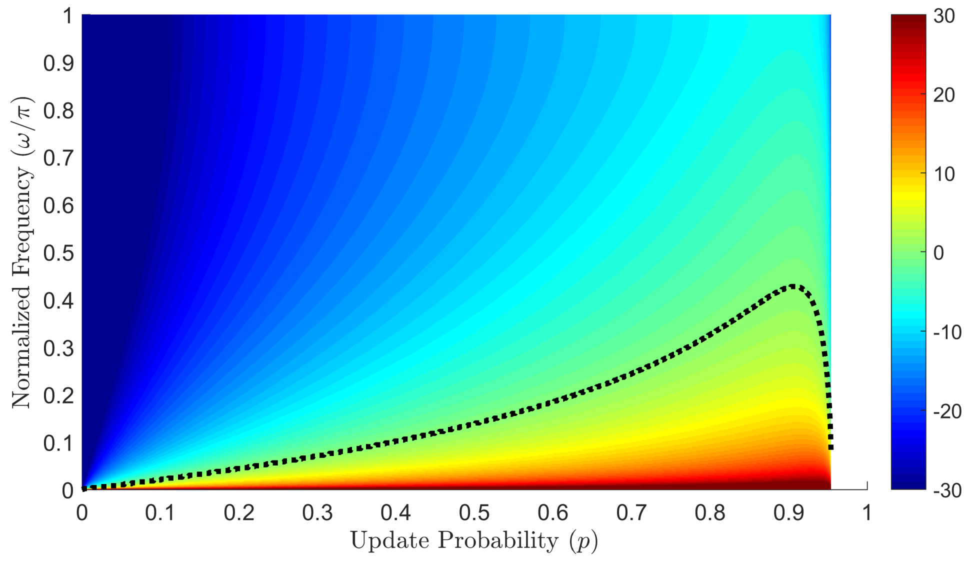


Figure 4.4: SRR of the example in (4.27) for different values of the input frequency ω and $\mathbf{P} = p\,\mathbf{I}$. The plot is in dB scale, i.e., $10\log_{10}(\text{SRR})$ is plotted. Dotted black curve corresponds to the case of 0 dB.

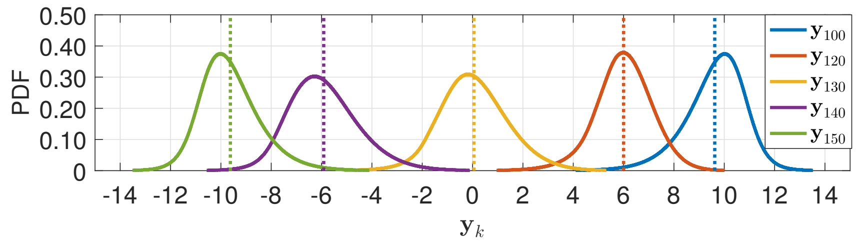
simultaneously with the input frequency and the update probabilities.

4.6.3 The Distribution of the Random Variables

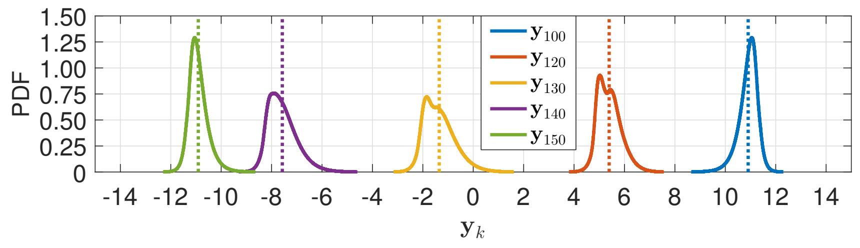
Due to random selection of the indices, the state vector $\mathbf{x}_k$ in the asynchronous model (4.13) is a discrete random vector with at most 2^{Nk} distinct values, as there are 2^N different ways of selecting an update set in any iteration. We note that the first and the second order statistics of $\mathbf{x}_k$ are described previously in Theorem 4.1 and Corollary 4.2, respectively. In this section, we will consider the distribution of the random vector $\mathbf{x}_k$ (as well as the output $\mathbf{y}_k$).

One can consider approximating $\mathbf{x}_k$ with a multivariate complex Gaussian random vector with mean $\mathbf{x}_k^{\text{ss}}$ and covariance $\mathbf{Q}_k$ due to the independent selection of the indices and the central limit theorem. Contrary to the anticipation, we have numerically observed that *the random vector* $\mathbf{x}_k$ *does not have a Gaussian distribution* in general as we demonstrate next.

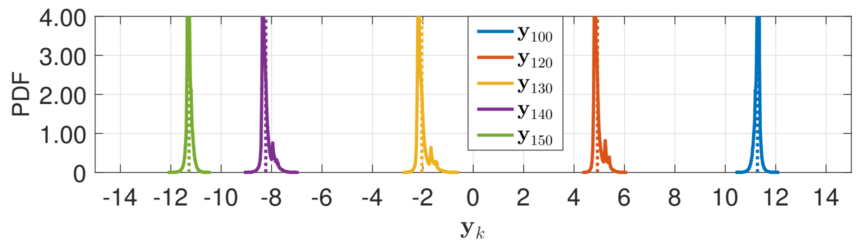
In this regard, we consider the system in (4.27) with the input $u_k = \cos(2\pi k/100)$, and we take $\mathbf{P} = p\,\mathbf{I}$ and $\mathbf{\Gamma} = \mathbf{0}$. Since the output is a scalar real random variable in this case, we will focus on $\mathbf{y}_k$ for the sake of simplicity. In particular, we present the empirical distribution of $\mathbf{y}_k$ for several different values of k for the probabilities $p = \{0.3,\ 0.6,\ 0.9\}$ in Figure 4.5.



(a)



(b)



(c)

Figure 4.5: Empirical distributions of the output random variable for the update probabilities (a) $p = 0.3$, (b) $p = 0.6$, (c) $p = 0.9$. Dotted lines denote the mean of the corresponding random variables. Distributions are obtained via 10^9 independent samples of the random variables.

Numerical results in Figure 4.5 show that the distributions are not necessarily symmetric with respect to their means, and they can be multimodal. So, it is clear that the random variable $\mathbf{y}_k$ does not have a Gaussian distribution in general.

In addition to the mean and the variance being a function of the iteration index k (see Theorem 4.1 and Corollary 4.2), Figure 4.5 shows that overall "shape" of the distributions also change with iterations. In particular, Figures 4.5b and 4.5c show that the distributions can be multimodal or unimodal depending on k. Similarly, update probabilities also affect the distributions. As discussed previously in Section 4.6.2, distributions tend to be "narrower" as the update probabilities get larger. However, the precise relationship between the update probabilities and the distributions is not know at this point.

4.6.4 Effect of the Stochastic Model

Although linear systems show an unexpected behavior under randomized asynchronicity, precise details of the stochastic model are also important in determining the mean-squared stability. The results presented in this chapter are valid only for the stochastic index selection model described in Section 4.2.1, and the stability condition can be different under different models. In this section we will demonstrate this point.

We start by considering the model in Section 4.2.1 with all the indices being updated with probability μ/N, so that μ uniformly randomly selected indices are updated *per iteration on average*, that is, $\mathbb{E}[|\mathcal{T}_k|] = \mu$ for all k. More precisely,

$$\mathbf{P} = \frac{\mu}{N}\mathbf{I} \quad \Longrightarrow \quad \bar{\mathbf{A}} = \mathbf{I} + \frac{\mu}{N}\big(\mathbf{A} - \mathbf{I}\big). \tag{4.76}$$

In this case, the matrix $\mathbf{\Phi}$ in (4.38) has the following form:

$$\mathbf{\Phi} = \bar{\mathbf{A}}^* \otimes \bar{\mathbf{A}} + \frac{\mu(N-\mu)}{N^2}\,\mathbf{J}\left((\mathbf{A}^* - \mathbf{I}) \otimes (\mathbf{A} - \mathbf{I})\right). \tag{4.77}$$

We now consider a slightly different model where we update *exactly μ indices per iteration*. So, we have $|\mathcal{T}_k| = \mu$ for all k, and the set $\mathcal{T}_k$ is selected uniformly randomly among all possible $\binom{N}{\mu}$ subsets of size μ. In this case the average state transition matrix $\bar{\mathbf{A}}$ still has the form in (4.76). However, it can be shown (using the identity (2.2)) that the error correlation matrix evolves according to the following function:

$$\begin{aligned}\varphi'(\mathbf{X}) = \bar{\mathbf{A}}\,\mathbf{X}\,\bar{\mathbf{A}}^{\mathrm{H}} &- \frac{\mu(N-\mu)}{N^2(N-1)}\,(\mathbf{A} - \mathbf{I})\,\mathbf{X}\,(\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \\ &+ \frac{\mu(N-\mu)}{N(N-1)}\left((\mathbf{A} - \mathbf{I})\,\mathbf{X}\,(\mathbf{A}^{\mathrm{H}} - \mathbf{I})\right) \odot \mathbf{I}.\end{aligned} \tag{4.78}$$

By vectorizing both sides of (4.78), matrix representation of the function $\varphi'(\cdot)$ can be found as follows:

$$\mathbf{\Phi}' = \bar{\mathbf{A}}^* \otimes \bar{\mathbf{A}} + \frac{\mu(N-\mu)}{N(N-1)}\left(\mathbf{J} - \frac{1}{N}\mathbf{I}\right)\left((\mathbf{A}^* - \mathbf{I}) \otimes (\mathbf{A} - \mathbf{I})\right). \tag{4.79}$$

It is clear that $\mathbf{\Phi}$ and $\mathbf{\Phi}'$ are different from each other although the difference $\mathbf{\Phi} - \mathbf{\Phi}'$ approaches zero as N gets larger. More importantly, there is no clear relation between $\rho(\mathbf{\Phi})$ and $\rho(\mathbf{\Phi}')$. Thus, random asynchronous updates may be stable under one stochastic model, but the iterations may get unstable under the other model although both models update μ uniformly selected random indices per iteration on average.

4.7 Concluding Remarks

In this chapter, we studied a randomized asynchronous variant of the discrete time-invariant state-space models, in which a state variable is updated with some probability independently and asynchronously (with respect to the others) in each iteration. We showed that the randomized model can be treated as a linear time-invariant system (with a frequency response in the expectation sense) despite its randomly time-varying behavior. We presented the necessary and sufficient condition for the mean-squared stability of the randomized state recursions and showed that stability of the underlying state transition matrix is neither necessary nor sufficient for the mean-squared stability. So, an unstable system (in the synchronous world) may get stable with randomized asynchronicity. We showed that the mean-squared stability of a randomized system can be altered by a similarity transformation.

We showed that the randomization error depends on the amount of variation in the input signal as well as the update probabilities. We showed that random state variables do not follow a Gaussian distribution in general. Precise analysis of these observations will be considered in future work. For future studies, it is also interesting to study the probabilities that are optimal in terms of minimizing the effect of the randomization error, the input noise, or the rate of convergence. In addition, similarity transforms that enable the mean-square stability in the randomized setting are worth studying in future.

4.8 Appendices

4.8.1 A Result on The Random Index Selections

Lemma 4.6. *For an arbitrary matrix $\mathbf{X} \in \mathbb{C}^{N \times N}$, the following identities hold true:*

$$\mathbb{E}\big[\mathbf{D}_{\mathcal{T}_k}\,\mathbf{X}\big] = \mathbf{P}\,\mathbf{X}, \tag{4.80}$$

$$\mathbb{E}\big[\mathbf{D}_{\mathcal{T}_k}\,\mathbf{X}\,\mathbf{D}_{\mathcal{T}_k}\big] = \mathbf{P}\,\mathbf{X}\,\mathbf{P} + \mathbf{X} \odot \big(\mathbf{P} - \mathbf{P}^2\big), \tag{4.81}$$

where the expectations are taken with respect to the random subset $\mathcal{T}_k$ under the stochastic model in (4.15)*, and $\odot$ denotes the Hadamard (element-wise) product.*

Proof. The identity in (4.80) follows directly from the linearity of the expectation and the definition of **P** in (4.16).

For the identity (4.81), we first write the following:

$$\big(\mathbf{D}_{\mathcal{T}_k}\,\mathbf{X}\,\mathbf{D}_{\mathcal{T}_k}\big)_{i,j} = \begin{cases} X_{i,j}, & i \in \mathcal{T}_k, \quad j \in \mathcal{T}_k, \\ 0, & \text{otherwise}. \end{cases} \tag{4.82}$$

Thus, we can write the following due to the binary nature of the index selections:

$$\mathbb{E}\Big[\big(\mathbf{D}_{\mathcal{T}_k}\ \mathbf{X}\ \mathbf{D}_{\mathcal{T}_k}\big)_{i,j}\Big] = X_{i,j}\ \mathbb{P}[i \in \mathcal{T}_k,\ j \in \mathcal{T}_k]. \tag{4.83}$$

Regarding the probabilities in (4.83), we have the following for the non-diagonal $(i \neq j)$ entries:

$$\mathbb{P}[i \in \mathcal{T}_k,\ j \in \mathcal{T}_k] = \mathbb{P}[i \in \mathcal{T}_k]\ \mathbb{P}[j \in \mathcal{T}_k] = p_i\ p_j, \tag{4.84}$$

which follows from the fact that indices get updated independently from each other. On the other hand, we have the following for the diagonal $(i = j)$ entries:

$$\mathbb{P}[i \in \mathcal{T}_k,\ j \in \mathcal{T}_k] = \mathbb{P}[i \in \mathcal{T}_k] = p_i, \tag{4.85}$$

which follows simply from the fact that $i \in \mathcal{T}_k$ if and only if $j \in \mathcal{T}_k$ when $i = j$.

Thus, we can write the following:

$$\Big(\mathbb{E}\big[\mathbf{D}_{\mathcal{T}_k}\ \mathbf{X}\ \mathbf{D}_{\mathcal{T}_k}\big]\Big)_{i,j} = \begin{cases} p_i\ p_j\ X_{i,j} & i \neq j, \\ p_i\ X_{i,i}, & i = j, \end{cases} \tag{4.86}$$

which is equivalent to the identity (4.81). □

4.8.2 Proof of Theorem 4.1

Due to asynchronous updates described in (4.13), state vector $\mathbf{x}_k$ can be written as follows:

$$\begin{aligned} \mathbf{x}_k &= (\mathbf{I} - \mathbf{D}_{\mathcal{T}_k})\ \mathbf{x}_{k\text{-}1} + \mathbf{D}_{\mathcal{T}_k}\ \Big(\mathbf{A}\ \mathbf{x}_{k\text{-}1} + \sum_{i=1}^{R} \mathbf{B}_i\ \alpha_i^{k\text{-}1} + \mathbf{w}_{k\text{-}1}\Big) \\ &= \big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_k}\ (\mathbf{A} - \mathbf{I})\big)\mathbf{x}_{k\text{-}1} + \mathbf{D}_{\mathcal{T}_k}\left(\sum_{i=1}^{R} \mathbf{B}_i\ \alpha_i^{k\text{-}1} + \mathbf{w}_{k\text{-}1}\right). \end{aligned} \tag{4.87}$$

Taking expectation of (4.87) and using the facts that updated indices are selected independently, the input noise has zero mean, and the noise is uncorrelated with the index selections, we have the following:

$$\mathbb{E}[\mathbf{x}_k] = \bar{\mathbf{A}}\ \mathbb{E}[\mathbf{x}_{k\text{-}1}] + \sum_{i=1}^{R} \bar{\mathbf{B}}_i\ \alpha_i^{k\text{-}1} = \bar{\mathbf{A}}^k\ \mathbb{E}[\mathbf{x}_0] + \sum_{n=0}^{k-1} \bar{\mathbf{A}}^n\ \sum_{i=1}^{R} \bar{\mathbf{B}}_i\ \alpha_i^{k\text{-}1\text{-}n} \tag{4.88}$$

$$= \sum_{i=1}^{R} (\alpha_i \mathbf{I} - \bar{\mathbf{A}})^{\text{-}1} \bar{\mathbf{B}}_i\ \alpha_i^k + \bar{\mathbf{A}}^k\Big(\mathbb{E}[\mathbf{x}_0] - \sum_{i=1}^{R} (\alpha_i \mathbf{I} - \bar{\mathbf{A}})^{\text{-}1} \bar{\mathbf{B}}_i\Big),$$

$$= \mathbf{x}_k^{\text{ss}} + \bar{\mathbf{A}}^k\Big(\mathbb{E}[\mathbf{x}_0] - \mathbf{x}_0^{\text{ss}}\Big) = \mathbf{x}_k^{\text{ss}} + \mathbf{x}_k^{\text{tr}} \tag{4.89}$$

where $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ are as in (4.19).

4.8.3 Proof of Theorem 4.2

Using the definition of the error term in (4.29) and substituting $\mathbf{x}_k = \mathbf{q}_k + \mathbf{x}_k^{\text{ss}}$ into (4.87), we have the following:

$$\mathbf{q}_{k+1} + \mathbf{x}_{k+1}^{\text{ss}} = \big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_{k+1}} (\mathbf{A} - \mathbf{I})\big)\ (\mathbf{q}_k + \mathbf{x}_k^{\text{ss}}) + \mathbf{D}_{\mathcal{T}_{k+1}}\ \mathbf{w}_k + \mathbf{D}_{\mathcal{T}_{k+1}} \mathbf{P}^{-1} \sum_{i=1}^{R} \bar{\mathbf{B}}_i\, \alpha_i^k, \tag{4.90}$$

which can be written as follows by rearranging the terms:

$$\mathbf{q}_{k+1} = \big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_{k+1}} (\mathbf{A} - \mathbf{I})\big)\ \mathbf{q}_k + \mathbf{D}_{\mathcal{T}_{k+1}}\ \mathbf{w}_k + \big(\mathbf{D}_{\mathcal{T}_{k+1}} \mathbf{P}^{-1} - \mathbf{I}\big)\, \boldsymbol{\delta}_k, \tag{4.91}$$

where the *deterministic* vector $\boldsymbol{\delta}_k$ is defined as in (4.36).

Using (4.91), we can express the outer product $\mathbf{q}_{k+1}\, \mathbf{q}_{k+1}^{\text{H}}$ recursively in terms of $\mathbf{q}_k\, \mathbf{q}_k^{\text{H}}$ as follows:

$$\begin{aligned}\mathbf{q}_{k+1}\, \mathbf{q}_{k+1}^{\text{H}} = {} & \big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_{k+1}} (\mathbf{A} - \mathbf{I})\big)\ \mathbf{q}_k \mathbf{q}_k^{\text{H}} \big((\mathbf{A}^{\text{H}} - \mathbf{I}) \mathbf{D}_{\mathcal{T}_{k+1}} + \mathbf{I}\big) \\ & + \mathbf{D}_{\mathcal{T}_{k+1}}\, \mathbf{w}_k\ \mathbf{w}_k^{\text{H}}\, \mathbf{D}_{\mathcal{T}_{k+1}} \\ & + \big(\mathbf{D}_{\mathcal{T}_{k+1}}\, \mathbf{P}^{-1} - \mathbf{I}\big)\ \boldsymbol{\delta}_k\ \boldsymbol{\delta}_k^{\text{H}}\ \big(\mathbf{P}^{-1}\, \mathbf{D}_{\mathcal{T}_{k+1}} - \mathbf{I}\big) \\ & + \big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_{k+1}} (\mathbf{A} - \mathbf{I})\big)\ \mathbf{q}_k\ \boldsymbol{\delta}_k^{\text{H}} \big(\mathbf{P}^{-1}\, \mathbf{D}_{\mathcal{T}_{k+1}} - \mathbf{I}\big) \\ & + \big(\mathbf{D}_{\mathcal{T}_{k+1}} \mathbf{P}^{-1} - \mathbf{I}\big)\ \boldsymbol{\delta}_k\ \mathbf{q}_k^{\text{H}}\ \big(\mathbf{I} + (\mathbf{A}^{\text{H}} - \mathbf{I})\, \mathbf{D}_{\mathcal{T}_{k+1}}\ \big),\end{aligned} \tag{4.92}$$

where the cross terms including $\mathbf{w}_k$ are left-out intentionally because these terms will disappear when we take the expectation since $\mathbf{w}_k$ has a zero mean and it is uncorrelated with the index selections.

We now take the expectation of both sides of (4.92) and use the identities given by Lemma 4.6, and the independence assumption regarding the index selections, input noise and the initial condition. Then, we obtain the following:

$$\begin{aligned}\mathbf{Q}_{k+1} = {} & \varphi(\mathbf{Q}_k) + \mathbf{P}\, \boldsymbol{\Gamma}\, \mathbf{P} + \boldsymbol{\Gamma} \odot \big(\mathbf{P} - \mathbf{P}^2\big) + \big(\boldsymbol{\delta}_k\ \boldsymbol{\delta}_k^{\text{H}}\big) \odot \big(\mathbf{P}^{-1} - \mathbf{I}\big) \\ & + \Big((\bar{\mathbf{A}} - \mathbf{I})\ \mathbf{x}_k^{\text{tr}}\ \boldsymbol{\delta}_k^{\text{H}}\Big) \odot \big(\mathbf{P}^{-1} - \mathbf{I}\big) + \Big(\boldsymbol{\delta}_k\ (\mathbf{x}_k^{\text{tr}})^{\text{H}}\ (\bar{\mathbf{A}}^{\text{H}} - \mathbf{I})\Big) \odot \big(\mathbf{P}^{-1} - \mathbf{I}\big),\end{aligned} \tag{4.93}$$

where the function $\varphi(\cdot)$ is defined in (4.33). We also note that $\mathbb{E}[\mathbf{q}_k] = \mathbf{x}_k^{\text{tr}}$ as given in (4.35).

Although $\mathbf{X} + \mathbf{X}^{\text{H}} \neq 2\, \Re\{\mathbf{X}\}$ in general, we note that the following equality holds true for any $\mathbf{X} \in \mathbb{C}^{N \times N}$:

$$\Big(\mathbf{X} + \mathbf{X}^{\text{H}}\Big) \odot \big(\mathbf{P}^{-1} - \mathbf{I}\big) = 2\, \Re\big\{\mathbf{X}\big\} \odot \big(\mathbf{P}^{-1} - \mathbf{I}\big), \tag{4.94}$$

where $\Re\{\cdot\}$ denotes the real part of its argument. So, using the identity (4.94) in (4.93) gives the result in (4.35).

4.8.4 Proof of Corollary 4.2

We first define the following:

$$\mathbf{Z}_k = \mathbf{Q}_k - \mathbf{Q}_k^{\mathrm{r}} - \mathbf{Q}^{\mathrm{n}}, \tag{4.95}$$

where $\mathbf{Q}_k^{\mathrm{r}}$ and $\mathbf{Q}^{\mathrm{n}}$ are given as the solutions of (4.44) and (4.43), respectively. Substituting (4.95) into the recursion (4.35), we get

$$\begin{aligned}\mathbf{Z}_{k+1} + \mathbf{Q}_{k+1}^{\mathrm{r}} + \mathbf{Q}^{\mathrm{n}} = \varphi(\mathbf{Z}_k) + \varphi(\mathbf{Q}_k^{\mathrm{r}}) + \varphi(\mathbf{Q}^{\mathrm{n}}) \\ + \mathbf{P}\,\mathbf{\Gamma}\,\mathbf{P} + \mathbf{\Gamma} \odot \left(\mathbf{P} - \mathbf{P}^2\right) + \Re\left\{\delta_k \delta_k^{\mathrm{H}}\right\} \odot \left(\mathbf{P}^{-1} - \mathbf{I}\right) \\ + \Re\left\{2\,(\bar{\mathbf{A}} - \mathbf{I})\,\mathbf{x}_k^{\mathrm{tr}}\,\delta_k^{\mathrm{H}}\right\} \odot \left(\mathbf{P}^{-1} - \mathbf{I}\right),\end{aligned} \tag{4.96}$$

which can be simplified as follows due to the defining equations in (4.44) and (4.43):

$$\mathbf{Z}_{k+1} = \varphi(\mathbf{Z}_k) + \Re\left\{2\,(\bar{\mathbf{A}} - \mathbf{I})\,\mathbf{x}_k^{\mathrm{tr}}\,\delta_k^{\mathrm{H}}\right\} \odot \left(\mathbf{P}^{-1} - \mathbf{I}\right). \tag{4.97}$$

Due to the stability assumption (4.41) and Lemma 4.3 we have $\rho(\bar{\mathbf{A}}) < 1$, so $\lim_{k\to\infty} \mathbf{x}_k^{\mathrm{tr}} = \mathbf{0}$. As a result,

$$\lim_{k\to\infty} \mathbf{Z}_k = \mathbf{0}, \tag{4.98}$$

which gives the desired result.

Necessity of the condition (4.41) follows from (4.40). That is, when $\rho(\mathbf{\Phi}) \geq 1$ there exists a nonzero positive semi-definite matrix $\mathbf{X}$ that cannot be reduced by the function $\varphi(\cdot)$.

4.8.5 Proof of Lemma 4.1

Assume that the stability condition (4.41) is met, and solution to (4.43) exists. Let $\mathbf{e}_i$ denote the i^{th} standard basis vector. It is readily verified that the following identity holds true for any $\mathbf{X} \in \mathbb{C}^{N\times N}$ and any index $1 \leq i \leq N$:

$$\mathbf{e}_i^{\mathrm{H}}\ \varphi(\mathbf{X})\ \mathbf{e}_i = (1 - p_i)\ \mathbf{e}_i^{\mathrm{H}}\,\mathbf{X}\,\mathbf{e}_i + p_i\,\mathbf{e}_i^{\mathrm{H}}\,\mathbf{A}\,\mathbf{X}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{e}_i. \tag{4.99}$$

Furthermore, we have the following:

$$\mathbf{e}_i^{\mathrm{H}}\left(\mathbf{P}\,\mathbf{\Gamma}\,\mathbf{P} + \mathbf{\Gamma} \odot \left(\mathbf{P} - \mathbf{P}^2\right)\right)\mathbf{e}_i = p_i\ \mathbf{e}_i^{\mathrm{H}}\,\mathbf{\Gamma}\,\mathbf{e}_i. \tag{4.100}$$

So, by left and right multiplying (4.43) with $\mathbf{e}_i^{\mathrm{H}}$ and $\mathbf{e}_i$ respectively, we get the following:

$$\mathbf{e}_i^{\mathrm{H}}\left(\mathbf{Q}^{\mathrm{n}} - \mathbf{\Gamma}\right)\mathbf{e}_i\ p_i = p_i\ \mathbf{e}_i^{\mathrm{H}}\,\mathbf{A}\,\mathbf{Q}^{\mathrm{n}}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{e}_i \geq 0, \tag{4.101}$$

where the inequality follows from $\mathbf{Q}^{\mathrm{n}} \geq \mathbf{0}$, and the desired result follows from the fact that $p_i > 0$ for all i.

4.8.6 Proof of Lemma 4.2

Enumerate all possible index update sets. Namely, let $\mathcal{T}_j$ denote the j^{th} update set, and let γ_j denote the probability of selecting the set $\mathcal{T}_j$ for $1 \leq j \leq 2^N$. So, the system is equivalent to the following state transition matrix:

$$\mathbf{A}_j = \mathbf{I} + \mathbf{D}_{\mathcal{T}_j} \, (\mathbf{A} - \mathbf{I}) \tag{4.102}$$

which now depends on the index j. This has probability γ_j in model (4.50), where the probability is given as follows (due to the stochastic model in (4.15)):

$$\gamma_j = \Bigg(\prod_{i \in \mathcal{T}_j} p_i \Bigg) \Bigg(\prod_{i \notin \mathcal{T}_j} (1 - p_i) \Bigg). \tag{4.103}$$

Then, for a given $\mathbf{X} \in \mathbb{C}^{N \times N}$ we have the following:

$$\begin{aligned}
&\sum_{j=1}^{2^N} \gamma_j \, \mathbf{A}_j \, \mathbf{X} \, \mathbf{A}_j^{\mathrm{H}} = \sum_{j=1}^{2^N} \gamma_j \, \Big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_j} \, (\mathbf{A} - \mathbf{I}) \Big) \, \mathbf{X} \, \Big(\mathbf{I} + (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{D}_{\mathcal{T}_j} \Big) \\
&= \mathbb{E}\Big[\Big(\mathbf{I} + \mathbf{D}_{\mathcal{T}_j} \, (\mathbf{A} - \mathbf{I}) \Big) \, \mathbf{X} \, \Big(\mathbf{I} + (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{D}_{\mathcal{T}_j} \Big) \Big] \\
&= \mathbf{X} + \mathbf{P} \, (\mathbf{A} - \mathbf{I}) \, \mathbf{X} + \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{P} + \mathbb{E}\Big[\mathbf{D}_{\mathcal{T}_j} \, (\mathbf{A} - \mathbf{I}) \, \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{D}_{\mathcal{T}_j} \Big] \qquad (4.104) \\
&= \mathbf{X} + \mathbf{P} \, (\mathbf{A} - \mathbf{I}) \, \mathbf{X} + \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{P} + \mathbf{P} \, (\mathbf{A} - \mathbf{I}) \, \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{P} \\
&\quad + \Big((\mathbf{A} - \mathbf{I}) \, \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \Big) \odot (\mathbf{P} - \mathbf{P}^2) \\
&= \Big(\mathbf{I} + \mathbf{P} \, (\mathbf{A} - \mathbf{I}) \Big) \, \mathbf{X} \, \Big(\mathbf{I} + (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \, \mathbf{P} \Big) + \Big((\mathbf{A} - \mathbf{I}) \, \mathbf{X} \, (\mathbf{A}^{\mathrm{H}} - \mathbf{I}) \Big) \odot (\mathbf{P} - \mathbf{P}^2) \\
&= \varphi(\mathbf{X}),
\end{aligned}$$

where we use the identity (4.81) in (4.104). Thus, the statement of the corollary follows directly from [42, Corollary 3.26].

4.8.7 Proof of Corollary 4.3

Using the recursive definition of the error correlation matrix in (4.35), we have the following regarding the trace of the error correlation matrix:

$$\begin{aligned}
\mathrm{tr}(\mathbf{Q}_{k+1}) &= \mathrm{tr}(\varphi(\mathbf{Q}_k)) + \mathrm{tr}(\mathbf{P} \, \mathbf{\Gamma}) + \mathrm{tr} \Big((\boldsymbol{\delta}_k \, \boldsymbol{\delta}_k^{\mathrm{H}}) \, (\mathbf{P}^{-1} - \mathbf{I}) \Big) \\
&\quad + \mathrm{tr} \Big(\Re\Big\{ 2 \, (\bar{\mathbf{A}} - \mathbf{I}) \, \mathbf{x}_k^{\mathrm{tr}} \, \boldsymbol{\delta}_k^{\mathrm{H}} \Big\} \, (\mathbf{P}^{-1} - \mathbf{I}) \Big),
\end{aligned} \tag{4.105}$$

where we use the fact that $\mathrm{tr}(\mathbf{X} \odot \mathbf{D}) = \mathrm{tr}(\mathbf{X}\mathbf{D})$ holds true for any diagonal matrix $\mathbf{D}$.

Next, we will provide bounds for individual elements on the right-hand-side of (4.105). We first get the following trace equality by summing (4.99) over all indices:

$$\mathrm{tr}\,\big(\varphi(\mathbf{X})\big) = \mathrm{tr}\,\Big(\mathbf{X} \, \big(\mathbf{I} + \mathbf{A}^{\mathrm{H}} \, \mathbf{P} \, \mathbf{A} - \mathbf{P} \big) \Big). \tag{4.106}$$

Now define Ψ as follows:

$$\Psi = \lambda_{\max}\big(\mathbf{I} + \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P}\big). \tag{4.107}$$

Then we have the following:

$$\Psi\,\mathbf{I} \geq \mathbf{I} + \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} - \mathbf{P} = \bar{\mathbf{A}}^{\mathrm{H}}\,\bar{\mathbf{A}} + (\mathbf{A}^{\mathrm{H}} - \mathbf{I})(\mathbf{P} - \mathbf{P}^2)(\mathbf{A} - \mathbf{I}) \geq \bar{\mathbf{A}}^{\mathrm{H}}\,\bar{\mathbf{A}}, \tag{4.108}$$

which implies the following:

$$\|\bar{\mathbf{A}}\|_2 \leq \Psi^{1/2} \qquad \text{and} \qquad \operatorname{tr}\big(\varphi(\mathbf{X})\big) \leq \Psi\,\operatorname{tr}(\mathbf{X}). \tag{4.109}$$

Using the bound in (4.53), we can write the following:

$$\operatorname{tr}\Big(\big(\boldsymbol{\delta}_k\,\boldsymbol{\delta}_k^{\mathrm{H}}\big)\big(\mathbf{P}^{-1} - \mathbf{I}\big)\Big) = \boldsymbol{\delta}_k^{\mathrm{H}}\,\left(\mathbf{P}^{-1} - \mathbf{I}\right)\,\boldsymbol{\delta}_k \leq \boldsymbol{\Delta}^2\,\left\|\mathbf{P}^{-1} - \mathbf{I}\right\|_2. \tag{4.110}$$

For the last term on the right-hand-side of (4.105), we can write the following inequalities:

$$\begin{aligned}\operatorname{tr}\Big(\Re\Big\{2\,(\bar{\mathbf{A}} - \mathbf{I})\,\mathbf{x}_k^{\mathrm{tr}}\,\boldsymbol{\delta}_k^{\mathrm{H}}\Big\}\big(\mathbf{P}^{-1} - \mathbf{I}\big)\Big) &= \Re\Big\{2\,\boldsymbol{\delta}_k^{\mathrm{H}}\,\left(\mathbf{P}^{-1} - \mathbf{I}\right)\,(\bar{\mathbf{A}} - \mathbf{I})\,\mathbf{x}_k^{\mathrm{tr}}\Big\} \\ &\leq \Big|2\,\boldsymbol{\delta}_k^{\mathrm{H}}\,(\mathbf{I} - \mathbf{P})(\mathbf{A} - \mathbf{I})\,\bar{\mathbf{A}}^k\,\left(\mathbb{E}[\mathbf{x}_0] - \mathbf{x}_0^{\mathrm{ss}}\right)\Big| \leq c\,\Psi^{k/2},\end{aligned} \tag{4.111}$$

where (4.111) follows from the definition of $\mathbf{x}_k^{\mathrm{tr}}$ in (4.18), and the constant c in (4.111) is given as follows:

$$c = 2\,\boldsymbol{\Delta}\,\left\|(\mathbf{I} - \mathbf{P})(\mathbf{A} - \mathbf{I})\right\|_2\,\left\|\mathbb{E}[\mathbf{x}_0] - \mathbf{x}_0^{\mathrm{ss}}\right\|_2, \tag{4.112}$$

where we use the bounds from (4.53) and (4.109).

Using the bounds in (4.109), (4.110), and (4.111) in the equality (4.105), we get the following:

$$\operatorname{tr}(\mathbf{Q}_{k+1}) \leq \Psi\,\operatorname{tr}(\mathbf{Q}_k) + \operatorname{tr}(\mathbf{P}\,\boldsymbol{\Gamma}) + \boldsymbol{\Delta}^2\,\|\mathbf{P}^{-1} - \mathbf{I}\|_2 + c\,\Psi^{k/2}. \tag{4.113}$$

Using the inequality (4.113) recursively, we get the following:

$$\begin{aligned}\operatorname{tr}(\mathbf{Q}_k) \leq{}& \Psi^k\,\operatorname{tr}(\mathbf{Q}_0) + c\,\big(\Psi^{k/2} - \Psi^k\big)/\big(\Psi^{1/2} - \Psi\big) \\ &+ \frac{1 - \Psi^k}{1 - \Psi}\Big(\operatorname{tr}(\mathbf{P}\,\boldsymbol{\Gamma}) + \boldsymbol{\Delta}^2\,\|\mathbf{P}^{-1} - \mathbf{I}\|_2\Big).\end{aligned} \tag{4.114}$$

In the final step we use the assumption $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}$ in (4.51), and conclude from (4.107) that $\Psi < 1$. So, the upper bound in (4.114) gives the following result:

$$\limsup_{k\to\infty}\ \operatorname{tr}(\mathbf{Q}_k) \leq \frac{\operatorname{tr}(\mathbf{P}\,\boldsymbol{\Gamma}) + \boldsymbol{\Delta}^2\,\|\mathbf{P}^{-1} - \mathbf{I}\|_2}{1 - \Psi}, \tag{4.115}$$

which is equivalent to (4.52) since $1 - \Psi = \lambda_{\min}(\mathbf{P} - \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A})$.

4.8.8 Proof of Lemma 4.5

Assume that $\mathbf{A} \in \mathbb{C}^{N \times N}$ is a triangular matrix with $A_{i,i}$ being the i^{th} diagonal entry. Then, we first note that both $\bar{\mathbf{A}}$ and $\mathbf{\Phi}$ are triangular matrices, which follows simply from the fact that $\mathbf{P}$ and $\mathbf{J}$ are diagonal matrices, and the Kronecker product of triangular matrices is a triangular matrix. In particular, we note that the i^{th} diagonal entry of $\bar{\mathbf{A}}$ is as follows:

$$\bar{A}_{i,i} = \Big(1 + p_i \, (A_{i,i} - 1)\Big). \tag{4.116}$$

Then, it is straightforward to verify that the k^{th} diagonal entry of $\mathbf{\Phi}$ is as follows for $k = i + (j-1)\,N$ with $1 \leq i, j \leq N$:

$$S_{k,k} = \bar{A}^*_{j,j} \; \bar{A}_{i,i} + \delta_{i,j} \; (p_i - p_i^2) \, |A_{i,i} - 1|^2, \tag{4.117}$$

where $\delta_{i,j} = 1$ if and only if $i = j$.

Since $\mathbf{\Phi}$ is triangular, the diagonal entries of $\mathbf{\Phi}$ correspond to the eigenvalues of $\mathbf{\Phi}$. Thus,

$$\rho(\mathbf{\Phi}) = \max_{1 \leq k \leq N^2} |S_{k,k}| = \max_{1 \leq i,j \leq N} \left| \bar{A}^*_{j,j} \; \bar{A}_{i,i} + \delta_{i,j} \; (p_i - p_i^2) \, |A_{i,i} - 1|^2 \right| \tag{4.118}$$

$$= \max_{1 \leq i \leq N} |\bar{A}_{i,i}|^2 + (p_i - p_i^2) \, |A_{i,i} - 1|^2 = \max_{1 \leq i \leq N} 1 + p_i \; \big(|A_{i,i}|^2 - 1\big). \tag{4.119}$$

We now prove the equivalence in (4.64). Assume that $\mathbf{A}$ is a stable matrix. Since $\mathbf{A}$ is a triangular matrix, its diagonal entries are the eigenvalues. Thus,

$$\rho(\mathbf{A}) < 1 \quad \Longrightarrow \quad |A_{i,i}| < 1 \quad \forall i \quad \Longrightarrow \quad \rho(\mathbf{\Phi}) < 1, \tag{4.120}$$

where the last implication follows from (4.119).

For the converse direction,

$$\rho(\mathbf{A}) \geq 1 \quad \Longrightarrow \quad \exists i \; \text{ s.t. } \; |A_{i,i}| \geq 1 \quad \Longrightarrow \quad \rho(\mathbf{\Phi}) \geq 1, \tag{4.121}$$

where the last implication follows from (4.119).

Chapter 5

RANDOMIZED ALGORITHMS AS SWITCHING SYSTEMS

5.1 Introduction

Solving linear systems of equations is an important task in numerical analysis, and it finds application in almost every field of science and engineering. Recent years have witnessed an elevated interest in randomized methods that iterate over the given data randomly in order to obtain the desired solution. Examples include the randomized variants of the Kaczmarz's method [165], Gauss-Seidel iterations [107], fixed-point iterations [9], and many others [141, 72, 55, 115]. These type of methods are better suited for distributed processing of large scaled data, and they may perform better than their non-random counter-parts in certain settings (see Chapter 2).

Control theory has a rich literature regarding systems whose dynamics change over time. One way to model such a behavior is to assume that the system "switches" between different operational modes. Compared to the standard state-space formulations, this model offers a great flexibility due to its rich and complex dynamics, and there are well-studied problems in the random and non-random switching settings [166, 42, 92].

Although solving linear system of equations and control of dynamical systems appear to be different problems, the purpose of this chapter is to show that randomized iterative algorithms can be regarded as randomly switching systems. So, one can utilize the already available stability theory of switching systems in order to study such randomized techniques. As an example, we will reformulate the Kaczmarz method, Gauss-Seidel iterations and asynchronous fixed-point iterations as switching systems, and then provide "simple" proofs for their convergence. We will also discuss the convergence of randomized asynchronous fixed-point iterations, which has been of interest in recent years for graph signal processing applications (see Chapters 2 and 3 of this thesis). While the convergence of these randomized techniques are already known, the main contribution of this chapter is to show that seemingly different randomized techniques can be unified under the viewpoint of switching systems.

5.1.1 Scope and Outline

In Section 5.2 we will briefly review discrete-time switching systems by showing their expected behavior (Lemma 5.1), mean-squared stability (Lemma 5.2), and alternative characterizations for the stability (Lemma 5.3). We will discuss randomized Kaczmarz method, Gauss-Seidel method, and asynchronous fixed-point iterations from the viewpoint of switching systems in Sections 5.3.1, 5.3.2, and 5.3.3, respectively. In these sections, we will also provide alternative proofs for the convergence of the randomized Kaczmarz (Corollary 5.1) and randomized Gauss-Seidel (Corollary 5.2) methods.

5.2 Discrete-Time Randomly Switching Systems

Assume that we are given a set of L state-transition matrices and input signals, denoted by $\mathcal{A}$ and $\mathcal{U}$, respectively:

$$\mathcal{A} = \{\mathbf{A}_1, \cdots, \mathbf{A}_L\}, \qquad\qquad \mathcal{U} = \{\mathbf{u}_1, \cdots, \mathbf{u}_L\}, \tag{5.1}$$

where $\mathbf{A}_j \in \mathbb{C}^{N\times N}$ and $\mathbf{u}_j \in \mathbb{C}^N$ for all $1 \leq j \leq L$. We consider the following switching model:

$$\mathbf{x}_{k+1} = \mathbf{A}_{i_k}\, \mathbf{x}_k + \mathbf{u}_{i_k} + \mathbf{w}_k. \tag{5.2}$$

So, the vector $\mathbf{x}_k \in \mathbb{C}^N$ is updated with a different pair of state-transition matrix and input signal in every iteration, and i_k denotes the index of the state-transition matrix and the input signal used in the k^{th} iteration. So, $1 \leq i_k \leq L$. The term $\mathbf{w}_k \in \mathbb{C}^N$ denotes noise with the following statistics:

$$\mathbb{E}[\mathbf{w}_k] = \mathbf{0}, \qquad \mathbb{E}\big[\mathbf{w}_k\, \mathbf{w}_s^{\rm H}\big] = \delta(k-s)\, \mathbf{\Gamma}, \tag{5.3}$$

where $\delta(\cdot)$ denotes the discrete Dirac delta function, and $\mathbf{\Gamma}$ is allowed to be non-diagonal.

In this chapter, we consider the model (5.2) in a randomized setting where the index i_k is selected randomly and independently in each iteration. That is, the vector $\mathbf{x}_k$ is updated with the pair $(\mathbf{A}_j, \mathbf{u}_j)$ with probability p_j for all k. More precisely,

$$\mathbb{P}\big[i_k = j\big] = p_j \qquad \forall\, k \geq 0, \quad 1 \leq j \leq L. \tag{5.4}$$

It is possible to extend the randomly switching model to a setting in which the index i_k is determined by a Markov chain. This is a well-studied model, but we will not review the details of the Markovian extension in this short chapter. We refer to [42] for a rigorous treatment of the topic.

We first establish the expected behavior in the model (5.2):

Lemma 5.1. *Expectation of the random vector* $\mathbf{x}_k$ *in the randomly switching model* (5.2) *is as follows:*

$$\mathbb{E}[\mathbf{x}_k] = \bar{\mathbf{x}} + \bar{\mathbf{A}}^k \left(\mathbb{E}[\mathbf{x}_0] - \bar{\mathbf{x}}\right), \tag{5.5}$$

where

$$\bar{\mathbf{A}} = \sum_{j=1}^{L} p_j \, \mathbf{A}_j, \qquad \bar{\mathbf{u}} = \sum_{j=1}^{L} p_j \, \mathbf{u}_j, \qquad \bar{\mathbf{x}} = (\mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{u}}. \tag{5.6}$$

Proof. We first write the expectation of the random vector $\mathbf{x}_k$ conditioned on the previous vector $\mathbf{x}_{k-1}$:

$$\mathbb{E}[\mathbf{x}_k \,|\, \mathbf{x}_{k-1}] = \sum_{j=1}^{L} p_j \left(\mathbf{A}_j \, \mathbf{x}_{k-1} + \mathbf{u}_j\right) = \bar{\mathbf{A}} \, \mathbf{x}_{k-1} + \bar{\mathbf{u}}, \tag{5.7}$$

where we use $\mathbb{E}[\mathbf{w}_k] = \mathbf{0}$, and the fact that pair $(\mathbf{A}_j, \mathbf{u}_j)$ is selected independently with probability p_j. Then, by taking one more expectation over the random vector $\mathbf{x}_{k-1}$ we get:

$$\begin{aligned}\mathbb{E}[\mathbf{x}_k] &= \bar{\mathbf{A}} \, \mathbb{E}[\mathbf{x}_{k-1}] + \bar{\mathbf{u}} = \bar{\mathbf{A}}^k \, \mathbb{E}[\mathbf{x}_0] + \sum_{n=0}^{k-1} \bar{\mathbf{A}}^n \, \bar{\mathbf{u}} \\ &= \bar{\mathbf{A}}^k \, \mathbb{E}[\mathbf{x}_0] + (\mathbf{I} - \bar{\mathbf{A}}^k) \, (\mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{u}}, \qquad (5.8) \\ &= (\mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{u}} + \bar{\mathbf{A}}^k \left(\mathbb{E}[\mathbf{x}_0] - (\mathbf{I} - \bar{\mathbf{A}})^{-1} \, \bar{\mathbf{u}}\right), \qquad (5.9)\end{aligned}$$

which is equivalent to (5.5). □

Here, $\bar{\mathbf{A}}$ will be referred to as *the average state-transition matrix*, and $\bar{\mathbf{u}}$ *the average input signal*. Thus, $\bar{\mathbf{x}}$ will be *the fixed-point of the average system*, i.e., $\bar{\mathbf{x}} = \bar{\mathbf{A}} \, \bar{\mathbf{x}} + \bar{\mathbf{u}}$.

From Lemma 5.1 it is clear that stability of the average state-transition matrix $\bar{\mathbf{A}}$ is both necessary and sufficient for the expectation of $\mathbf{x}_k$ to converge to $\bar{\mathbf{x}}$. That is,

$$\lim_{k \to \infty} \mathbb{E}[\mathbf{x}_k] = \bar{\mathbf{x}} \quad \iff \quad \rho\left(\bar{\mathbf{A}}\right) < 1. \tag{5.10}$$

Unlike its expectation, the random vector $\mathbf{x}_k$ itself may not converge to the point $\bar{\mathbf{x}}$ in general. This is because the fixed-point of the average system $\bar{\mathbf{x}}$ is not necessarily the fixed-point for all the individual systems. Namely, $\bar{\mathbf{x}} = \mathbf{A}_j \, \bar{\mathbf{x}} + \mathbf{u}_j$ may not hold true for all $1 \leq j \leq L$. When this happens, the random vector $\mathbf{x}_k$ "wanders around" the point $\bar{\mathbf{x}}$ as long as some stability conditions (to be stated next) are met. In order to describe this behavior more rigorously, we define the following error term and its (auto)-correlation matrix:

$$\mathbf{r}_k = \mathbf{x}_k - \bar{\mathbf{x}}, \qquad \mathbf{R}_k = \mathbb{E}\big[\mathbf{r}_k\, \mathbf{r}_k^{\mathrm{H}}\big]. \tag{5.11}$$

The following lemma describes the limiting behavior of the error correlation matrix:

Lemma 5.2. *The error correlation matrix converges to* $\mathbf{R}$*, i.e.,*

$$\lim_{k\to\infty} \mathbf{R}_k = \mathbf{R} \tag{5.12}$$

if the following condition holds true:

$$\rho(\mathbf{\Phi}) < 1 \qquad \text{where} \qquad \mathbf{\Phi} = \sum_{j=1}^{L} p_j\, \mathbf{A}_j^{*} \otimes \mathbf{A}_j. \tag{5.13}$$

Furthermore, the limit point $\mathbf{R}$ *is given as follows:*

$$\operatorname{vec}(\mathbf{R}) = (\mathbf{I} - \mathbf{\Phi})^{-1} \operatorname{vec}(\mathbf{Z} + \mathbf{\Gamma}), \tag{5.14}$$

where $\mathbf{\Gamma}$ *is the noise correlation matrix, and* $\mathbf{Z}$ *is as follows:*

$$\mathbf{Z} = \sum_{j=1}^{L} p_j\, \mathbf{z}_j\, \mathbf{z}_j^{\mathrm{H}}, \qquad \text{where} \qquad \mathbf{z}_j = \bar{\mathbf{x}} - \big(\mathbf{A}_j\, \bar{\mathbf{x}} + \mathbf{u}_j\big). \tag{5.15}$$

If the condition in (5.13) *is violated, then* $\mathbf{R}_k$ *increases unboundedly as* k *goes to infinity, that is,* $\mathbf{R}$ *is not bounded.*

Proof. See Section 5.5.1. □

The convergence of $\mathbf{R}_k$ means that the random vector $\mathbf{x}_k$ continues to have a finite mean-squared ℓ_2-distance to the point $\bar{\mathbf{x}}$ as the iterations progress. Thus, the condition in (5.13) is equivalent to the mean-squared stability of the switching system. In particular, if $\mathbf{R}_k$ converges to zero (i.e., $\mathbf{R} = \mathbf{0}$), then $\mathbf{x}_k$ converges to $\bar{\mathbf{x}}$ in the mean-squared sense. In fact, mean-squared convergence implies almost sure convergence as well in this setting [42, Corollary 3.46].

The matrix $\mathbf{Z}$ in (5.15) quantifies the mismatch between the individual systems $(\mathbf{A}_j, \mathbf{u}_j)$ and the average system $(\bar{\mathbf{A}}, \bar{\mathbf{u}})$. That is, $\mathbf{Z} = \mathbf{0}$ if and only if the fixed-point of the average system $\bar{\mathbf{x}}$ is also the fixed point of all the individual systems.

The condition in (5.13) precisely describes the mean-squared stability of the randomly switching system. Thus, the stability of a given system can be determined by checking the eigenvalues of the matrix $\mathbf{\Phi} \in \mathbb{C}^{N^2 \times N^2}$. Despite its numerical simplicity, this approach may not be feasible when analytically proving the stability of a randomized system. In this regard, the following lemma provides alternative characterizations of the mean-squared stability of a randomly switching system:

Lemma 5.3 (See [42, Corollary 3.26])**.** *The following statements are equivalent:*

- *The model in* (5.2) *is stable in the mean-squared sense,*
- $\rho(\mathbf{\Phi}) < 1$,
- *There exists a* $\mathbf{X} \succ \mathbf{0}$ *such that* $\mathbf{X} \succ \sum_{j=1}^{L} p_j \, \mathbf{A}_j \, \mathbf{X} \, \mathbf{A}_j^{\mathrm{H}}$,
- *For any* $\mathbf{Y} \succ \mathbf{0}$, *there exists a unique* $\mathbf{X} \succ \mathbf{0}$ *such that* $\mathbf{X} = \sum_{j=1}^{L} p_j \, \mathbf{A}_j \, \mathbf{X} \, \mathbf{A}_j^{\mathrm{H}} + \mathbf{Y}$.

In fact, we will use Lemma 5.3 in Section 5.3 when proving the convergence of the randomized Gauss-Seidel method.

5.2.1 Non-Random Case

The switching model in (5.2) can also be considered from a non-random viewpoint, in which the convergence is ensured for any sequence of index selection, which can be thought of as the *sure convergence* in the randomized setting.

In the non-random setting, the switching model converges if and only if *the joint spectral radius of the set* $\mathcal{A}$ is strictly less than unity [92, Corollary 1.1]. Note that the joint spectral radius of a set of matrices $\mathcal{A}$ is first defined in [148] as follows:

$$\rho(\mathcal{A}) = \lim_{k \to \infty} \max_{\sigma \in \{1, \cdots, L\}^k} \left\| \mathbf{A}_{\sigma_k} \, \cdots \, \mathbf{A}_{\sigma_2} \, \mathbf{A}_{\sigma_1} \right\|^{1/k}, \tag{5.16}$$

where $\| \cdot \|$ is *an arbitrary* matrix norm.

In words, the joint spectral radius considers the *worst-case* behavior among all possible index sequences. When the set $\mathcal{A}$ consists of a single matrix, i.e., $\mathcal{A} = \{\mathbf{A}\}$, the joint spectral radius becomes $\rho(\mathcal{A}) = \rho(\mathbf{A})$. Namely,

$$\rho(\mathbf{A}) = \lim_{k \to \infty} \left\| \mathbf{A}^k \right\|^{1/k}. \tag{5.17}$$

So, it is a natural extension of the spectral radius of a matrix to a set of matrices.

Although the joint spectral radius characterizes the stability of switching systems in the non-random setting, the computation of the joint spectral radius is NP-hard. Thus, there is no tractable way of determining the stability for a given arbitrary $\mathcal{A}$. We refer to [92] for an excellent review of the joint spectral radius. On the contrary, stability of switching systems in the randomized setting can be verified in tractable ways as described in Lemma 5.3. In fact, we will use these results to ensure the convergence of some randomized algorithms in the next section.

5.3 Randomized Kaczmarz and Gauss-Seidel Algorithms

Given a complex matrix $\mathbf{A} \in \mathbb{C}^{M\times N}$ and a complex vector $\mathbf{u} \in \mathbb{C}^{M}$, we will consider the following linear system:

$$\mathbf{A}\,\mathbf{x} = \mathbf{u}, \tag{5.18}$$

where we assume that $M \geq N$, i.e., the system is overdetermined, and the matrix $\mathbf{A}$ has full column rank. When the system is consistent we will use $\mathbf{x}^{\star}$ to denote the solution of the system. When the system is inconsistent, we will use $\mathbf{x}_{\mathrm{LS}}$ to denote the least-squares solution, that is,

$$\mathbf{x}_{\mathrm{LS}} = \arg\min_{\boldsymbol{\xi}} \|\mathbf{A}\boldsymbol{\xi} - \mathbf{u}\|_2^2 = (\mathbf{A}^{\mathrm{H}}\,\mathbf{A})^{-1}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{u}. \tag{5.19}$$

5.3.1 Randomized Kaczmarz Algorithm

In order to solve the linear system of equations in (5.18), the Kaczmarz algorithm [93] considers the following iterative updates on the solution vector $\mathbf{x}_k$:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{u_{i_k} - \mathbf{a}_{(i_k)}^{\mathrm{H}}\,\mathbf{x}_k}{\|\mathbf{a}_{(i_k)}\|_2^2}\,\mathbf{a}_{(i_k)}, \tag{5.20}$$

where $\mathbf{a}_{(j)}^{\mathrm{H}}$ denotes the j^{th} *row* of the matrix $\mathbf{A}$, and i_k denotes the index selected at the k^{th} iteration of the algorithm. In words, the Kaczmarz algorithm selects a row from the matrix $\mathbf{A}$, and then updates the solution $\mathbf{x}_k$ accordingly.

In the randomized variant of the algorithm, the j^{th} row is selected randomly and independently with probability p_j [81, 165]. In this case, the Kaczmarz algorithm can be represented as a randomly switching system, where the sets $\mathcal{A}$ and $\mathcal{U}$ consist of the following elements for $1 \leq j \leq M$:

$$\mathbf{A}_j = \mathbf{I} - \frac{1}{\|\mathbf{a}_{(j)}\|_2^2}\,\mathbf{a}_{(j)}\,\mathbf{a}_{(j)}^{\mathrm{H}}, \qquad \mathbf{u}_j = \frac{u_j}{\|\mathbf{a}_{(j)}\|_2^2}\,\mathbf{a}_{(j)}. \tag{5.21}$$

Note that $\mathbf{A}_j$'s are orthogonal projections, thus the Kaczmarz algorithm "switches between orthogonal projections."

When the formalism of the switching systems is applied to the Kaczmarz algorithm, we get the following average state-transition matrix and the average input signal:

$$\bar{\mathbf{A}} = \mathbf{I} - \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{W}^{-2}\mathbf{A}, \quad \bar{\mathbf{u}} = \mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{W}^{-2}\,\mathbf{u}, \tag{5.22}$$

where $\mathbf{P}$ and $\mathbf{W}$ are diagonal matrices of size M with j^{th} diagonal entry being p_j and $\|\mathbf{a}_{(j)}\|_2$, respectively. Thus, the fixed-point of the average system can be found as follows:

$$\bar{\mathbf{x}} = (\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{W}^{-2}\,\mathbf{A})^{-1}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{W}^{-2}\,\mathbf{u}. \tag{5.23}$$

So, the fixed-point of the average system depends on the update probabilities in general. Nevertheless, when the probabilities are selected as $p_j = \|\mathbf{a}_{(j)}\|_2^2/\|\mathbf{A}\|_{\mathrm{F}}^2$ (i.e., $\mathbf{P} = \mathbf{W}^2/\|\mathbf{A}\|_{\mathrm{F}}^2$), the fixed-point of the average system becomes $\bar{\mathbf{x}} = \mathbf{x}_{\mathrm{LS}}$.

When the linear system of equations in (5.18) is *inconsistent*, the fixed-point of the average system does not satisfy the individual systems in (5.21). Namely, $\bar{\mathbf{x}} = \mathbf{A}_j\,\bar{\mathbf{x}} + \mathbf{u}_j$ does not hold true for all $1 \le j \le M$. Thus, the random vector $\mathbf{x}_k$ does not converge to $\bar{\mathbf{x}}$ even when $\bar{\mathbf{x}}$ corresponds to $\mathbf{x}_{\mathrm{LS}}$. As a result, Kaczmarz iterations cannot obtain the least-squares solution.

When the linear system of equations in (5.18) is *consistent*, i.e., there exists $\mathbf{x}^\star$ such that $\mathbf{A}\mathbf{x}^\star = \mathbf{u}$, the fixed-point of the average system becomes $\bar{\mathbf{x}} = \mathbf{x}^\star$ irrespective of the update probabilities. Furthermore, $\bar{\mathbf{x}} = \mathbf{A}_j\,\bar{\mathbf{x}} + \mathbf{u}_j$ holds true for all $1 \le j \le M$. Thus, the random vector $\mathbf{x}_k$ converges to the solution of the linear system $\mathbf{x}^\star$ as long as the condition given by Lemma 5.2 is met. In fact, the convergence of the randomized Kaczmarz algorithm can be guaranteed for any set of probabilities. More precisely, we have the following:

Corollary 5.1. *When the linear system in* (5.18) *is consistent, randomized Kaczmarz algorithm converges to the unique solution of* (5.18) *in the mean-squared sense (and almost surely) for any set of nonzero probabilities.*

Proof. We note the following inequalities:

$$\mathbf{\Phi} = \sum_{j=1}^{M} p_j\,\mathbf{A}_j^* \otimes \mathbf{A}_j \preceq \sum_{j=1}^{M} p_j\,\mathbf{I}_N \otimes \mathbf{A}_j = \mathbf{I}_N \otimes \bar{\mathbf{A}} \tag{5.24}$$

$$= \mathbf{I}_{N^2} - \Big(\mathbf{I}_N \otimes \big(\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{W}^{-2}\mathbf{A}\big)\Big) \prec \mathbf{I}_{N^2}, \tag{5.25}$$

where $\mathbf{I}_N$ denotes the identity matrix of size N. The inequality in (5.24) follows from the fact that $\mathbf{0} \preceq \mathbf{A}_j \preceq \mathbf{I}$. The strict inequality in (5.25) follows from the assumption that the matrix $\mathbf{A}$ has full column rank, and the probabilities satisfy $p_j > 0$. The fact that $\mathbf{\Phi} \succeq \mathbf{0}$ and the inequality (5.25) imply that $\rho(\mathbf{\Phi}) < 1$. Then, Lemma 5.2 proves the claimed convergence. □

We note that the mean-squared convergence of the randomized Kaczmarz algorithm was first proved explicitly in [165] for the probabilities $p_j = \|\mathbf{a}_{(j)}\|_2^2 / \|\mathbf{A}\|_{\mathrm{F}}^2$. The almost sure convergence of the algorithm was shown in [36]. The study [46] considered the optimal set of probabilities that minimizes the upper bound in (5.24), and [3] considered the optimal set of probabilities that minimizes $\rho(\mathbf{\Phi})$ itself. Since $\mathbf{\Phi}$ is a positive semi-definite matrix by construction in the case of Kaczmarz algorithm, the optimal selection of the probabilities was based on semi-definite programming in both [46] and [3].

Although the first convergence proof of the randomized Kaczmarz algorithm is attributed to the study in [165], it is also possible to find convergence proofs for the algorithm in control theory literature. In particular, the study [23] considered the updates in (5.21) as an "adaptive filtering" (see [23, Eq. (20)]) and proved the almost sure convergence of the iterations (see [23, Theorem 7]). In addition, the book [42] considered the same update scheme as an application of its results (see [42, Section 3.6.2]) and proved the almost sure convergence for the more general case of indices being selected according to an *ergodic Markov chain*. We note that the original form of the Kaczmarz algorithm [93] considers the use of consecutive indices, which is, in fact, equivalent to a Markov chain on a directed cycle graph. So, [42, Lemma 3.53] proves the convergence of the original Kaczmarz algorithm as well as its randomized variant from the viewpoint of switching systems.

Although the random vector $\mathbf{x}_k$ in Kaczmarz algorithm does not converge to the least squares solution $\mathbf{x}_{\mathrm{LS}}$ in the case of inconsistent system of equations, it is in fact possible to obtain the solution $\mathbf{x}_{\mathrm{LS}}$ using a *sample averaging*. In this regard, we first define the sample average (of the first K iterations) $\mathbf{y}_K$ as follows:

$$\mathbf{y}_K = \frac{1}{K} \sum_{k=1}^{K} \mathbf{x}_k. \tag{5.26}$$

We now note that $\mathbb{E}[\mathbf{x}_k]$ is *the ensemble average* of the random vector $\mathbf{x}_k$, and as the iterations progress the ensemble average converges to $\bar{\mathbf{x}}$. See (5.10). This is due to the stability of the matrix $\bar{\mathbf{A}}$ proven in Corollary 5.1. Since the independent selection of the indices forms an *ergodic Markov chain*, the sample average converges to the ensemble average as more samples are used. More precisely,

$$\lim_{K\to\infty} \|\mathbf{y}_K - \bar{\mathbf{x}}\|_2^2 = 0, \tag{5.27}$$

which follows from the stability of the matrix $\mathbf{\Phi}$. Thus, the random vector $\mathbf{y}_K$ converges to $\bar{\mathbf{x}}$ in (5.23) in the mean-squared sense whether the system of equations are consistent or not. Although this convergence behavior holds true for any set of index selection probabilities, the fixed-point of the average system $\bar{\mathbf{x}}$ becomes the least-squares solution only when the probabilities are selected as $\mathbf{P} = \mathbf{W}^2/\|\mathbf{A}\|_{\mathrm{F}}^2$. With this particular selection of the probabilities, we can claim for the following:

$$p_j = \|\mathbf{a}_{(j)}\|_2^2 \,/\, \|\mathbf{A}\|_{\mathrm{F}}^2 \qquad \Longrightarrow \qquad \lim_{K\to\infty} \|\mathbf{y}_K - \mathbf{x}_{\mathrm{LS}}\|_2^2 = 0, \tag{5.28}$$

which shows that the sample average $\mathbf{y}_K$ converges to the least-squares solution in the mean-squared sense.

5.3.2 Randomized Gauss-Seidel Algorithm

Another approach for solving the system of equations in (5.18) is the Gauss-Seidel algorithm, which updates the solution vector $\mathbf{x}_k$ iteratively according to the following scheme:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{\mathbf{a}_{i_k}^{\mathrm{H}}\left(\mathbf{u} - \mathbf{A}\,\mathbf{x}_k\right)}{\|\mathbf{a}_{i_k}\|_2^2}\,\mathbf{e}_{i_k}, \tag{5.29}$$

where $\mathbf{a}_j$ and $\mathbf{e}_j$ denote the j^{th} *column* of the matrix $\mathbf{A}$ and the identity matrix $\mathbf{I}$, respectively. The index selected at the k^{th} iteration is denoted by i_k. In words, the Gauss-Seidel algorithm selects a column from the matrix $\mathbf{A}$, and then updates only the corresponding entry of the solution vector. In the randomized variant, the j^{th} column is selected randomly and independently with probability p_j [107]. So, the algorithm can be represented as a randomly switching system, where the sets $\mathcal{A}$ and $\mathcal{U}$ consist of the following elements for $1 \leq j \leq N$:

$$\mathbf{A}_j = \mathbf{I} - \frac{1}{\|\mathbf{a}_j\|_2^2}\,\mathbf{e}_j\,\mathbf{a}_j^{\mathrm{H}}\,\mathbf{A}, \qquad \mathbf{u}_j = \mathbf{e}_j\,\frac{\mathbf{a}_j^{\mathrm{H}}\,\mathbf{u}}{\|\mathbf{a}_j\|_2^2}. \tag{5.30}$$

We note that $\mathbf{A}_j$'s defined in (5.30) satisfy $\mathbf{A}_j^2 = \mathbf{A}_j$, but they are not Hermitian, i.e., $\mathbf{A}_j^{\mathrm{H}} \neq \mathbf{A}_j$, in general. Thus, the Gauss-Seidel algorithm "switches between *oblique* projections."

When the formalism of the switching systems is applied to the Gauss-Seidel algorithm, we get the following average state-transition matrix and the average input signal:

$$\bar{\mathbf{A}} = \mathbf{I} - \mathbf{P}\,\mathbf{W}^{-2}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{A}, \qquad \bar{\mathbf{u}} = \mathbf{P}\,\mathbf{W}^{-2}\,\mathbf{A}^{\mathrm{H}}\,\mathbf{u}, \tag{5.31}$$

where $\mathbf{P}$ and $\mathbf{W}$ are diagonal matrices of size N with j^{th} diagonal entry being p_j and $\|\mathbf{a}_j\|_2$, respectively. So, the fixed-point of the average system becomes $\bar{\mathbf{x}} = \mathbf{x}_{\mathrm{LS}}$ irrespective of the update probabilities. More importantly, unlike the Kaczmarz method, the point $\bar{\mathbf{x}}$ satisfies $\bar{\mathbf{x}} = \mathbf{A}_j\,\bar{\mathbf{x}} + \mathbf{u}_j$ for all $1 \leq j \leq N$ whether the set of linear equations (5.18) is consistent or not. Thus, the random vector $\mathbf{x}_k$ indeed converges to the least-squares solution in the mean-squared sense as long as the condition given by Lemma 5.2 is met. In fact, the convergence of the randomized Gauss-Seidel algorithm can be guaranteed for any set of probabilities. More precisely, we have the following:

Corollary 5.2. *The randomized Gauss-Seidel method converges to the least-squares solution of* (5.18) *in the mean-squared sense (and almost surely) for any set of nonzero probabilities.*

Proof. We will show that the third statement in Lemma 5.3 holds, which in turn implies the convergence of the iterations. In this regard take $\mathbf{X} = (\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1}$, and note that $\mathbf{X} > \mathbf{0}$. Then,

$$\mathbf{X} - \sum_{j=1}^{N} p_j\,\mathbf{A}_j\,\mathbf{X}\,\mathbf{A}_j^{\mathrm{H}} = \mathbf{P}\,\mathbf{W}^{-2} > \mathbf{0}, \tag{5.32}$$

where the positive-definiteness follows from the fact that all the probabilities are nonzero. This proves the claim. □

We note that the mean-squared convergence of the randomized Gauss-Seidel algorithm was first proved explicitly in [107] for the probabilities $p_j = \|\mathbf{a}_j\|_2^2/\|\mathbf{A}\|_{\mathrm{F}}^2$. We refer to [132, 80, 110, 122] (and references therein) for more results involving randomized Kaczmarz and Gauss-Seidel algorithms and their extensions.

5.3.3 Randomized Asynchronous Fixed-Point Iterations

When the system in (5.18) is "square," i.e., $M = N$, fixed-point iterations provide an alternative approach for obtaining numerical solutions to linear systems of equations [30, 31, 190]. Asynchronous variants of fixed-point iterations are also studied in detail in non-random [32, 14, 19, 20] and random settings [9]. We have also studied random asynchronous fixed-point iterations are studied in the context of graph signal processing for distributed and asynchronous implementation of graph filters in Chapters 4 and 3.

For a given matrix $\mathbf{A} \in \mathbb{C}^{N \times N}$ and an input signal $\mathbf{u} \in \mathbb{C}^N$, in this section we consider the following random asynchronous fixed-point iterations:

$$(\mathbf{x}_{k+1})_i = \begin{cases} (\mathbf{A}\,\mathbf{x}_k)_i + u_i, & \text{w.p.} \quad p_i, \\ (\mathbf{x}_k)_i, & \text{w.p.} \quad 1 - p_i, \end{cases} \tag{5.33}$$

where the i^{th} index of the vector $\mathbf{x}_k$ is updated with probability p_i independently in every iteration. So, there are 2^N different ways of updating the vector $\mathbf{x}_k$ in every iteration, and the updates in (5.33) can be written as a randomly switching system with the sets $\mathcal{A}$ and $\mathcal{U}$ consisting of 2^N elements.

More precisely, first enumerate all subsets of $\{1, \cdots, N\}$, and let $\mathcal{T}_j$ denote the j^{th} subset. Then, the j^{th} element of the sets $\mathcal{A}$ and $\mathcal{U}$ can be written as follows for $1 \leq j \leq 2^N$:

$$\mathbf{A}_j = \mathbf{I} - \mathbf{D}_{\mathcal{T}_j}\,\big(\mathbf{I} - \mathbf{A}\big), \qquad \mathbf{u}_j = \mathbf{D}_{\mathcal{T}_j}\,\mathbf{u}, \tag{5.34}$$

where $\mathbf{D}_{\mathcal{T}_j}$ is a diagonal matrix that has 1's at the indices belonging to the set $\mathcal{T}_j$ and 0 elsewhere. Furthermore, the probability q_j of switching to $\mathbf{A}_j$ can be written as follows:

$$q_j = \Bigg(\prod_{i \in \mathcal{T}_j} p_i\Bigg)\Bigg(\prod_{i \notin \mathcal{T}_j} (1 - p_i)\Bigg). \tag{5.35}$$

When the formalism of the switching systems is applied to the model (5.33), we get the following average state-transition matrix and the average input signal:

$$\bar{\mathbf{A}} = \mathbf{I} - \mathbf{P}\,(\mathbf{I} - \mathbf{A}), \qquad \bar{\mathbf{u}} = \mathbf{P}\,\mathbf{u}, \tag{5.36}$$

where $\mathbf{P} = \operatorname{diag}([p_1 \quad p_2 \quad \cdots \quad p_N])$ is a diagonal matrix consisting of the update probabilities of the model (5.33). So, the fixed point of the average system is the same as the fixed point of the pair $(\mathbf{A}, \mathbf{u})$, namely, $\bar{\mathbf{x}} = (\mathbf{I} - \mathbf{A})^{-1}\,\mathbf{u}$ irrespective of the update probabilities. Furthermore, the point $\bar{\mathbf{x}}$ satisfies $\bar{\mathbf{x}} = \mathbf{A}_j\,\bar{\mathbf{x}} + \mathbf{u}_j$ for all $1 \leq j \leq 2^N$. Thus, the random vector $\mathbf{x}_k$ converges to $\bar{\mathbf{x}}$ as long as the condition (5.13) is met.

Unlike the randomized Kaczmarz and Gauss-Seidel algorithms, random asynchronous fixed-point iterations may not converge for an arbitrary set of probabilities. Nevertheless, the convergence of the updates can be verified via stability of the matrix $\mathbf{\Phi}$ defined in (5.13). With the matrices in (5.34) and probabilities in (5.35), the matrix $\mathbf{\Phi}$ can be written explicitly as follows in the case of asynchronous fixed-point iterations (See Corollary 4.2 and the proof in Section 4.8.6):

$$\mathbf{\Phi} = \bar{\mathbf{A}}^* \otimes \bar{\mathbf{A}} + \Big((\mathbf{I} - \mathbf{P}) \otimes \mathbf{P}\Big)\,\mathbf{J}\,\Big((\mathbf{A}^* - \mathbf{I}) \otimes (\mathbf{A} - \mathbf{I})\Big), \tag{5.37}$$

where $\mathbf{J}$ is a diagonal matrix of size N^2 with diagonal entries being equal to the vectorized identity matrix of size N. See (4.39).

We know that stability of $\mathbf{A}$, i.e., $\rho(\mathbf{A}) < 1$, is both necessary and sufficient for synchronous fixed-point iterations to converge. However, the most important observation regarding (5.37) is that stability of the matrices $\mathbf{\Phi}$ and $\mathbf{A}$ do *not* imply each other. So, an unstable system (in the synchronous world) may converge with randomized asynchronicity. Furthermore, eigenvectors (and not just eigenvalues) of the matrix $\mathbf{A}$ are also important in determining the convergence in the random asynchronous case (see Section 4.4.4). We also note that the condition $\mathbf{A}^{\mathrm{H}}\,\mathbf{P}\,\mathbf{A} \prec \mathbf{P}$ is shown to be *sufficient* for the convergence of the updates in Corollary 3.1 and Corollary 4.3, and it is more relaxed than the necessary condition in the non-random setting [32, 14] (see Lemma 3.1).

The randomized model (5.33) can also be extended to have *time-varying* input signals, where the vector $\mathbf{u}$ changes as the iterations progress. In this case, the updates become a randomized and asynchronous variant of the discrete-time state-space model, and it is possible to study the "frequency response" of such systems in a statistical sense. This aspect is studied in Chapter 4.

5.4 Concluding Remarks

This chapter showed that randomized versions of Kaczmarz's method, Gauss-Seidel iterations, and asynchronous fixed-point iterations can be represented as specific instances of randomly switching systems. So, convergence properties of these randomized algorithms follow directly from the stability properties of randomly switching systems, which are well studied in the control theory literature. Thus, this chapter shows that randomized iterative algorithms can be studied with the tools already available in control theory.

5.5 Appendices

5.5.1 Proof of Lemma 5.2

When the error vector $\mathbf{r}_k$ defined in (5.11) is substituted into the switching model (5.2), we get the following:

$$\mathbf{r}_{k+1} + \bar{\mathbf{x}} = \mathbf{A}_{i_k}\,(\mathbf{r}_k + \bar{\mathbf{x}}) + \mathbf{u}_{i_k} + \mathbf{w}_k, \tag{5.38}$$

which can be written equivalently as

$$\mathbf{r}_{k+1} = \mathbf{A}_{i_k}\,\mathbf{r}_k - \mathbf{z}_{i_k} + \mathbf{w}_k, \tag{5.39}$$

where $\mathbf{z}_j$ is defined as follows for $1 \leq j \leq L$:

$$\mathbf{z}_j = \bar{\mathbf{x}} - \left(\mathbf{A}_j\, \bar{\mathbf{x}} + \mathbf{u}_j\right). \tag{5.40}$$

We now consider the following conditional expectation:

$$\mathbb{E}[\mathbf{r}_{k+1}\mathbf{r}_{k+1}^{\mathrm{H}} \,|\, \mathbf{r}_k] = \sum_{j=1}^{L} p_j\left(\mathbf{A}_j\, \mathbf{r}_k\, \mathbf{r}_k^{\mathrm{H}}\, \mathbf{A}_j^{\mathrm{H}} + \mathbf{z}_j\, \mathbf{z}_j^{\mathrm{H}}\right) + \mathbf{\Gamma} - \sum_{j=1}^{L} p_j\left(\mathbf{A}_j\, \mathbf{r}_k\, \mathbf{z}_j^{\mathrm{H}} + \mathbf{z}_j\, \mathbf{r}_k^{\mathrm{H}}\, \mathbf{A}_j^{\mathrm{H}}\right), \tag{5.41}$$

where we use the fact that systems are selected independently with probability p_j in each iteration, and the noise $\mathbf{w}_k$ has zero-mean and it is uncorrelated with the random selections.

Then, taking one more expectation with respect to $\mathbf{r}_k$, we get the following:

$$\mathbf{R}_{k+1} = \sum_{j=1}^{L} p_j\left(\mathbf{A}_j\, \mathbf{R}_k\, \mathbf{A}_j^{\mathrm{H}}\right) + \mathbf{\Gamma} + \mathbf{Z} \tag{5.42}$$

$$- \sum_{j=1}^{L} p_j\left(\mathbf{A}_j\, \bar{\mathbf{A}}^k\, \mathbb{E}[\mathbf{r}_0]\, \mathbf{z}_j^{\mathrm{H}} + \mathbf{z}_j\, \mathbb{E}[\mathbf{r}_0]^{\mathrm{H}}\, \bar{\mathbf{A}}^{k\mathrm{H}}\, \mathbf{A}_j^{\mathrm{H}}\right), \tag{5.43}$$

where $\mathbf{Z}$ is defined as in (5.15).

We now vectorize both sides of (5.42) and obtain the following:

$$\mathrm{vec}(\mathbf{R}_{k+1}) = \mathbf{\Phi}\ \mathrm{vec}(\mathbf{R}_k) + \mathrm{vec}(\mathbf{Z} + \mathbf{\Gamma}) + \mathbf{c}_k, \tag{5.44}$$

where $\mathbf{c}_k$ is the vectorized version of the quantity in (5.43), and the matrix $\mathbf{\Phi}$ is as in (5.13).

It is clear from the linear recursion in (5.44) that when $\rho(\mathbf{\Phi}) \geq 1$, the vector $\mathrm{vec}(\mathbf{R}_k)$ diverges as k goes to infinity.

When $\rho(\mathbf{\Phi}) < 1$, we first note that the vector $\mathbf{c}_k$ converges to zero. This is due to [42, Proposition 3.6], which states that convergence of the second moment implies the convergence of the first moment. Namely, $\rho(\mathbf{\Phi}) < 1$ implies $\rho(\bar{\mathbf{A}}) < 1$. Furthermore, $\rho(\mathbf{\Phi}) < 1$ implies that the vector $\mathrm{vec}(\mathbf{R}_k)$ converges as well. By taking limits of both sides of (5.44), we get the following:

$$\mathrm{vec}(\mathbf{R}) = \mathbf{\Phi}\ \mathrm{vec}(\mathbf{R}) + \mathrm{vec}(\mathbf{Z} + \mathbf{\Gamma}), \tag{5.45}$$

where $\mathbf{R}$ denotes the limit of $\mathbf{R}_k$ as k goes to infinity.

The solution to (5.45) is given as follows:

$$\mathrm{vec}(\mathbf{R}) = (\mathbf{I} - \mathbf{\Phi})^{-1}\ \mathrm{vec}(\mathbf{Z} + \mathbf{\Gamma}), \tag{5.46}$$

which completes the proof.

Chapter 6

EXTENDING CLASSICAL MULTIRATE SIGNAL PROCESSING THEORY TO GRAPHS

6.1 Introduction

Multirate analysis for graph signals has been of interest since the introduction of the field of graph signal processing. The first set of papers, pioneered by Narang and Ortega [129, 127, 125, 4, 66, 160] showed how two-channel filter banks can be constructed on graphs, and went on to develop elegant techniques for the design of down-sampled, two-channel perfect reconstruction filter banks on bipartitie graphs. These results were developed for graphs that have a real, symmetric adjacency matrix, and all results were based on the graph Laplacian. Studies in [57, 56, 58] mainly focus on circulant graphs and analyze two-channel decomposition of graph signals. Multirate decomposition can be achieved by iterative application of 2-channel systems. The study in [192] proposes to combine decimators and filters for construction of a filter bank on a graph. Motivated by Haar filter banks in the classical theory, a graph filter bank is developed using the partitions of the graph.

Inspired by the pioneering contributions of [152] and [129], this chapter extends many of the basic concepts of classical multirate signal processing and filter bank theory to graphs. We first develop the equivalent of fundamental ideas such as noble identities, aliasing, and polyphase decompositions in graph multirate systems. Then, a detailed general theory for M-channel filter banks is developed. The graphs are assumed to be very general as in [152], with a possibly non-symmetric and complex adjacency matrix.

In the context of graph signal processing a linear filter is just a square matrix. By a cascade of such matrices one can trivially construct a graph filter bank. Problems with this approach and reasons why we focus on polynomial filters are detailed in in Section 6.3. We will see in this chapter that the extension of classical multirate signal processing theory to graphs is nontrivial, and requires certain mathematical restrictions on the graph adjacency matrix $\mathbf{A}$. While some of the results of classical filter bank theory extend easily, some of the deeper results unfold a lot of surprises – some extend and some do not extend to the case of graphs. For example, the classical noble identities [200] cannot be taken for granted, and require some restrictions on the graph matrix **A**. Similarly, one cannot take it for granted that the delay chain system [200] is a perfect reconstruction filter bank (an easily proved result in the case of classical filter banks). It will also be shown that M-partite extensions of the bipartite results in [129] will not in general work for M channel filter banks, but a more restrictive condition called M-block cyclic property should be imposed on the graph. While a number of results in this chapter require this property, there are ways to relax it as explained in Section 6.15, and also in specific theorem statements.

While dealing with graphs, we often make comparisons with conventional multirate systems and filter banks defined in the time domain [200, 199, 204, 164, 43, 79]. On rare occasions we also make comparisons with systems defined in the cyclic (periodic) time domain that is equivalent to a graph with a specific cyclic adjacency matrix (Eq. (12) in [153]). These systems defined in the time-domain (or cyclical time domain on occasions) will be referred to as "classical" systems, "classical" filter banks, and so forth.

6.1.1 Scope and Outline of This Chapter

After introducing the canonical downsampling and upsampling operators on graphs, we begin with a study of noble identities. These identities are known to be important in theoretical developments and practical implementations of classical multirate systems [200, 43]. For the case of graphs we will show in Section 6.2.2 that the noble identities make sense only for graphs with a certain specific structure on the adjacency matrix (Theorems 6.1 and 6.2). We then show in Section 6.2.3 that the delay chain filter bank (or the lazy filter bank) does not in general have perfect reconstruction property for arbitrary graphs. We introduce Type-1 and Type-2 polyphase representation of polynomial filters in Section 6.2.4. Section 6.3 discusses how one can trivially construct a graph filter bank, and motivates the use of polynomial filters. In order to extend the results for bipartite graphs on 2-channel systems to M-channels, one may propose to use M-partite graphs rather than bipartite graphs. In Section 6.4 we briefly discuss that such a generalization is not useful. Section 6.5 introduces M-block cyclic graphs that are important for many of the later developments in this chapter. The eigenstructure of M-block cyclic graphs, which forms the foundation for many of these results, is developed in Section 6.6 (Theorem 6.9). Many of the results developed in this thesis are therefore valid only for graphs that satisfy either the M-block cyclic property or the eigenvector structure of M-block cyclic graphs. In Section 6.15 we also discuss how this restriction can be removed, and what the price paid is.

The concepts of spectrum folding and aliasing are developed in Section 6.8 for graphs that have an eigenvector structure similar to those of M-block cyclic graphs. These will be used later to develop perfect reconstruction filter banks.

Section 6.9 embarks on a study of three related properties of linear systems on graphs: namely the polynomial property, the shift invariance property, and the so-called alias-free property. While these properties are identical in classical signal processing theory, such is not the case on graphs. Some of these interrelations were developed in [152], but Section 6.9 goes deeper and establishes the complete picture. This will be useful for obtaining a deeper understanding of alias-free maximally decimated M-channel graph filter banks to be studied in this chapter.

With the graph adjacency matrix regarded as a shift operator [152, 153], it will be shown in Section 6.10.1 that the graph filter bank is a shift-variant system, although it is in general not *periodically* shift-variant as in classical time domain filter banks. We then establish the conditions on the adjacency matrix $\mathbf{A}$ for the periodically shift-varying property and show that it is exactly identical to the conditions for the existence of graph noble identities (Theorem 6.17).

Then in Section 6.11 we consider graphs that satisfy the specific eigenvector structure (i.e., $\mathbf{\Omega}$-graphs). These graphs are more general than M-block cyclic graphs, which satisfy both the eigenvalue and eigenvector conditions. For such graphs we define band-limited graph signals and polynomial perfect interpolation filters for decimated versions of such signals. This allows us to develop a class of perfect reconstruction filter banks for $\mathbf{\Omega}$-graphs (Theorem 6.19), similar to ideal brickwall filter banks of classical sub-band coding theory. Such graph filter bank designs are usually not practical because the polynomial filters have order N-1 (where the graph has N vertices and N can be very large). Furthermore these specific filters for perfect reconstruction are very sensitive to our knowledge of the graph eigenvalues.

In Section 6.12 we develop graph filter banks on M-block cyclic graphs (defined and studied in Section 6.5). For such graphs the eigenvalues and eigenvectors are *both* constrained as in (6.56), (6.57). We show that for such graphs the condition for perfect reconstruction is very similar to the PR condition in classical filter banks (Theorem 6.21). For such graphs, it is therefore possible to design PR filter banks by starting from any classical PR system. In particular it is possible to obtain PR systems with arbitrarily small orders (independent of the size of the graph) for the polynomial filters. Furthermore the PR solutions $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ are not sensitive to graph eigenvalues.

In Section 6.13 we consider polyphase representations for graph filter banks. This is useful to obtain alternative theoretical representations, as well as implementations. Unlike classical filter banks, these polyphase representations are not always valid. They are valid only for those graphs that satisfy the noble identity requirements (Theorem 6.3). For such graphs, the PR condition and the alias cancellation condition can further be expressed in terms of the analysis and synthesis polyphase matrices if the graphs also satisfy the M-block cyclic property (Theorems 6.25 and 6.26). Interestingly these alias cancellation conditions are somewhat similar to the pseudo-circulant property developed for classical filter banks in [200].

In Section 6.14 we consider frequency responses of graph filter banks inspired by similar ideas in [153, 160]. We will see that this concept can be meaningfully developed for filter banks on M-cyclic graphs with all eigenvalues on the unit circle, but not for arbitrary graphs.

Finally in Section 6.15 we show that the eigenvector structure in (6.57) ($\mathbf{\Omega}$-structure) can be relaxed simply by considering a transformed graph based on similarity transformations. This generalization therefore extends many of the results in this chapter to more general graphs. In short, all results that we developed for $\mathbf{\Omega}$-graphs (e.g., Theorems 6.18 and 6.19) generalize to arbitrary graphs. Similarly all results which we developed for M-block cyclic graphs (e.g., Theorems 6.20 and 6.21) generalize to graphs that are subject only to the eigenvalue constraint (6.56) and not the eigenvector constraint (6.57).

Section 6.17 concludes the chapter. Sections 6.18.1 - 6.18.4 provide supplementary theorems and detailed proofs of some theorems presented in this chapter

The content of this chapter is mainly drawn from [172, 173], and parts of it have been presented in [174, 175, 171, 177].

6.1.2 Notation

Given a graph, $\mathbf{A}$ represents the adjacency matrix of the graph. We often refer to a graph with adjacency matrix $\mathbf{A}$ as "graph $\mathbf{A}$" for convenience. Throughout the chapter, N denotes the size of the graph and length of the signal and M denotes the decimation ratio or the number of filters in a graph filter bank, according to context. Hence, $\mathbf{A} \in \mathbb{C}^{N\times N}$. In this chapter, the $(i, j)^{th}$ block of the adjacency matrix $\mathbf{A}$ is denoted by $(\mathbf{A})_{i,j}$ and $(\mathbf{v})_i$ denotes the $(i)^{th}$ block of the vector $\mathbf{v}$. Throughout the chapter, when it is not indicated, it should be clear that $(\mathbf{A})_{i,j} \in \mathbb{C}^{(N/M)\times(N/M)}$ and $(\mathbf{v})_i \in \mathbb{C}^{N/M}$. Otherwise, they are clearly indicated to have the specified sizes. The cyclic permutation matrix of size N is denoted by $\mathbf{C}_N$, and it is defined as:

$$\mathbf{C}_N = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \in \mathbb{C}^{N\times N}. \tag{6.1}$$

6.1.3 Review of DSP on Graphs

In DSP on graphs, the adjacency matrix of the graph of interest is considered to be *the unit shift* operator for a signal on the graph [152]. Namely, let $\mathbf{x}$ be a signal on a graph with the adjacency matrix $\mathbf{A}$. Then the signal $\mathbf{y}$ computed as

$$\mathbf{y} = \mathbf{A}\,\mathbf{x}, \tag{6.2}$$

is called as the unit shifted version of $\mathbf{x}$. We also would like to indicate that the adjacency matrix is not the only choice for the shift operator in general. The study in [68] proposes alternative definitions for the shift operator. Nevertheless, for simplicity, we will stick with the adjacency matrix as done in [152, 153].

In general, any square matrix of size N, $\mathbf{H} \in \mathbb{C}^{N\times N}$, is considered as a *linear graph filter* on the graph. When we have $\mathbf{y} = \mathbf{H}\,\mathbf{x}$, we call $\mathbf{y}$ as the filtered version of the signal $\mathbf{x}$. In this study, we are interested in a special type of linear filters, namely polynomial filters, which are defined as follows.

Definition 6.1 (Polynomial filters [152, 126])**.** *A linear system* $\mathbf{H}$ *on a graph* $\mathbf{A}$ *is said to be a* polynomial *filter if*

$$\mathbf{H} = H(\mathbf{A}) = \sum_{k=0}^{L} h_k\,\mathbf{A}^k, \tag{6.3}$$

for a set of $h_k \in \mathbb{C}$*. Here* L *is called the* order *of the filter.*

We can assume without loss of generality that $L < N$. This is because, according to Cayley-Hamiltion theorem, powers A^k for $k \geq N$ can be expressed as linear combinations of smaller powers [85].[1]

For a graph with the adjacency matrix $\mathbf{A}$, let the following denote the Jordan decomposition [85, 152] of the adjacency matrix

$$\mathbf{A} = \mathbf{V}\,\mathbf{J}\,\mathbf{V}^{-1}, \tag{6.4}$$

[1]Also see Theorem 3 of [152].

where $\mathbf{V}$ is composed of the (generalized) eigenvectors of the adjacency matrix and $\mathbf{J}$ is the Jordan normal form of $\mathbf{A}$. When $\mathbf{A}$ is diagonalizable, (6.4) reduces to the following form

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{-1}, \tag{6.5}$$

for some diagonal $\mathbf{\Lambda}$ consisting of the eigenvalues and some invertible square matrix $\mathbf{V}$ consisting of the eigenvectors of the adjacency matrix. When $\mathbf{A}$ has distinct eigenvalues, it is necessarily diagonalizable, but not vice versa.

Using the Jordan decomposition in (6.4), we then have the following definitions.

Definition 6.2 (Graph Fourier transform [152, 153])**.** *Let* $\mathbf{x}$ *be a signal on a graph with the adjacency matrix* $\mathbf{A}$. *Then the graph Fourier transform of* $\mathbf{x}$ *on the graph* $\mathbf{A}$ *is given by*

$$\widehat{\mathbf{x}} = \mathbf{V}^{-1}\,\mathbf{x}, \tag{6.6}$$

where $\mathbf{V}$ *has the (generalized) eigenvectors of* $\mathbf{A}$ *as in* (6.4).

Definition 6.3 (Frequency domain operation)**.** *Let* $\mathbf{H}$ *be a linear filter on a graph with the adjacency matrix* $\mathbf{A}$. *Then the* frequency domain operator *corresponding to* $\mathbf{H}$ *is defined by*

$$\widehat{\mathbf{H}} = \mathbf{V}^{-1}\,\mathbf{H}\,\mathbf{V}, \tag{6.7}$$

where $\mathbf{V}$ *has the (generalized) eigenvectors of* $\mathbf{A}$ *as in* (6.4).

Definiton 6.3 does not imply that $\mathbf{V}$ diagonalizes the filter $\mathbf{H}$, that is, $\widehat{\mathbf{H}}$ is not diagonal in general.

Notice that Definitions 6.2 and 6.3 are consistent with each other, that is, for a graph signal $\mathbf{x}$ and a linear filter $\mathbf{H}$, we have $\mathbf{y} = \mathbf{H}\mathbf{x}$ if and only if $\widehat{\mathbf{y}} = \widehat{\mathbf{H}}\,\widehat{\mathbf{x}}$. As explained in Section 6.8 (and Definition 6.7) later, $\widehat{\mathbf{H}}$ will be referred to as the *frequency response* of $\mathbf{H}$ only when $\widehat{\mathbf{H}}$ is a diagonal matrix.

6.2 Building Blocks for Multirate Processing on Graphs

6.2.1 Downsampling and Upsampling Operations

One of the most essential building blocks for multirate signal processing is the *decimation operation* [200]. In the graph signal processing, we will assume that this operator retains N/M samples of the original graph signal $\mathbf{x}$ that has N samples. It will be assumed that M is a divisor of N. Since the numbering of the graph vertices is flexible [152, 45], we will assume, without loss of generality, that the first N/M

samples of **x** are retained by the decimator. Thus the *graph decimation* operator is defined as:

Definition 6.4 (Canonical Decimator)**.** *The M-fold graph decimation operator is defined by the matrix*

$$\mathbf{D} = \begin{bmatrix} \mathbf{I}_{N/M} & \mathbf{0}_{N/M} & \cdots & \mathbf{0}_{N/M} \end{bmatrix} \in \mathbb{C}^{(N/M)\times N}. \tag{6.8}$$

Given a graph signal **x**, decimated graph signal *is then denoted as* $\mathbf{D}\,\mathbf{x}$.

We refer to **D** as *canonical decimator* with decimation ratio M. This is a mapping from N dimensional complex space to N/M dimensional complex space. Similar definitions for the decimator operator have been introduced in the literature [129, 136, 35, 127].

Next, the *upsampling operation* $\mathbf{U} \in \mathbb{C}^{N\times(N/M)}$ is a mapping from N/M dimensional complex space to N dimensional complex space. Once we define the downsampling, we cannot arbitrarly select the upsampling operator, they should be consistent with each other. In general, downsample-then-upsample is a lossy operation. Contrary to that, upsample-then-downsample operator is expected to be equal to identity. That is to say

$$\mathbf{D}\,\mathbf{U} = \mathbf{I}_{N/M}. \tag{6.9}$$

For a given **D**, the right inverse **U** is not unique. When we look for the minimum norm solution, we get

$$\mathbf{U} = \mathbf{D}^{\mathrm{H}} \left(\mathbf{D}\,\mathbf{D}^{\mathrm{H}}\right)^{-1}, \tag{6.10}$$

assuming that **D** has full row rank. This result reduces to

$$\mathbf{U} = \mathbf{D}^{\mathrm{T}} = \begin{bmatrix} \mathbf{I}_{N/M} \\ \mathbf{0}_{N/M} \\ \vdots \\ \mathbf{0}_{N/M} \end{bmatrix} \in \mathbb{C}^{N\times(N/M)}. \tag{6.11}$$

for the decimator operator defined in (6.8). Hence, the corresponding uniform upsampler with expansion ratio M is defined by the matrix $\mathbf{D}^{\mathrm{T}}$. This operation merely inserts blocks of zeros, analogous to conventional expanders [200, 199].

With this selection of the upsampler, we have the following equalities for upsample-then-downsample and downsample-then-upsample operations:

$$\mathbf{D}\,\mathbf{D}^{\mathrm{T}} = \mathbf{I}_{N/M}, \qquad \mathbf{D}^{\mathrm{T}}\,\mathbf{D} = \begin{bmatrix} \mathbf{I}_{N/M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{N\times N}, \tag{6.12}$$

respectively, where zero blocks have appropriate sizes.

In the following, our result will be based on the simple canonical **D** defined in (6.8). More generally, decimator can be selected as an arbitrary $(N/M) \times N$ matrix with full row-rank. Such a definition provides an extension to the results presented in the following sections and allows us to remove some of the restrictions on the adjacency matrix. These details are elaborated in Section 6.15.

6.2.2 The Noble Identities

In classical signal processing, we have the first noble identity described in Figure 6.1a, where $H(z)$ denotes the transfer function of an LTI filter [200]. For graph signals, it is possible to obtain a similar result under some conditions on the graph. The result is given in Figure 6.1b and requires some explanation.

In the classical case, the unit delay z^{-1} has the same meaning for both the original and the decimated signals. But for graph signals, the elementary shift operator should match size of the signal. Since the decimated signal has length N/M, we need to define a different shift operator for the decimated signal. The matrix $\bar{\mathbf{A}}$ in the figure denotes this *adjusted shift operator*.

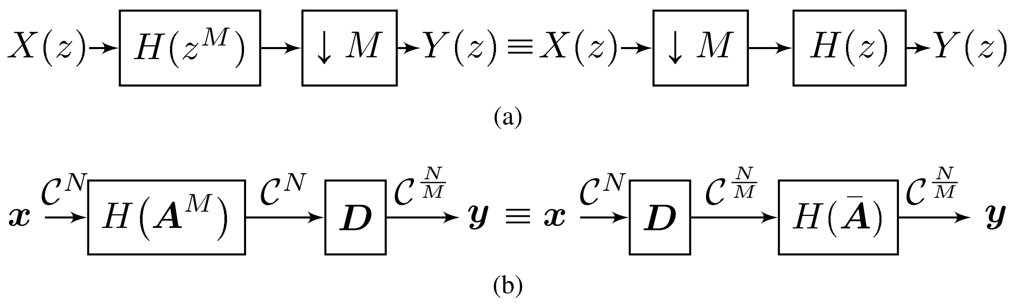


Figure 6.1: The first noble identity (a) for classical multirate signal processing where $\downarrow M$ denotes decimator operation, (b) for graph signals on the adjacency matrix **A**.

With the adjusted shift operator for the decimated signal, we have the following form of the first noble identity for graph signals:

$$\mathbf{D}\ H(\mathbf{A}^M) = H(\bar{\mathbf{A}})\ \mathbf{D}. \tag{6.13}$$

This is shown schematically in Figure 6.1b. It is important to notice that the required adjusted shift operator $\bar{\mathbf{A}}$ that satisfies the noble identity in (6.13) may not exist in general. In the following we will provide the sufficient and necessary condition on **A** so that an adjusted shift operator exists and satisfies (6.13).

Theorem 6.1 (The first noble identity)**.** *Let the decimator* $\mathbf{D}$ *be as in* (6.8)*. If the noble identity* (6.13) *is satisfied by a graph* $\mathbf{A}$ *for* all *polynomial filters* $H(\cdot)$ *for some* $\bar{\mathbf{A}}$, *then* $\mathbf{A}^M$ *has to have the form*

$$\mathbf{A}^M = \begin{bmatrix} (\mathbf{A}^M)_{1,1} & \mathbf{0} \\ (\mathbf{A}^M)_{2,1} & (\mathbf{A}^M)_{2,2} \end{bmatrix}, \tag{6.14}$$

where $(\mathbf{A}^M)_{1,1} \in \mathbb{C}^{(N/M)\times(N/M)}$, *and furthermore*

$$\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}^M\,\mathbf{D}^{\mathrm{T}}, \tag{6.15}$$

i.e., $\bar{\mathbf{A}} = (\mathbf{A}^M)_{1,1}$*. Conversely if* $\mathbf{A}^M$ *and* $\bar{\mathbf{A}}$ *have the above form, then noble identity* (6.13) *holds for all polynomial filters. In short,* (6.13) *holds for all polynomial filters if and only if both* (6.14) *and* (6.15) *are true.*

Proof. First assume (6.13) holds for all polynomials $H(\cdot)$, i.e., $\mathbf{D}\sum_k h_k\,\mathbf{A}^{Mk} = \sum_k h_k\,\bar{\mathbf{A}}^k\,\mathbf{D}$ for all $\{h_k\}$. Then

$$\mathbf{D}\,\mathbf{A}^{Mk} = \bar{\mathbf{A}}^k\,\mathbf{D} \tag{6.16}$$

for all $k \geq 0$. Now express $\mathbf{A}^M$ in partitioned form

$$\mathbf{A}^M = \begin{bmatrix} (\mathbf{A}^M)_{1,1} & (\mathbf{A}^M)_{1,2} \\ (\mathbf{A}^M)_{2,1} & (\mathbf{A}^M)_{2,2} \end{bmatrix}, \tag{6.17}$$

where $(\mathbf{A}^M)_{1,1} \in \mathbb{C}^{(N/M)\times(N/M)}$. For $k = 1$, (6.16) yields $\mathbf{D}\,\mathbf{A}^M = \bar{\mathbf{A}}\,\mathbf{D}$. Using (6.8) this becomes

$$\begin{bmatrix} (\mathbf{A}^M)_{1,1} & (\mathbf{A}^M)_{1,2} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{A}} & \mathbf{0}_{M,(N\text{-}N/M)} \end{bmatrix}, \tag{6.18}$$

which proves $\bar{\mathbf{A}} = (\mathbf{A}^M)_{1,1}$ and $(\mathbf{A}^M)_{1,2} = \mathbf{0}$ indeed. Thus (6.13) implies (6.14) and (6.15).

Conversely assume the form (6.14) and the relation (6.15) are true. First observe that when (6.14) holds, we have $(\mathbf{A}^{Mk})_{1,1} = ((\mathbf{A}^M)_{1,1})^k$. Since (6.15) also holds, it follows that

$$(\mathbf{A}^{Mk})_{1,1} = \bar{\mathbf{A}}^k \tag{6.19}$$

for all $k \geq 0$. This is equivalent to (6.16), as seen by substituting from (6.14) and (6.8). Thus (6.14) and the relation (6.15) imply the noble identity (6.13) indeed. □

The second noble identity in classical signal processing [200] is described schematically in Figure 6.2a. For graph signals, the analogous identity would be as in

Figure 6.2b, where input and output are called as lower and higher rate signal, respectively. Let $\widetilde{\mathbf{A}}$ denote the adjusted shift operator for the lower rate signal in the second noble identity. We have the following form of the second noble identity for graph signals.

$$H(\mathbf{A}^M)\, \mathbf{D}^{\mathrm{T}} = \mathbf{D}^{\mathrm{T}}\, H(\widetilde{\mathbf{A}}). \tag{6.20}$$

$$\begin{tikzcd} X(z) \arrow[r] & \boxed{\uparrow M} \arrow[r] & \boxed{H(z^M)} \arrow[r] & Y(z) \equiv X(z) \arrow[r] & \boxed{H(z)} \arrow[r] & \boxed{\uparrow M} \arrow[r] & Y(z) \end{tikzcd}$$

(a)

$$\begin{tikzcd} \boldsymbol{x} \arrow[r, "\mathcal{C}^{\frac{N}{M}}"] & \boxed{\boldsymbol{D}^T} \arrow[r, "\mathcal{C}^N"] & \boxed{H(\boldsymbol{A}^M)} \arrow[r, "\mathcal{C}^N"] & \boldsymbol{y} \equiv \boldsymbol{x} \arrow[r, "\mathcal{C}^{\frac{N}{M}}"] & \boxed{H(\widetilde{\boldsymbol{A}})} \arrow[r, "\mathcal{C}^{\frac{N}{M}}"] & \boxed{\boldsymbol{D}^T} \arrow[r, "\mathcal{C}^N"] & \boldsymbol{y} \end{tikzcd}$$

(b)

Figure 6.2: The second noble identity for (a) classical multirate signal processing where $\uparrow M$ denotes expander operation, (b) graph signals on the adjacency matrix $\mathbf{A}$.

Theorem 6.2 (The second noble identity)**.** *If the noble identity* (6.20) *is satisfied by a graph* $\mathbf{A}$ *for* all *polynomial filters* $H(\cdot)$ *for some* $\widetilde{\mathbf{A}}$*, then* $\mathbf{A}^M$ *has to have the form*

$$\mathbf{A}^M = \begin{bmatrix} (\mathbf{A}^M)_{1,1} & (\mathbf{A}^M)_{1,2} \\ \mathbf{0} & (\mathbf{A}^M)_{2,2} \end{bmatrix}, \tag{6.21}$$

where $(\mathbf{A}^M)_{1,1} \in \mathbb{C}^{(N/M)\times(N/M)}$*, and furthermore*

$$\widetilde{\mathbf{A}} = \mathbf{D}\, \mathbf{A}^M\, \mathbf{D}^{\mathrm{T}}, \tag{6.22}$$

i.e., $\widetilde{\mathbf{A}} = (\mathbf{A}^M)_{1,1}$*. Conversely if* $\mathbf{A}^M$ *and* $\widetilde{\mathbf{A}}$ *have the above form, then noble identity* (6.20) *holds for all polynomial filters. In short,* (6.20) *holds for all polynomial filters if and only if both* (6.21) *and* (6.22) *are true.*

Proof. First assume there exists an $\widetilde{\mathbf{A}}$ such that (6.20) is true for *all* polynomial filters $H(\cdot)$. This implies in particular

$$\mathbf{A}^{Mk}\, \mathbf{D}^{\mathrm{T}} = \mathbf{D}^{\mathrm{T}}\, \widetilde{\mathbf{A}}^k \tag{6.23}$$

for all $k \geq 0$. Now consider the partitioned form in (6.17). Setting $k = 1$ in (6.23) and using the form of $\mathbf{D}^{\mathrm{T}}$ in (6.11), we get

$$\begin{bmatrix} \widetilde{\mathbf{A}} \\ \mathbf{0}_{(N\text{-}N/M),M} \end{bmatrix} = \begin{bmatrix} (\mathbf{A}^M)_{1,1} \\ (\mathbf{A}^M)_{2,1} \end{bmatrix}, \tag{6.24}$$

which shows that if (6.20) has to be true for all polynomial filters, then $\widetilde{\mathbf{A}} = (\mathbf{A}^M)_{1,1}$ and $(\mathbf{A}^M)_{2,1} = \mathbf{0}$.

Conversely, suppose the form (6.21)) and the relation (6.22) are true. Then $(\mathbf{A}^{Mk})_{1,1} = ((\mathbf{A}^M)_{1,1})^k = \widetilde{\mathbf{A}}^k$. But this is equivalent to (6.23) as seen by substituting from (6.21) and (6.8). So (6.20) holds for all polynomials $H(\cdot)$ indeed. □

Combining the preceding two theorems we get

Theorem 6.3 (The noble identities)**.** *For a graph* $\mathbf{A}$*, the two noble identities*

$$\mathbf{D}\, H(\mathbf{A}^M) = H(\bar{\mathbf{A}})\, \mathbf{D}, \tag{6.25}$$

$$H(\mathbf{A}^M)\, \mathbf{D}^{\mathrm{T}} = \mathbf{D}^{\mathrm{T}}\, H(\bar{\mathbf{A}}), \tag{6.26}$$

are simultaneously satisfied for all polynomial filters $H(\cdot)$ *if and only if the following two equations are satisfied:* $\mathbf{A}^M$ *has the form*

$$\mathbf{A}^M = \begin{bmatrix} (\mathbf{A}^M)_{1,1} & \mathbf{0} \\ \mathbf{0} & (\mathbf{A}^M)_{2,2} \end{bmatrix} \tag{6.27}$$

and

$$\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}^M\,\mathbf{D}^{\mathrm{T}} \in \mathbb{C}^{(N/M)\times(N/M)}, \tag{6.28}$$

where $(\mathbf{A}^M)_{1,1} \in \mathbb{C}^{(N/M)\times(N/M)}$.

It is clear that an arbitrary graph may not satisfy the condition in (6.27). Specific examples of graphs that meet, or do not meet, the condition of Theorem 6.3 will be presented in Sections 6.4 and 6.5.

6.2.3 Lazy Graph Filter Banks

An important theoretical example of a maximally decimated filter bank in classical signal processing is the M-channel delay-chain filter bank, also known as the lazy filter bank, shown in Figure 6.3a. This is a perfect reconstruction system [200], and serves as a starting point for developing more useful filter bank systems. Such a development is typically based on the use of polyphase representations and noble identities [200]. We have already developed noble identities for graph signals above. In the following subsection we will develop polyphase representations for graph filters and in Section 6.13 we shall develop graph filter banks. In the present subsection we consider the graph equivalent of the lazy filter bank shown in Figure 6.3b. In this system the graph signal $\mathbf{x} \in \mathbb{C}^N$ is passed through a chain of graph shift operators

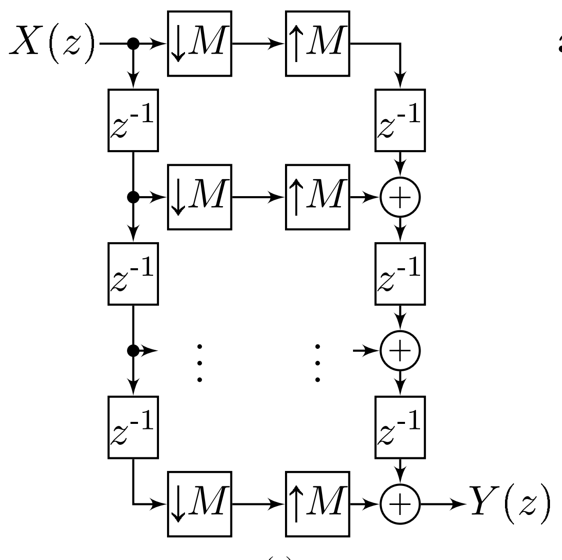
$\mathbf{A}^k$, $0 \leq k \leq M$-1 and each shifted version is passed through the downsampler $\mathbf{D}$ and upsampler $\mathbf{D}^{\mathrm{T}}$. The resulting M signals are then graph-shifted again and added. It is clear that the system is linear with the input-output relation $\mathbf{y} = T(\mathbf{A})\,\mathbf{x}$ where

$$T(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^k. \tag{6.29}$$

For the classical lazy filter bank we have $Y(z) = z^{\text{-}(M\text{-}1)}X(z)$, and it is a perfect reconstruction system. Similarly, we say that the lazy graph filter bank has perfect reconstruction (PR) if $T(\mathbf{A}) = \mathbf{A}^{M\text{-}1}$, that is,

$$\sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^k = \mathbf{A}^{M\text{-}1}. \tag{6.30}$$

This will be referred to as the *lazy FB PR condition*. We will return to more general filter banks on graphs, along with the theory of perfect reconstruction and alias cancellation in Sections 6.10 - 6.13.



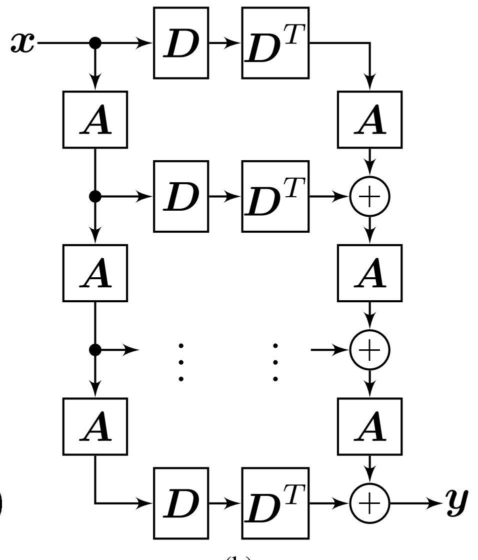


Figure 6.3: (a) M-channel lazy filter bank in classical multirate signal processing, (b) M-channel lazy filter bank on a graph with adjacency matrix $\mathbf{A}$. The decimation matrix $\mathbf{D}$ is as in (6.8) with decimation ratio M.

6.2.4 Polyphase Implementation of Decimation and Interpolation Filters

A useful tool in multirate signal processing is the polyphase representation of linear time-invariant filters [200, 199, 43]. Similar to the classical case, for a given polynomial graph filter, we can write Type-1 polyphase decomposition of the filter as follows:

$$H(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} \mathbf{A}^k \, E_k(\mathbf{A}^M), \tag{6.31}$$

and Type-2 polyphase decomposition as follows:

$$H(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k} \, R_k(\mathbf{A}^M). \tag{6.32}$$

Notice that there is no assumption on the structure of the adjacency matrix, hence any polynomial filter on any graph has a polyphase representation. As in the classical theory [200], Type-1 and Type-2 polyphase components are related as $R_k(\mathbf{A}) = E_{M\text{-}1\text{-}k}(\mathbf{A})$.

Figure 6.4a shows a graph filter followed by a graph decimator on $\mathbf{A}$. This is called a decimation filter, in analogy with classical theory. Similarly the system in Figure 6.5a is called an interpolation filter. For graphs that satisfy the conditions of the noble identities (6.27), these filters can be implemented in simplified form using the polyphase representation as shown next.

Let $T(\mathbf{A})$ denote the overall response of the system in Figure 6.4a. Then we can write it as:

$$T(\mathbf{A}) = \mathbf{D}\, H(\mathbf{A}) = \mathbf{D} \sum_{k=0}^{M\text{-}1} \mathbf{A}^k \, E_k(\mathbf{A}^M) = \sum_{k=0}^{M\text{-}1} \mathbf{D}\, E_k(\mathbf{A}^M)\, \mathbf{A}^k = \sum_{k=0}^{M\text{-}1} E_k(\bar{\mathbf{A}})\, \mathbf{D}\, \mathbf{A}^k, \tag{6.33}$$

where we use the fact that $E_k(\mathbf{A}^M)$ and $\mathbf{A}$ commute since E_k is a polynomial in $\mathbf{A}$, hence it is shift invariant (see Section 6.9). We then utilize the noble identity in (6.25) to get the final result. The adjusted shift operator given in (6.28) is denoted by $\bar{\mathbf{A}}$. Figure 6.4b and Figure 6.4c schematically show the steps in (6.33).

Complementary to (6.33), upsampling followed by a filtering operation can be implemented via Type-2 polyphase decomposition of the filter. Namely, let $T(\mathbf{A})$ denote the overall response of the system in Figure 6.5a. Then we can write it as:

$$T(\mathbf{A}) = H(\mathbf{A})\, \mathbf{D}^{\mathrm{T}} = \sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k} \, R_k(\mathbf{A}^M)\, \mathbf{D}^{\mathrm{T}} = \sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k} \, \mathbf{D}^{\mathrm{T}}\, R_k(\bar{\mathbf{A}}). \tag{6.34}$$

where we use the fact that $R_k(\mathbf{A}^M)$ and $\mathbf{A}$ commute since R_k is a polynomial in $\mathbf{A}$, hence it is shift invariant (see Section 6.9). We then utilize the noble identity in

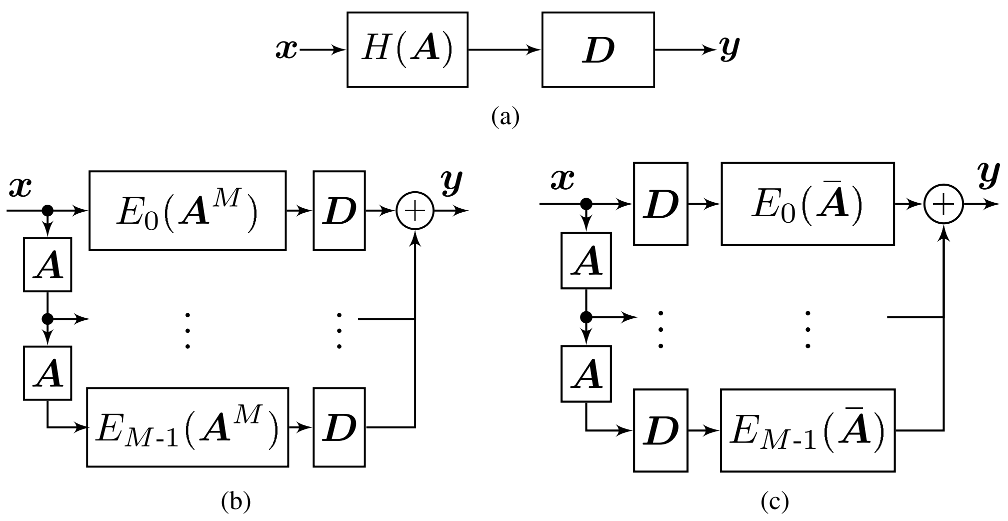


Figure 6.4: (a) Polynomial filtering then decimation operation on a graph with the adjacency matrix **A**. (b) Polyphase implementation of (a). (c) Simplification of (b) using the first noble identity (6.25). The decimation matrix **D** is as in (6.8) with decimation ratio M. Implementation in (b) exists without any restriction on the adjacency matrix. However, **A** should satisfy (6.14) in order to utilize the first noble identity for the implementation in (c).

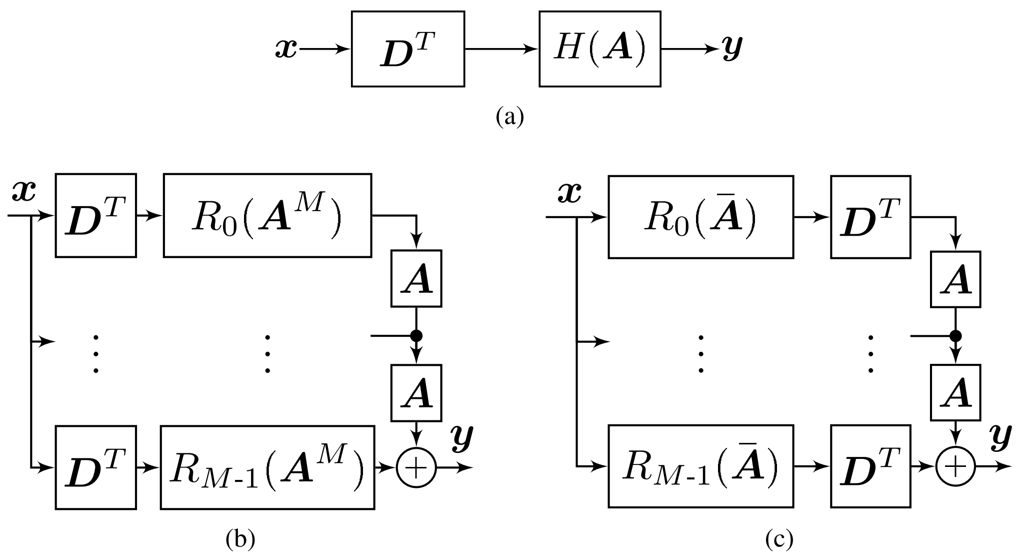


Figure 6.5: (a) Expansion then polynomial filtering on a graph with the adjacency matrix **A**. (b) Polyphase implementation of (a). (c) Simplification of (b) using the second noble identity (6.26). The expansion matrix $\mathbf{D}^{\mathrm{T}}$ with expansion ratio M is transpose of **D**, which is in (6.8). Implementation in (b) exists without any restriction on the adjacency matrix. However, **A** should satisfy (6.21) in order to utilize the second noble identity for the implementation in (c).

(6.26) to get the final result. Figure 6.5b and Figure 6.5c schematically show the steps in (6.34).

We will use polyphase implementation of decimation and interpolation filters when we develop polyphase implementation of filter banks in Section 6.13.

6.3 Graph Filter Banks and Polynomial Filters

The ultimate goal in this thesis is to develop a theory of analysis/synthesis filter banks for graphs with properties such as perfect reconstruction, alias cancellation, and so forth. Figure 6.6 shows such a filter bank for a signal $\mathbf{x} \in \mathbb{C}^N$ defined on the graph $\mathbf{A}$. Here each analysis filter $\mathbf{H}_k$ is an $N \times N$ matrix, and the decimator $\mathbf{D}$ is as defined in Section 6.2. Since there are M analysis filters and each decimator retains N/M samples, this constitutes a *maximally decimated* analysis bank. The expanders $\mathbf{D}^\mathrm{T}$ (defined as in Section 6.2.1) are followed by synthesis filters $\mathbf{F}_k$, which are also $N \times N$ matrices.

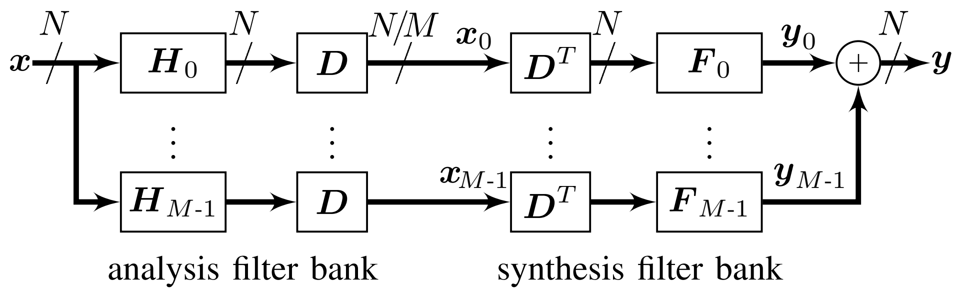


Figure 6.6: A maximally decimated graph filter bank where the filters $\mathbf{H}_k$ and $\mathbf{F}_k$ are arbitrary matrices (i.e., not necessarily polynomials in $\mathbf{A}$).

When the filters and decimators are cascaded, the maximally decimated analysis bank can clearly be defined by the M matrices $\{\mathbf{D}\,\mathbf{H}_0,\ \mathbf{D}\,\mathbf{H}_1,\ \ldots,\ \mathbf{D}\,\mathbf{H}_{M\text{-}1}\}$ where $\mathbf{D}\mathbf{H}_k \in \mathbb{C}^{(N/M)\times N}$. Similarly, the expander and the synthesis filters can be lumped into one matrix $\mathbf{F}_k\,\mathbf{D}^\mathrm{T} \in \mathbb{C}^{N\times(N/M)}$. Therefore, the entire analysis bank, $\mathbf{H}_{\mathrm{anl}}$, and the synthesis bank, $\mathbf{F}_{\mathrm{syn}}$, are just $N \times N$ matrices as follows:

$$\mathbf{F}_{\mathrm{syn}} = \begin{bmatrix} \mathbf{F}_0\,\mathbf{D}^\mathrm{T} & \cdots & \mathbf{F}_{M\text{-}1}\mathbf{D}^\mathrm{T} \end{bmatrix}, \qquad \mathbf{H}_{\mathrm{anl}} = \begin{bmatrix} \mathbf{D}\,\mathbf{H}_0 \\ \vdots \\ \mathbf{D}\,\mathbf{H}_{M\text{-}1} \end{bmatrix} \tag{6.35}$$

Thus, perfect reconstruction property is equivalent to having $\mathbf{F}_{\mathrm{syn}}\mathbf{H}_{\mathrm{anl}} = \mathbf{I}$, so that as long as $\mathbf{H}_{\mathrm{anl}}$ has full rank, we can find synthesis filters for perfect reconstruction. But there are practical difficulties in taking this "brute force" approach with "unconstrained" filter matrices. The complexity of the analysis bank (including decimators) is N^2 **multiplications**, and so is the complexity of the synthesis bank. For large graphs (large N), this complexity can be impractical.

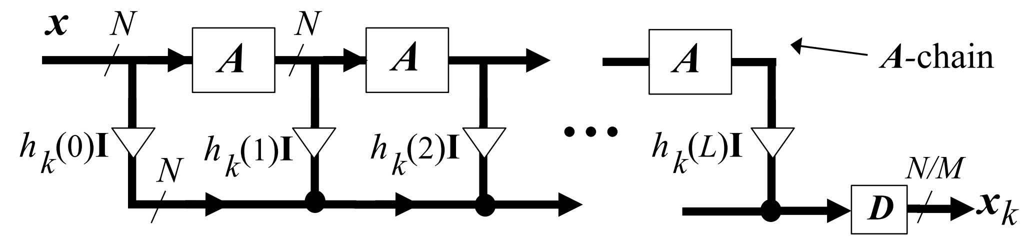


Figure 6.7: Implementation of the polynomial graph filter $H_k(\mathbf{A}) = \sum_{n=0}^{L} h_k(n)\mathbf{A}^n$. When the graph is sparse and has simple edge weights this system requires only LN multiplications for its implementation, compared to N^2 in the brute force method of Figure 6.6.

In Section 6.2.2 we showed that if we use graph filters that are polynomials in $\mathbf{A}$, and insist on the existence of noble identities with canonical decimators and expanders, then some restrictions are imposed on the graph $\mathbf{A}$. So it is necessary here to explain the motivation for using polynomial filters, and reasons for looking into classical tools such as noble identities and so forth, in the graph context. "After all," the astute reader may argue, "we can surely build filters and filter banks for graphs without building these parallels to classical tools." Sure enough, this is a correct statement as we shall now elaborate. But then, there are also some advantages if we build parallels to classical tools, as we shall also explain here. The price paid for the reliance on classical parallels is that the graph $\mathbf{A}$ is restricted in some ways (as we saw in Theorem 6.3, and shall again see in Section 6.4.2). In Section 6.15 we will show how these restrictions can, to some extent, be removed.

The ultimate goal in this thesis is to develop the theory of analysis/synthesis filter banks for graphs with properties such as perfect reconstruction, alias cancellation, and so forth. Figure 6.6 shows such a filter bank for a signal $\mathbf{x} \in \mathbb{C}^N$ defined on the graph $\mathbf{A}$. Here each analysis filter $\mathbf{H}_k$ is an $N \times N$ matrix, and the decimator $\mathbf{D}$ is as defined in Section 6.2. Since there are M analysis filters and each decimator retains N/M samples, this constitutes a *maximally decimated* analysis bank. The expanders $\mathbf{D}^{\mathrm{T}}$ (defined as in Section 6.2.1) are followed by synthesis filters $\mathbf{F}_k$ which are also $N \times N$ matrices.

The maximally decimated analysis bank can clearly be defined by the M matrices $\{\mathbf{DH}_0,\ \mathbf{DH}_1,\ \ldots,\ \mathbf{DH}_{M\text{-}1}\}$, and represented as shown in Figure 6.8 on the left. Similarly, with the expander $\mathbf{D}^{\mathrm{T}}$ and the synthesis filters $\mathbf{F}_k$ lumped into one matrix

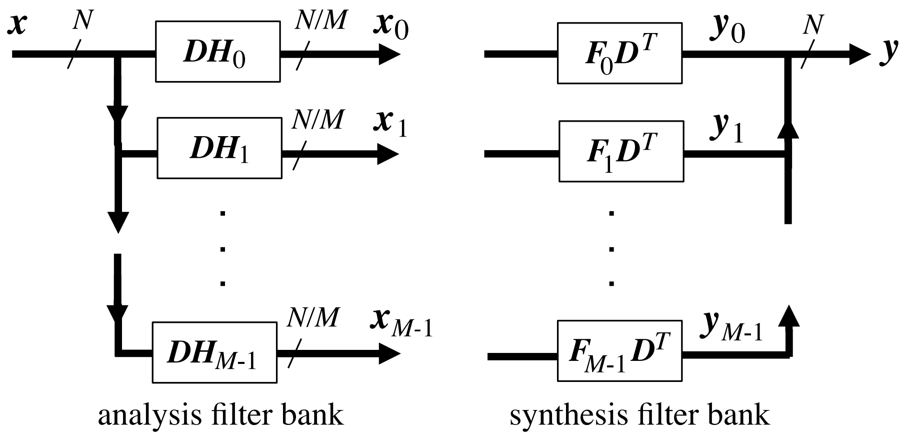


Figure 6.8: The maximally decimated graph filter bank redrawn. See text.

$\mathbf{F}_k\,\mathbf{D}^{\mathrm{T}}$, the synthesis bank can be drawn as shown on the right in Figure 6.8. Notice from the figure that the entire analysis bank is just an $N \times N$ matrix $\mathbf{H}_{\text{anal}}$ and the synthesis bank is another $N \times N$ matrix $\mathbf{F}_{\text{syn}}$ as follows:

$$\mathbf{F}_{\text{syn}} = \begin{bmatrix} \mathbf{F}_0\,\mathbf{D}^{\mathrm{T}} & \cdots & \mathbf{F}_{M\text{-}1}\,\mathbf{D}^{\mathrm{T}} \end{bmatrix}, \qquad \mathbf{H}_{\text{anal}} = \begin{bmatrix} \mathbf{D}\,\mathbf{H}_0 \\ \vdots \\ \mathbf{D}\,\mathbf{H}_{M\text{-}1} \end{bmatrix}. \tag{6.36}$$

Thus perfect reconstruction is equivalent to $\mathbf{F}_{\text{syn}}\,\mathbf{H}_{\text{anal}} = \mathbf{I}$, so that as long as $\mathbf{H}_{\text{anal}}$ has full rank N, we can find synthesis filters for perfect reconstruction. In short the design of maximally decimated PR filter banks for graphs appears to be a trivial matrix-inversion problem, deserving no deeper attention. But there are practical difficulties in taking this "brute force" approach with "unconstrained" filter matrices. The complexity of the analysis bank (including decimators) is N^2 multiplications, and so is the complexity of the synthesis bank. For large graphs (large N), this complexity can be impractical. From a conceptual point of view, the number of graph vertices N is analogous to the extent of time domain in classical case, and the of extent time domain is usually assumed to be infinite, i.e., $-\infty < n < \infty$. *So, the unconstrained system of Fig. 6.8 (where filters are not polynomials) is "analogous" to infinite-order filters (e.g., ideal filters) in the classical time domain case.*

There are other ways to design maximally decimated graph filter banks with much lower complexity. One of these is to constrain the filters to be polynomials in the graph adjacency matrix $\mathbf{A}$. That is, we take the k^{th} analysis filter to be of the form

$$\mathbf{H}_k = h_k(0)\,\mathbf{I} + h_k(1)\,\mathbf{A} + h_k(2)\,\mathbf{A}^2 + \ldots + h_k(L)\,\mathbf{A}^L, \tag{6.37}$$

and similarly for the synthesis filters, where the filter order L is a flexible design parameter. This is in analogy with FIR filters $H_k(z) = \sum_{n=0}^{L} h_k(n)\, z^{-n}$ used in classical filter bank designs. The advantage is that the graph filters can now be implemented as in Figure 6.7 where the scalars $h_k(n)$ are the coefficients of the k^{th} filter. The cascade of $\mathbf{A}$ matrices, called the $\mathbf{A}$-*chain* (analogous to the classical delay chain), is common to all the analysis filters and can be *shared*. This implementation is especially attractive when $\mathbf{A}$ is sparse and has simple entries like $0, 1, -1$, etc., as in many practical graphs (e.g., the Minnesotta traffic graph in [129, 128]). In this case the implementation of the matrix multipliers $\mathbf{A}$ has negligible complexity. Each multiplier $h_k(n)\mathbf{I}$ is a diagonal matrix requiring N multipliers. But since the decimator keeps only N/M samples, only N/M multipliers in $h_k(n)\mathbf{I}$ are needed before we add the signals in the bottom of Figure 6.7. So there are about LN/M multiplications per channel. So the entire maximally decimated analysis bank requires only LN multiplications, compared to the N^2 multipliers in the case of unconstrained filters. This is a significant saving when $L \ll N$.

As mentioned earlier, the number of graph vertices N is analogous to the extent of time domain in the classical time domain case, which is usually assumed to be infinite. The unconstrained system of Figure 6.8 (where filters are not polynomials) is analogous to infinite order filters (ideal filters) in the classical time domain. The polynomial graph filter bank (filters as in Figure 6.7) is analogous to FIR filter banks in the classical case. In the classical case short FIR filters can approximate the ideal infinite-order filter quite well. In the same way, we expect that short polynomial graph filters can approximate arbitrary $N \times N$ matrix filters “well enough.” We realize that at this point the idea is rather in its infancy. But since the design of such graph filters is itself likely to be a major topic, it is not elaborated further here.

The next question is, how do we design such polynomial filter banks and how do we achieve properties similar to perfect reconstruction, freedom from aliasing, and so forth. Since the filters are polynomials, one approach would be to draw a parallel with classical filter banks and develop an analogy for the most basic tools used therein such as noble identities, polyphase representations, and so forth. This may or may not be the best approach, but it is certainly of interest to explore this avenue, and find conditions under which this will work. This is the viewpoint we take. We will find that developing such a parallel to classical tools imposes some restrictions on the graph (Section 6.2 and Section 6.4.2). We will also show how some of these restrictions can be removed (Section 6.15).

In the next section we return to a deeper study of the properties of graphs which allow such parallels to be built. A detailed study of graph filter banks and polyphase matrices will be given in Sections 6.10 - 6.13, where the advantage of polyphase matrices in the context of graph filter banks will also be explained. As a minor point, it should also be noticed that once the advantage of polynomial filters and other parallels to classical multirate systems is established, it is really not "necessary" (although aesthetically satisfying) to look for a "physical" meaning for the graph shift operator **A** [152, 153]. The real benefit comes from polynomial filters, and the ability to use classical tools.

6.4 Relations to M-Partite Graphs

From Theorem 6.1 and 6.2, it is clear that some structure on the adjacency matrix is required in order to generalize the basic concepts in the classical multirate signal processing theory to graph signals. In this section we consider a number of examples of graphs (especially M-partite graphs) and examine whether some or all of these conditions are satisfied.

6.4.1 Some Important Examples

We begin with the example where **A** is an arbitrary diagonal matrix of size N. In this case, it is clear that the noble identity condition (6.27) is satisfied for any M that divides N. However, consider the overall response of the lazy filter bank in Figure 6.3:

$$T(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}k}\ \mathbf{D}^{\mathrm{T}}\ \mathbf{D}\ \mathbf{A}^{k} = \begin{bmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{6.38}$$

where **X** is a diagonal matrix of size N/M and zero blocks have appropriate sizes. This $T(\mathbf{A})$ cannot be written as the $(M\text{-}1)^{th}$ power of the adjacency matrix unless **A** has zeros on the diagonal except for the first N/M place. Therefore, the lazy filter bank is not a PR system even for the simple example where the graph adjacency matrix is diagonal! The main point of this example is that some of the "obvious" results of classical multirate theory can fail in the case of some graphs. The fact that the diagonal matrix **A** is not a useful graph is secondary to this discussion.

For the next example, consider bipartite graphs. In [129], bipartite graphs are shown to be useful for 2-channel filter banks where the development was based on the graph Laplacian. We now claim the following:

Theorem 6.4 (Bipartite graphs and filter banks)**.** *Let* **A** *be the adjacency matrix of* *a directed or undirected, bipartite graph in the following form*

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_2 \\ \mathbf{A}_1 & \mathbf{0} \end{bmatrix}, \tag{6.39}$$

where $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{C}^{(N/2)\times(N/2)}$*. Then, noble identity condition in* (6.27) *and the lazy filter bank perfect reconstruction condition in* (6.30) *are both satisfied for* $M = 2$*.*

Proof. If the adjacency matrix of $\mathbf{A}$ has the form in (6.39), then we have

$$\mathbf{A}^2 = \begin{bmatrix} \mathbf{A}_2\,\mathbf{A}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_1\,\mathbf{A}_2 \end{bmatrix}. \tag{6.40}$$

Since $\mathbf{A}^M = \mathbf{A}^2$ satisfies the necessary structure given by Theorem 6.3, the noble identities (6.25) and (6.26) are satisfied with the adjusted shift operator $\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}^2\,\mathbf{D}^{\mathrm{T}} = \mathbf{A}_2\,\mathbf{A}_1$ for any polynomial filter $H(\cdot)$.

Next, for the lazy filter bank notice that

$$\mathbf{A}^1\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^0 = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{A}_1 & \mathbf{0} \end{bmatrix}, \quad \mathbf{A}^0\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^1 = \begin{bmatrix} \mathbf{0} & \mathbf{A}_2 \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \tag{6.41}$$

Therefore, $T(\mathbf{A}) = \sum_{k=0}^{1} \mathbf{A}^{1\text{-}k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^k = \mathbf{A}$, hence the lazy filter bank provides perfect reconstruction. □

Even though Theorem 6.4 is stated for bipartite graphs with equal sized partitions, it extends to arbitrary bipartite graphs with a proper update on the size of the decimation operator. More importantly, we also have the following result:

Theorem 6.5. *If a graph with the adjacency matrix* $\mathbf{A}$ *provides PR in the* 2*-channel lazy filter bank, then the graph is necessarily bi-partite.*

Proof. Let $\mathbf{A}$ be partitioned as

$$\mathbf{A} = \begin{bmatrix} (\mathbf{A})_{1,1} & (\mathbf{A})_{1,2} \\ (\mathbf{A})_{2,1} & (\mathbf{A})_{2,2} \end{bmatrix}, \tag{6.42}$$

where $(\mathbf{A})_{1,1} \in \mathbb{C}^{(N/2)\times(N/2)}$. For $M = 2$, the lazy FB PR condition in (6.30) becomes $\mathbf{A}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D} + \mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A} = \mathbf{A}$. When (6.12) and (6.42) are considered, the lazy FB PR condition results in $(\mathbf{A})_{1,1} = \mathbf{0}$ and $(\mathbf{A})_{2,2} = \mathbf{0}$. Since $\mathbf{A}$ has the form in (6.39), the graph is bi-partite. □

6.4.2 M-partite graphs

An M-partite graph is one whose vertex set can be partitioned into M subsets so that no edge has both ends in any one subset [26]. Under suitable labelling of the vertices, the adjacency matrix of an M-partite graph can be written as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & (\mathbf{A})_{1,2} & \cdots & (\mathbf{A})_{1,M} \\ (\mathbf{A})_{2,1} & \mathbf{0} & \ddots & \vdots \\ \vdots & \ddots & \ddots & (\mathbf{A})_{M\text{-}1,M} \\ (\mathbf{A})_{M,1} & \cdots & (\mathbf{A})_{M,M\text{-}1} & \mathbf{0} \end{bmatrix} \tag{6.43}$$

where $(\mathbf{A})_{i,j}$'s have arbitrary but appropriate sizes. In particular, the diagonal blocks of the adjacency matrix of an arbitrary M-partite graph are zero. An M-partite graph is *balanced* when each vertex set has the same size, that is, $(\mathbf{A})_{i,j} \in \mathbb{C}^{(N/M)\times(N/M)}$.

Even though bipartite graphs are in conformity with the 2-channel systems as shown in Theorem 6.4 and 6.5, this relation cannot be generalized to M-channel systems on M-partite graphs due to following result.

Theorem 6.6 (Noble identities and M-partite graphs)**.** *For $M > 2$, the M-partite property* (6.43) *is neither necessary nor sufficient for validity of the noble identity condition in* (6.27).

Proof. We already know that any diagonal matrix $\mathbf{A}$ (which is clearly not M-partite) satisfies the noble identity condition, so the M-partite property is not necessary. To prove it is not sufficient, we construct a counter example: let $\mathbf{A}$ be as in (6.43) and assume all the elements in all the non diagonal blocks $(\mathbf{A})_{i,j}$ are strictly positive. Now consider the product $\mathbf{A}^2$. Its $(i,j)^{th}$ block has the form

$$(\mathbf{A}^2)_{i,j} = \sum_{k=1}^{M} (\mathbf{A})_{i,k}\,(\mathbf{A})_{k,j}. \tag{6.44}$$

Since $M > 2$ it follows that there is at least one term $(\mathbf{A})_{i,k}(\mathbf{A})_{k,j}$ such that $k \neq i$ and $k \neq j$. And since every element of the matrix $(\mathbf{A})_{m,n}$ is strictly positive unless $m = n$, it follows that every element of $(\mathbf{A})_{i,k}(\mathbf{A})_{k,j}$ is strictly positive. Thus every element of $\mathbf{A}^2$ is strictly positive. Repeating this argument, it follows that every element of $\mathbf{A}^M$ is strictly positive. So the noble identity condition (6.27) can never be satisfied by this M-partite example! □

Moreover, an arbitrary M-partite graph does not satisfy the lazy filter bank condition. We state this observation as the following theorem.

Theorem 6.7 (Lazy filter banks and M-partite graphs)**.** *For $M > 2$, the M-partite property of the graph is in general not sufficient to ensure perfect reconstruction property of the lazy filter bank of Figure 6.3b.*

Proof. We will provide a counter-example to disprove the sufficiency. Let $\mathbf{A}$ be the adjacency matrix of a balanced M-partite graph. Therefore it can be written as in (6.43). Let $(\mathbf{A})_{i,j} = \mathbf{I}_{N/M}$ for all $i \neq j$. Therefore $\mathbf{A}$ can be written as

$$\mathbf{A} = (\mathbf{1}\mathbf{1}^{\mathrm{T}} - \mathbf{I}_M) \otimes \mathbf{I}_{N/M},$$

where $\mathbf{1} \in \mathbb{C}^M$ is a column vector with all 1 entries. Then we have the following equation for the powers of $\mathbf{A}$:

$$\mathbf{A}^k = (-1)^k \Big(\alpha_k \, \mathbf{1}\mathbf{1}^{\mathrm{T}} + \mathbf{I}_M\Big) \otimes \mathbf{I}_{N/M}, \tag{6.45}$$

where

$$\alpha_k = \frac{(1\text{-}M)^k\text{-}1}{M}.$$

To see this, first observe that $\mathbf{A}^0$ and $\mathbf{A}^1$ satisfy (6.45) with $\alpha_0 = 0$ and $\alpha_1 = \text{-}1$. Now assume that (6.45) is correct for $\mathbf{A}^k$. Then we have the following for $\mathbf{A}^{k+1}$:

$$\begin{aligned}\mathbf{A}^{k+1} = \mathbf{A}^k \, \mathbf{A} &= (-1)^k \Big(\alpha_k \, \mathbf{1}\mathbf{1}^{\mathrm{T}} + \mathbf{I}_M\Big)(\mathbf{1}\,\mathbf{1}^{\mathrm{T}} - \mathbf{I}_M) \otimes \mathbf{I}_{N/M}, \\ &= (-1)^k \Big(\mathbf{1}\,\mathbf{1}^{\mathrm{T}}(\alpha_k M - \alpha_k + 1) - \mathbf{I}_M\Big) \otimes \mathbf{I}_{N/M}, \\ &= (-1)^{k+1} \Big(\mathbf{1}\mathbf{1}^{\mathrm{T}}(-\alpha_k M + \alpha_k - 1) + \mathbf{I}_M\Big) \otimes \mathbf{I}_{N/M}.\end{aligned}$$

Hence, we have the form in (6.45) also for $\mathbf{A}^{k+1}$ with

$$\alpha_{k+1} = \alpha_k(1\text{-}M) - 1.$$

When we expand the recursive relation above, we get

$$\alpha_k = \Big(\alpha_{k\text{-}2}(1\text{-}M)\text{-}1\Big)(1\text{-}M)\text{-}1 = \cdots = \text{-}\sum_{l=0}^{k-1}(1\text{-}M)^l = \frac{(1\text{-}M)^k\text{-}1}{M}.$$

Notice that this satisfies, $\alpha_0 = 0$ and $\alpha_1 = \text{-}1$.

From (6.12), we have that $\mathbf{D}^{\mathrm{T}}\,\mathbf{D} = \big(\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{T}}\big) \otimes \mathbf{I}_{N/M}$ where $\mathbf{e}_1$ is the first vector of the standard basis for $\mathbb{C}^M$. Remember that overall response of the lazy filter bank is given in (6.29). When we substitute (6.45) into (6.29), we get

$$\begin{aligned}T(\mathbf{A}) &= (\text{-}1)^{M\text{-}1}\sum_{k=0}^{M\text{-}1}\Big(\alpha_{M\text{-}1\text{-}k}\;\mathbf{1} + \mathbf{e}_1\Big)\Big(\alpha_k\,\mathbf{1} + \mathbf{e}_1\Big)^{\mathrm{T}} \otimes \mathbf{I}_{N/M} \\ &= (\text{-}1)^{M\text{-}1}\Big(c_1\,\mathbf{1}\,\mathbf{1}^{\mathrm{T}} + c_2\,\big(\mathbf{1}\,\mathbf{e}_1^{\mathrm{T}} + \mathbf{e}_1\,\mathbf{1}^{\mathrm{T}}\big) + M\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{T}}\Big) \otimes \mathbf{I}_{N/M},\end{aligned}$$

where c_2 is given as

$$c_2 = \sum_{k=0}^{M\text{-}1} \alpha_k = \sum_{k=0}^{M\text{-}1} \frac{(1-M)^k - 1}{M} = -\left(\frac{(1-M)^M - 1}{M^2} + 1\right)$$

and c_1 is given as

$$c_1 = \sum_{k=0}^{M\text{-}1} \alpha_{M\text{-}1\text{-}k}\ \alpha_k = \frac{(1\text{-}M)^{(M\text{-}1)} - 2\,c_2 - 1}{M}.$$

Now consider the matrix $\mathbf{A}^k$ from (6.45). For the $(1,1)^{th}$ block we have $(\mathbf{A}^k)_{1,1} = (\text{-}1)^k\ (\alpha_k{+}1)\,\mathbf{I}_{N/M}$ whereas we have $(\mathbf{A}^k)_{1,2} = (\text{-}1)^k\ (\alpha_k)\,\mathbf{I}_{N/M}$ for the $(1,2)^{th}$ block. Hence, we have

$$(\mathbf{A}^k)_{1,1} - (\mathbf{A}^k)_{1,2} = (\text{-}1)^k\,\mathbf{I}_{N/M}$$

for all k. When we look at $T(\mathbf{A})$, we have

$$\big(T(\mathbf{A})\big)_{1,1} - \big(T(\mathbf{A})\big)_{1,2} = (\text{-}1)^{M\text{-}1}\,(c_2 + M)\,\mathbf{I}_{N/M}.$$

In order to have $T(\mathbf{A})$ in the form of (6.45), we therefore need $c_2 + M = \pm 1$, which in turn implies

$$M \pm 1 = \frac{(1\text{-}M)^M - 1}{M^2} + 1.$$

Notice that the above equation (either plus or minus case) is satisfied only for $M = \{1, 2\}$ for an integer M. For $M > 2$, we conclude that $T(\mathbf{A})$ does not have the form in (6.45), hence it is not a power of $\mathbf{A}$. □

6.5 M-block cyclic graphs

Contrary to intuition, the two channel filter bank results on bipartite graphs do not extend to M-channel filter banks on M-partite graphs, as discussed in Section 6.4. In the following, we will show that, with more restrictive conditions on the graph, it is possible to generalize the classical multirate theory to graph signals for arbitrary M. For this purpose we define the following graph.

Definition 6.5 (M-block cyclic graphs)**.** *A graph is said to be M-block cyclic if the adjacency matrix of the graph has the following form:*

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{A}_M \\ \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_3 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_{M\text{-}1} & \mathbf{0} \end{bmatrix} \in \mathbb{C}^{N \times N}, \tag{6.46}$$

where each $\mathbf{A}_j$ has arbitrary but appropriate sizes. Furthermore, such a graph is said to be balanced *M-block cyclic, when $\mathbf{A}_j$'s have the same size, that is $\mathbf{A}_j \in \mathbb{C}^{(N/M)\times(N/M)}$. In this case, we can write the adjacency matrix as*

$$(\mathbf{A})_{i,j} = \mathbf{A}_j \ \delta(j\text{-}i\text{+}1), \tag{6.47}$$

where $(\cdot)_{i,j}$ denotes the $(i,j)^{th}$ block of the adjacency matrix and $\delta(\cdot)$ is the M-periodic discrete Dirac function, that is $\delta(Mj) = 1$ for all integer j and zero otherwise.

In the rest of the chapter, when we refer to M-block cyclic graphs, *we always mean balanced M-block cyclic graphs*. Some of the results presented in this study can be generalized to unbalanced M-block cyclic graphs also. However, the adjacency matrix of an unbalanced M-block cyclic graph can be shown to be non-invertible and non-diagonalizable. Such a case requires a careful treatment that falls outside of the scope of this study and will be elaborated elsewhere.

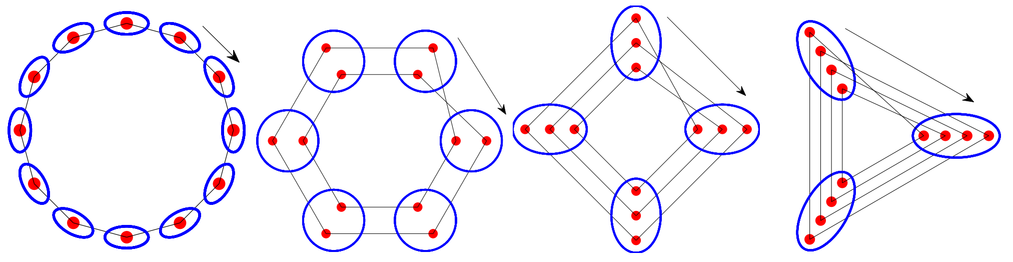
For the visual representation of M-block cyclic graphs, see Figure 6.10a for a balanced 5-block cyclic graph of size 20. Also consider Figure 6.9 to see the relation between the cyclic shift matrix in (6.1) and M-block cyclic matrices. We now state some properties of M-block cyclic graphs that can be readily verified:

Fact 6.1. *If a graph is M-block cyclic, then it is M-partite, but not vice-versa.*

Fact 6.2. *A graph is 2-block cyclic if and only if it is bi-partite.*

Fact 6.3. *An M-block cyclic graph is necessarily a directed graph for $M > 2$, hence its adjacency matrix does not have any symmetry property in terms of edge weights.*

Fact 6.4. *A cyclic graph of size N, $\mathbf{C}_N$, is an M-block cyclic graph for all M that divides N. See Figure 6.9.*



(a) $\mathbf{C}_{12}$ as 12-block cyclic (b) $\mathbf{C}_{12}$ as 6-block cyclic (c) $\mathbf{C}_{12}$ as 4-block cyclic (d) $\mathbf{C}_{12}$ as 3-block cyclic

Figure 6.9: Under suitable permutation of the vertices, cyclic graph of size N can be represented as an M-block cyclic graph of size N where M divides N. Notice that cyclic graph of size N is equivalent to N-block cyclic graph of size N. All the edges are directed clock-wise as indicated by the arrow.

Some other properties of the adjacency matrix of an M-block cyclic graph are presented in Section 6.18.1.

Even though arbitrary M-partite graphs are not suitable for M-channel systems as discussed in Section 6.4, imposing more restrictions and having M-block cyclic structure in (6.46) provides much more freedom in terms of multirate processing on graphs, which is formally stated in the following theorem.

Theorem 6.8 (M-block cyclic graphs, noble identities, and lazy filter banks)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of a balanced M-block cyclic graph. Then, noble identity condition in Theorem 6.3 and lazy FB PR condition in* (6.30) *are satisfied.*

Proof. According to Corollary 6.4 in Section 6.18.1, $\mathbf{A}^M$ is a block diagonal matrix with blocks of size $\mathbb{C}^{(N/M)\times(N/M)}$, which satisfies the condition in Theorem 6.3. Therefore, noble identities hold true with the adjusted shift operator

$$\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}^M\,\mathbf{D}^{\mathrm{T}} = \mathbf{A}_M\cdots\mathbf{A}_1. \tag{6.48}$$

For the lazy filter bank condition, consider Corollary 6.6 and 6.7 in Section 6.18.1. Since $\mathbf{A}^{-k}\mathbf{D}^{\mathrm{T}}$ is a block-column vector and $\mathbf{D}\mathbf{A}^k$ is a block-row vector, we have

$$\left(\mathbf{A}^{-k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^k\right)_{i,j} = (\mathbf{A}^{-k}\mathbf{D}^{\mathrm{T}})_i\;(\mathbf{D}\mathbf{A}^k)_j = \mathbf{I}_{N/M}\;\;\delta(i\text{-}1\text{+}k)\;\delta(j\text{-}1\text{+}k).$$

Therefore,

$$\left(\sum_{k=0}^{M-1}\mathbf{A}^{-k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^k\right)_{i,j} = \mathbf{I}_{N/M}\;\;\delta(i\text{-}j),$$

that is $\sum_{k=0}^{M\text{-}1} \mathbf{A}^{\text{-}k}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^{k} = \mathbf{I}_N$. Hence, $T(\mathbf{A}) = \mathbf{A}^{M\text{-}1}$, that is, M-channel lazy filter bank provides perfect reconstruction due to condition in (6.30). Notice that this proof implicitly assumes that $\mathbf{A}$ is invertible. However, the result still holds true even if the adjacency matrix is not invertible as long as it is M-block cyclic. We omit these details for brevity. □

6.6 Eigen-properties of M-block cyclic graphs

M-block cyclic graphs have an important eigenvalue-eigenvector structure that will play a key role in the development of graph filter banks. This property is as follows.

Theorem 6.9 (Eigen-families of M-block cyclic graphs)**.** *Eigenvalues and eigenvectors of the adjacency matrix of an M-block cyclic graph come as families of size M. That is, if $(\lambda, \mathbf{v})$ is an eigenpair of M-block cyclic graph, then $\{(\lambda,\mathbf{v}),\ (w\lambda,\mathbf{\Omega v}),\ (w^2\lambda,\mathbf{\Omega}^2\mathbf{v}),\ \cdots\ (w^{M\text{-}1}\lambda,\mathbf{\Omega}^{M\text{-}1}\mathbf{v})\}$ are all eigenpairs of the same graph, where*

$$w = e^{-j2\pi/M}, \tag{6.49}$$

$$\mathbf{\Omega} = \mathrm{diag}\left(\left[1\ w^{\text{-}1}\ w^{\text{-}2}\ \cdots\ w^{\text{-}(M\text{-}1)}\right]\right) \otimes \mathbf{I}_{N/M}. \tag{6.50}$$

Proof. Let $(\lambda, \mathbf{v})$ be an eigenpair of a balanced M-block cyclic graph. Assume that we have the following partitions for the eigenvector

$$\mathbf{v} = \begin{bmatrix}(\mathbf{v})_1^{\mathrm{H}} & (\mathbf{v})_2^{\mathrm{H}} & \cdots & (\mathbf{v})_M^{\mathrm{H}}\end{bmatrix}^{\mathrm{H}}, \tag{6.51}$$

where $(\mathbf{v})_i \in \mathbb{C}^{N/M}$ for all $1 \leq i \leq M$. Then,

$$\mathbf{Av} = \begin{bmatrix} \mathbf{A}_M\ (\mathbf{v})_M \\ \mathbf{A}_1\ (\mathbf{v})_1 \\ \vdots \\ \mathbf{A}_{M\text{-}1}\ (\mathbf{v})_{M\text{-}1} \end{bmatrix} = \lambda\mathbf{v} = \begin{bmatrix} \lambda\ (\mathbf{v})_1 \\ \lambda\ (\mathbf{v})_2 \\ \vdots \\ \lambda\ (\mathbf{v})_M \end{bmatrix}, \tag{6.52}$$

that is,

$$\mathbf{A}_i\ (\mathbf{v})_i = \lambda\ (\mathbf{v})_{i+1}. \tag{6.53}$$

When both sides of (6.53) are multiplied by $w^{1\text{-}i}$, we get

$$\mathbf{A}_i\ \big(w^{1\text{-}i}(\mathbf{v})_i\big) = \big(w\lambda\big)\ \big(w^{\text{-}i}(\mathbf{v})_{i+1}\big). \tag{6.54}$$

Therefore $w\lambda$ is also an eigenvalue with the corresponding eigenvector

$$\mathbf{v}' = \begin{bmatrix}w^0\,(\mathbf{v})_1^{\mathrm{T}} & w^{\text{-}1}\,(\mathbf{v})_2^{\mathrm{T}} & \cdots & w^{\text{-}(M\text{-}1)}\,(\mathbf{v})_M^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}.$$

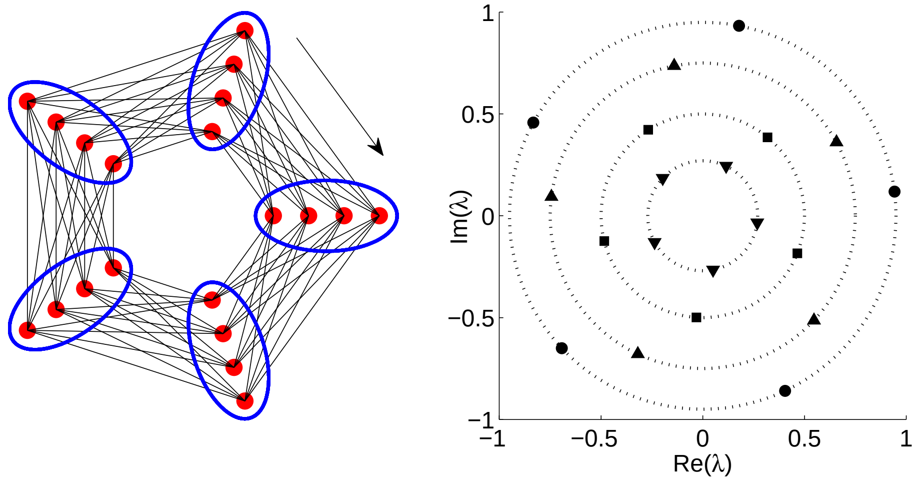
Due to definition of $\mathbf{\Omega}$ in (6.50), we have the following:

$$\left[w^0\,(\mathbf{v})_1^{\mathrm{T}}\ w^{-1}\,(\mathbf{v})_2^{\mathrm{T}}\ \cdots\ w^{-(M-1)}\,(\mathbf{v})_M^{\mathrm{T}}\right]^{\mathrm{T}} = \mathbf{\Omega}\ \mathbf{v}, \tag{6.55}$$

hence, $(w\lambda, \mathbf{\Omega v})$ is also an eigen-pair.

Iterating this argument k times, we get $(w^k\lambda,\ \mathbf{\Omega}^k\mathbf{v})$ as an eigenpair. However notice that $w^{M+k} = w^k$ and $\mathbf{\Omega}^{M+k} = \mathbf{\Omega}^k$. Therefore, starting from an eigenpair and iteratively using (6.54), we can produce at most M-1 distinct eigenpairs. As a result, if $(\lambda, \mathbf{v})$ is an eigenpair, $(w^k\lambda, \mathbf{\Omega}^k\mathbf{v})$ is also an eigenpair for $0 \leq k \leq M$-1. $\square$

This eigenvalue relation of block cyclic matrices has also been observed in earlier studies [209, 64, 119, 45].



(a) (b)

Figure 6.10: (a) 5-block cyclic graph of size 20, (b) eigenvalues of the graph. Notice that all the edges are directed along with the clock-wise direction and they have complex valued weights. As given in Theorem 6.9, eigenvalues of a balanced M-block cyclic graph come as families of size M. Eigenvalues belonging to the same family are equally spaced on a circle in the complex plane. Actual values of the eigenvalues depend on the weight of the edges.

Figure 6.10b visualizes the relation between the eigenvalues of an M-block cyclic graph. There are N/M concentric circles centered at the origin. Each circle has M eigenvalues equispaced in angle. The circles need not have distinct radii.

One immediate consequence of this eigenfamily structure of the M-block cyclic graph is that eigenvalues can be real only for $M = 2$. We formally state this property as follows.

Corollary 6.1 (Complex eigenvalues of M-block cyclic)**.** *For $M > 2$, if an M-block cyclic graph has a non-zero eigenvalue, then it has at least one complex valued eigenvalue.*

Proof. Let λ be a non-zero eigenvalue of an M-block cyclic graph. Then $w^k\lambda$ is also eigenvalue for $0 \leq k \leq M$-1 due to Theorem 6.9. Therefore for $M > 2$, there exists a k such that $w^k\lambda$ is complex valued. □

It should be clear that Theorem 6.9 gives information about only one eigen-family and does *not* imply diagonalizability of the adjacency matrix in general. When $\mathbf{A}$ is not diagonalizable, Theorem 6.9 still applies to its proper eigenvectors, whereas we cannot say too much for the generalized eigenvectors coming from the Jordan chain. However, we note that a randomly generated balanced M-block cyclic matrix is diagonalizable with probability 1.

Assuming that the adjacency matrix is diagonalizable, we will use double indexing to represent the eigenvalues and the eigenvectors of M-block cyclic graphs, since they come as families of size M. That is, the eigenpair $(\lambda_{i,j}, \mathbf{v}_{i,j})$ will denote the j^{th} eigenpair of the i^{th} family, where $1 \leq i \leq N/M$ and $1 \leq j \leq M$. Using this indexing scheme, with the use of Theorem 6.9, we have the following form:

$$\lambda_{i,\,j+k} = w^k\ \lambda_{i,j}, \tag{6.56}$$

$$\mathbf{v}_{i,\,j+k} = \mathbf{\Omega}^k\ \mathbf{v}_{i,j}. \tag{6.57}$$

It is important to state that this indexing scheme has a circular structure. Even though we do not explicitly indicate this fact in the notation, it should be clear that $\lambda_{i,j+M} = \lambda_{i,j}$ and $\mathbf{v}_{i,j+M} = \mathbf{v}_{i,j}$ for all i and j. This property comes from the fact that $w^M = 1$ and $\mathbf{\Omega}^M = \mathbf{I}$.

With this specific family structure of the eigenvalues of an M-block cyclic graph, when we talk about the eigenvalue decomposition of the adjacency matrix,

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{-1}, \tag{6.58}$$

we will assume that eigenvalues and the eigenvectors are ordered as follows:

$$\mathbf{\Lambda} = \text{diag}\left([\lambda_{1,1}\cdots\lambda_{1,M}\ \ \cdots\ \ \lambda_{N/M,1}\cdots\lambda_{N/M,M}]\right), \tag{6.59}$$

$$\mathbf{V} = \left[\mathbf{v}_{1,1}\cdots\mathbf{v}_{1,M}\ \ \cdots\ \ \mathbf{v}_{N/M,1}\cdots\mathbf{v}_{N/M,M}\right]. \tag{6.60}$$

It is also important to notice that the eigenfamily structure described in (6.56) and (6.57) is unique to M-block cyclic graphs. This fact is stated in the following theorem whose proof is given in Section 6.18.2.

Theorem 6.10 (Eigen-structure of M-block cyclic graphs)**.** *Let* $\mathbf{V}$ *be an invertible matrix indexed as in* (6.60) *with columns that have the property in* (6.57)*. Let* $\mathbf{\Lambda}$ *be a diagonal matrix indexed as in* (6.59) *with diagonal entries that have the property in* (6.56)*. Then* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ *is diagonalizable* M*-block cyclic graph. Conversely the adjacency matrix of a diagonalizable* M*-block cyclic graph always has the form* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ *where* $\mathbf{V}$ *and* $\mathbf{\Lambda}$ *are as described above.*

In order to enhance our motivation for M-block cyclic graphs, we would like to consider a specific case where $M = 2$. Due to Fact 6.2 in Section 6.5, this is equivalent to bipartite graphs.

For bipartite graphs, we now present Theorem 6.11 given below. In [129], 2-channel filter banks on bipartite graphs are developed using this result from the spectral graph theory. Note here that the Laplacian of a graph is given as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix and the normalized Laplacian is given as $\mathcal{L} = \mathbf{D}^{-1/2}\,\mathbf{L}\,\mathbf{D}^{-1/2}$.

Theorem 6.11 (Lemma 1.8 in [39] or Lemma 1 in [129])**.** *The following statements are equivalent for an undirected graph with real non-negative edge weights:*

1. $\mathbf{A}$ *is bipartite.*
2. *The spectrum of* $\mathcal{L}$ *is symmetric about 1 and the minimum and maximum eigenvalues of* $\mathcal{L}$ *are 0 and 2, respectively.*
3. *If* $\mathbf{v} = [(\mathbf{v})_1^{\mathrm{T}}\ (\mathbf{v})_2^{\mathrm{T}}]^{\mathrm{T}}$ *is an eigenvector of* $\mathcal{L}$ *with eigenvalue* λ*, then the deformed vector* $\widehat{\mathbf{v}} = [(\mathbf{v})_1^{\mathrm{T}}\ \text{-}(\mathbf{v})_2^{\mathrm{T}}]^{\mathrm{T}}$ *is also an eigenvector of* $\mathcal{L}$ *with eigenvalue* $2\text{-}\lambda$*.*

Notice that Theorem 6.11 is valid for the normalized Laplacian of the graph. Since we work directly on the adjacency matrix rather than the Laplacian, we will not utilize this result in our development. Interestingly, Theorem 6.9 provides a very similar statement for the adjacency matrix of the graph when $M = 2$. To see this, observe the following corollary.

Corollary 6.2 (Bipartite as 2-block cyclic)**.** *If* λ *is an eigenvalue of the adjacency matrix of an arbitrary balanced bipartite graph with the eigenvector* $\mathbf{v} = [(\mathbf{v})_1^{\mathrm{T}}\ (\mathbf{v})_2^{\mathrm{T}}]^{\mathrm{T}}$, *then* $\text{-}\lambda$ *will be an eigenvalue of the same graph with the eigenvector* $\mathbf{v}' = [(\mathbf{v})_1^{\mathrm{T}}\ \text{-}(\mathbf{v})_2^{\mathrm{T}}]^{\mathrm{T}}$.

Proof. Set $M = 2$ in Theorem 6.9. Then $w = -1$. □

When the graph is bipartite, $\mathcal{L}$ and $\mathbf{A}$ have the same eigenvector structure even though they may have different eigenvectors. However, due to normalization by the degree matrix $\left(\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\right)$, symmetric eigenvalues of $\mathcal{L}$ add up to 2, whereas symmetric eigenvalues of $\mathbf{A}$ add up to 0, which agrees with the fact that trace of $\mathbf{A}$ is zero when it is M-block cyclic. Notice that Corollary 6.2 is valid for arbitrary bipartite graphs with complex edge values whereas Theorem 6.11 is constrained to undirected graphs with non-negative edge weights. From this comparison we can conclude that use of $\mathbf{A}$ as the unit shift operator rather than $\mathcal{L}$ allows more general class of bipartite graphs. Furthermore, Theorem 6.9 generalizes this property of 2-channel systems on arbitrary bipartite graphs to M-channel systems on M-block cyclic graphs.

Due to Theorem 6.8 and 6.9, we conclude that M-block cyclic graphs defined in (6.46) have all the necessary properties to generalize the classical multirate theory to the graph signals.

At this point it is interesting to notice the connection to circulant graphs discussed in [57, 56, 58]. Circulant graphs *do* satisfy the eigenvector condition in (6.57). This result follows from the fact that DFT matrix diagonalizes *any* circulant matrix. Further, with proper permutations (relabelling of the nodes), DFT matrix satisfies the condition in (6.57). An example of such a permutation will be demonstrated on the directed cyclic graph, which is a circulant graph, in the following paragraph. *This is very interesting because some of our theorems (Theorems 6.12, 6.18 and 6.19) that* only *require the eigenvector condition are now applicable to circulant graphs.* However, the eigenvalue condition of an M-block cyclic graph, (6.56), is not satisfied by the circulant graphs in general.

The connection to the classical cyclic graph (Section II-C of [153]) is also important to understand. For the classical cyclic shift matrix, the classical time domain decimator retains every M^{th} sample (rather than the first N/M samples). But our convention for graphs is that the first N/M samples are retained. To match with our convention, we permute the vertices (i.e., change the numbering convention). This converts the classical cyclic shift matrix into an M-block cyclic matrix. For example, suppose $N = 4$ and $M = 2$. The classical cyclic shift matrix, $\mathbf{C}_4$, is

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{6.61}$$

where rows and columns are numbered as $0, 1, 2$, and 3. The classical decimator retains samples 0 and 2 whereas our canonical decimator, by convention, retains 0 and 1. So we simply exchange columns 1 and 2 and also exchange rows 1 and 2. The resulting matrix is

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \tag{6.62}$$

which satisfies the requirements of Theorem 6.1, 6.2, and 6.3 (for $M = 2$). In fact the above matrix is a 2-block cyclic matrix. As stated in Fact 6.4, this permutation is possible for any (N, M) pair where M divides N. For a visual example with $N = 12$, please see Figure 6.9.

6.7 Identification of the M-Block Structure from a Noisy Permuted Version

Although many results from classical multirate theory can be extended to graphs with the help of M-Block cyclic graphs, the main limitation of these results is the restrictive nature of this condition. Unfortunately, most of the real world examples of graphs fail to satisfy this condition. The M-Block cyclic structure in (6.46) can be equivalently expressed in terms of the eigenvalue and eigenvector structure as given by Theorem 6.9. Therefore, M-Block cyclic structure can be considered as constraints on both the eigenvalues and the eigenvectors. Using the idea of similarity transform it is possible to remove the constraint on the eigenvectors (see Section 6.15). Nevertheless, the condition on the eigenvalues is still necessary. Therefore, the results overviewed here are not applicable to all graphs. In some cases a given graph might be *close* to being an M-Block cyclic graph, but not exactly so. In this case it is necessary to identify the hidden M-Block cyclic structure in the graph, which can be considered as an approximation of a graph with an M-Block cyclic graph.

Another subtle point with M-Block cyclic graphs is that Definition 6.5 implicitly assumes a proper labeling of the nodes such that the adjacency matrix can be written in the form of (6.46). However, the adjacency information of the graph might be provided *without* the required labeling of the nodes. In such a case the adjacency matrix cannot be written in the form of (6.46), hence the simple decimator and expander defined in (6.8) and (6.11) cannot be used directly.

Due to above mentioned points, this section considers an identification problem. Consider an unweighted M-Block cyclic graph where each of the nonzero blocks in $\mathbf{A}$ has the form $\mathbf{1}\mathbf{1}^{\mathrm{T}}$ (i.e., these blocks are complete subgraphs). This is demonstrated in Figure 6.11a. Instead of being given the exact adjacency matrix in Figure 6.11a (demonstrated for $M = 5$), imagine we are given a modified version with the following modifications:

1. Some of the existing edges are removed randomly and independently with probability q resulting in Figure 6.11b.

2. Some new edges are inserted randomly and independently with probability p resulting in Figure 6.11c.

3. The N nodes are randomly relabeled resulting in Figure 6.11d.

Given the adjacency matrix corresponding to Figure 6.11d, is it possible to recover the structure of the original M-Block cyclic graph? That is, invert the permutation that converted Figure 6.11c to Figure 6.11d? Once this is done, we can remove the randomly inserted edges to obtain Figure 6.11b, which is an M-Block cyclic graph with proper labeling of nodes. Of course we can never recover Figure 6.11a because of the randomly removed edges from the dark blocks. The main goal therefore is to identify the correct permutation of the labeling of the vertices so that we can go back from Figure 6.11d to Figure 6.11c.

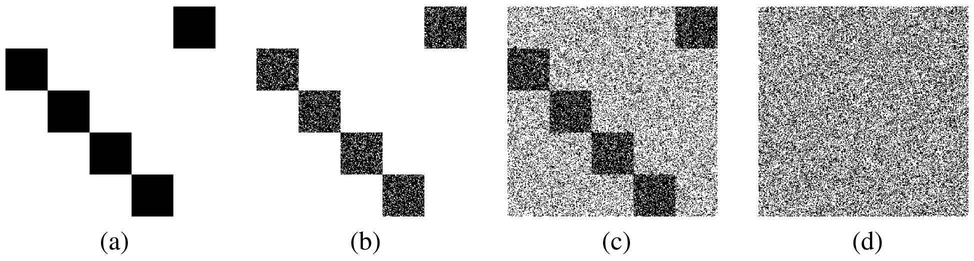

(a) (b) (c) (d)

Figure 6.11: (a) The adjacency matrix of a 5-Block cyclic graph of size 300, (b) existing edges are removed with probability $q = 0.2$, (c) noisy edges are added with probability $p = 0.3$, (d) random re-labeling of the nodes.

In the case of $q = 0$ and $p = 0$ (no missing edges and noisy edges) a simple search can solve the labeling problem: start from a node. Outgoing edges reveals a block. Select a node from the identified block and repeat this procedure M times. Since this simple technique implicitly assumes that the underlying graph is exactly M-Block cyclic, in the presence of missing and noisy edges such a procedure will fail. However, it is observed that the unique eigenvalue-eigenvector structure of M-Block cyclic matrices (given in Theorem 6.9) are robust to the missing and noisy edges, which can be used to identify the hidden blocks. In the following we will elaborate on this.

We will first consider the M-Block cyclic matrix whose adjacency matrix is visualized in Figure 6.11a. Since all the edges are assumed to have unit weights, $\mathbf{A}$ can be written in the following close form:

$$\mathbf{A} = \mathbf{C}_M \otimes (\mathbf{1}\mathbf{1}^{\mathrm{T}}), \tag{6.63}$$

where $\mathbf{C}_M$ is defined in (6.1) and $\mathbf{1} \in \mathbb{R}^{N/M}$ is the vector of all ones. Eigenvalues of the adjacency matrix $\mathbf{A}$ are then obtained as $\lambda(\mathbf{A}) = \lambda(\mathbf{C}_M)\ \lambda(\mathbf{1}\mathbf{1}^{\mathrm{T}})$. Therefore $\lambda(\mathbf{A}) = \{\frac{N}{M},\ \omega\frac{N}{M},\ \omega^2\frac{N}{M}, \cdots, \omega^{M\text{-}1}\frac{N}{M},\ 0, \cdots, 0\}$ where ω is given in (6.49). The adjacency matrix $\mathbf{A}$ has M non-zero eigenvalues equally-spaced on a circle with radius N/M. The remaining N-M eigenvalues are all zeros. These eigenvalues are visualized in Figure 6.12a for $M = 5$ and $N = 300$. When some of existing edges are removed, the adjacency matrix is no longer in the form of (6.63). Hence, the eigenvalues of the matrix in Figure 6.11b are different than Figure 6.12a. Since the removal of the edges does *not* disturb the M-Block cyclic property, all the eigenvalues still come as families of size M due to Theorem 6.9. In addition, M dominant eigenvalues continue to exist while the remaining ones are clustered around zero. Even though the magnitude of the dominant eigenvalues is a random variable (due to random removal of the edges), one can use Perron-Frobenius theorem for irreducible matrices[85] to argue that the magnitude can be well approximated by $(1\text{-}q)\,N/M$. The eigenvalues of the matrix in Figure 6.11b are visualized in Figure 6.12b. In this figure notice that q is selected to be $q = 0.2$.

In the presence of noisy edges M-Block cyclic structure no longer exist in the adjacency matrix. As a result, eigenvalues lose the family structure. However, it is interesting to observe that random addition of edges does *not* affect the structure of the dominant eigenvalues significantly. M dominant eigenvalues (except for the one on the real line) continue to exist equally spaced on a circle (approximately) while the remaining ones are clustered around zero. The eigenvalues of the matrix in Figure 6.11c are visualized in Figure 6.12c.

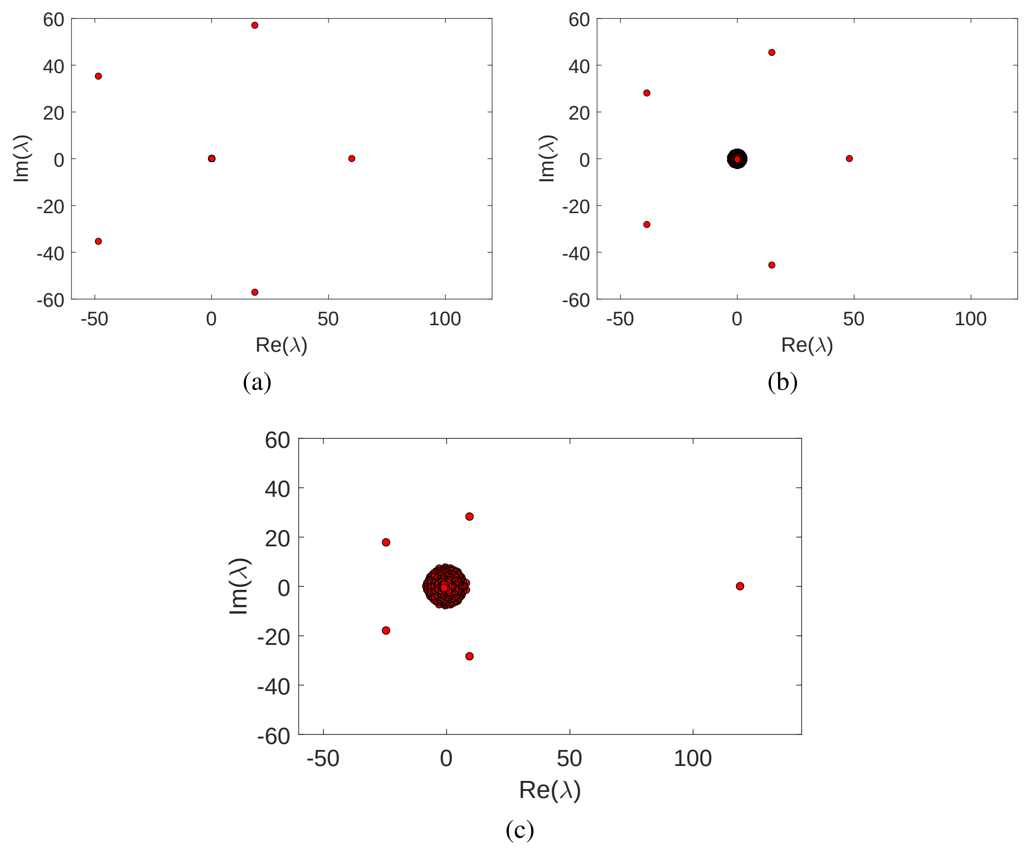


Figure 6.12: Eigenvalues of the matrices in (a) Figure 6.11a, (b) Figure 6.11b and (c) Figures 6.11c and 6.11d.

Due to stability of the eigenvalues one can hope to recover the underlying M-Block cyclic structure in the presence of noisy and missing edges. Based on this observation and inspired by the spectral graph coloring approach [5], we propose a 3-step routine to identify the blocks of the underlying graph, which is outlined in Algorithm 6.

Algorithm 6 Identification of the M-Block cyclic structure

```
1: Given the adjacency matrix $\mathbf{A}$, compute its eigenvalue decomposition as $\mathbf{A} = \mathbf{V}\,\boldsymbol{\Lambda}\,\mathbf{V}^{-1}$ where eigenvalues are assumed to be ordered in the *absolute* sense, that is, $|\lambda_1| \geq \cdots |\lambda_M| \geq \cdots |\lambda_N|$. Let $\{\mathbf{v}_1, \cdots, \mathbf{v}_M\}$ be the eigenvectors that correspond to the largest (in the absolute sense) $M$ eigenvalues.
2: Stack the eigenvectors into a matrix as $\mathbf{S} = [\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_M]$, and compute the following $\mathbf{C} = \mathbf{S}\,\mathbf{S}^{\mathrm{H}}$.
3: Cluster the rows of $\mathbf{C}$ into $M$ equal size partitions.
```

In this procedure the last step is not defined precisely. Various different techniques can be implemented. In our implementation we use a simple greedy technique to avoid the computational complexity: we start with the first row of the matrix $\mathbf{C}$. We find the indices of the elements of the vector with the largest N/M values and assign these indices to a block. Then, we consider the smallest un-assigned index and repeat this procedure M times. The rationale behind this greedy technique comes from the eigenvectors structure of the M-Block cyclic matrices (Theorem 6.9). In the case of $p = 0$, the matrix $\mathbf{S}$ has the following structure:

$$\mathbf{S} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_1\,\omega^{0} & \mathbf{u}_1\,\omega^{0} & \cdots & \mathbf{u}_1\,\omega^{0} \\ \mathbf{u}_2 & \mathbf{u}_2\,\omega^{-1} & \mathbf{u}_2\,\omega^{-2} & \cdots & \mathbf{u}_2\,\omega^{-(M-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_M & \mathbf{u}_M\,\omega^{-(M-1)} & \mathbf{u}_M\,\omega^{-2(M-1)} & \cdots & \mathbf{u}_M\,\omega^{-(M-1)(M-1)} \end{bmatrix}, \tag{6.64}$$

where $\mathbf{u}_i \in \mathbb{R}^{N/M}$ are the partitions of the eigenvector $\mathbf{v}_1$. As a result, the matrix $\mathbf{C}$ has the following block diagonal form:

$$\mathbf{C} = M \begin{bmatrix} \mathbf{u}_1\,\mathbf{u}_1^{\mathrm{H}} & & \\ & \ddots & \\ & & \mathbf{u}_M\,\mathbf{u}_M^{\mathrm{H}} \end{bmatrix}, \tag{6.65}$$

In the case of $p > 0$, the adjacency matrix is no longer M-Block cyclic. However, as suggested by Figure 6.12, the eigenvalue structure is not lost completely. Similarly, we have observed that the matrix $\mathbf{C}$ is "very close" to a block diagonal matrix. As a result, even a simple thresholding (finding the indices of the maximum values) is sufficient to identify the blocks of the hidden structure under a random re-labeling of the nodes.

When the proposed scheme is applied to the matrix in Figure 6.11d, it finds a permutation (re-labeling of the nodes) that results in the adjacency matrix given in Figure 6.13 from which it is apparent that the algorithm successfully reveals the hidden M-Block cyclic structure. When carefully observed, one can realize that the matrices in Figures 6.11d and 6.13 are not equal to each other. There are two reasons for this mismatch: 1) The main purpose of the algorithm is to identify the blocks. The labeling of the nodes *within a block* is arbitrary. 2) Once the blocks are identified, numbering of the blocks (from 1 to M) is arbitrary. Nevertheless, from the multirate processing perspective Figure 6.11c and Figure 6.13 are equivalent to each other. The noisy edges can be removed, and previously discussed multirate techniques can be applied accordingly.

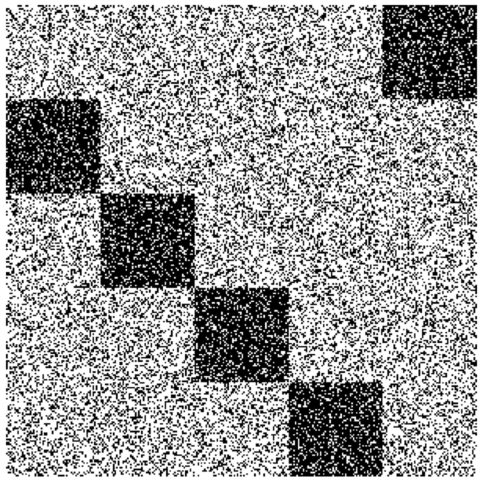


Figure 6.13: The algorithm proposed in Algorithm 6 finds a permutation that converts Figure 6.11d to the above form.

We have tested the proposed scheme (Algorithm 6) with various different values of p and q. In order to evaluate the recovery performance we consider the fraction of the nodes that are labeled correctly. In the worst case scenario (the adjacency matrix does not have an M-Block cyclic structure) the routine randomly assigns the nodes to the blocks. Since there are M blocks, the fraction can be at least $1/M$ on the average. The recovery performance of the proposed technique is provided in Figure 6.14. The number of blocks is selected to be $M = 5$, and the number of nodes are selected to be $N = 100, 300$ and 500 in Figures 6.14a-6.14c, respectively.

It is important to note the trade-off between the parameters p and q. The recovery performance of the proposed approach depends on the *density* of the edges in the regions. The average edge density between the consecutive blocks is 1-q (since edges are removed with probability q), and the density in the remaining parts is p. Figure 6.11c visually describes the difference between the densities for the case of $q = 0.2$ and $p = 0.3$. In fact this difference keeps the structure on the eigenvalues intact. When $1 - q \approx p$, the regions are no different from each other. Hence, the underlying blocks are not expected to be identified correctly. Figure 6.14 follows this expectation. Notice that the algorithm behaves poorly around the line $p + q = 1$. In the vicinity of this region the fraction of correctly labeled nodes is $1/M$, which corresponds to the random assignments of the nodes. Figures 6.14a-6.14c also suggest that larger number of nodes allows correct identification of the blocks for a larger set of values of p and q.

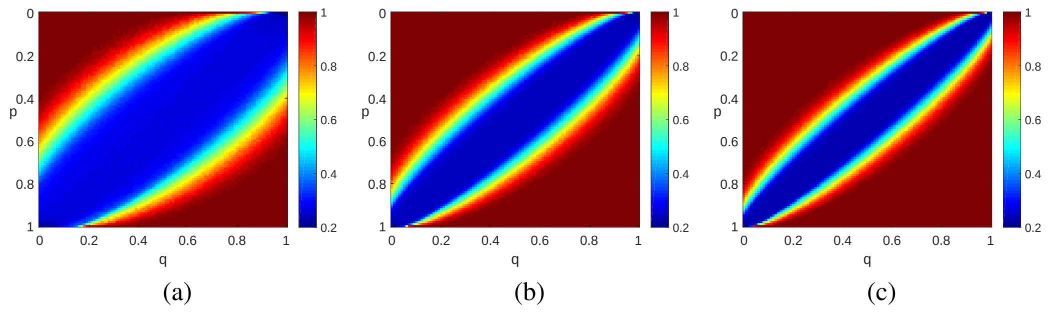


Figure 6.14: Recovery performance of the proposed technique for $M = 5$. The fraction of the nodes that are labeled correctly is color coded. Number of nodes are (a) $N = 100$, (b) $N = 300$, and (c) $N = 500$. Fractions are averaged over 350 experiments.

As a final note, in the proposed algorithm the value of M is assumed to be known, which may not be available in some cases. However, as Figure 6.12 suggests, even a visual analysis of the eigenvalues is enough to identify the value of M. If we were given only the Figure 6.12c, then it would be straightforward to conclude that the eigenvalues belong to a 5-Block cyclic adjacency matrix. Hence, we can conclude that knowledge about M is a very mild assumption.

6.8 Concept of Spectrum Folding and Aliasing

In order to talk about alias-free and perfect reconstruction graph filter banks, we need to first define what aliasing is in graph signals. For this purpose we now revisit the downsample-then-upsample (DU) operation. According to our canonical definition of decimator in (6.8), DU operator is given in (6.12). Since DU replaces samples with zeros, it is a lossy operation and the erased samples cannot be reconstructed back from the remaining data in general. We now analyze the effect of the DU operator from the frequency domain viewpoint, and explain the *spectrum folding* or *aliasing* effect. A similar approach is presented for two-channel systems in [129], where graph signal processing is based on the graph Laplacian. In our development the graph $\mathbf{A}$ is allowed to have complex edge weights and can be directed.

Using the canonical definition of the decimator in (6.8) and eigenvector-shift operator $\mathbf{\Omega}$ in (6.50), the DU operator can be written as a sum of powers of $\mathbf{\Omega}$. That is,

$$\mathbf{D}^{\mathrm{T}}\,\mathbf{D} = \frac{1}{M}\sum_{k=0}^{M\text{-}1}\mathbf{\Omega}^{k}. \tag{6.66}$$

Now consider the DU version of a graph signal $\mathbf{x}$, namely

$$\mathbf{y} = \mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{x}. \tag{6.67}$$

Remember that graph Fourier transform of a graph signal $\mathbf{x}$ is given in Definition 6.2. Let $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{y}}$ denote the graph Fourier transform of the input and output signal of the DU system. Let $\mathbf{G}$ denote the frequency domain operation of the DU operator. That is, $\widehat{\mathbf{y}} = \mathbf{G}\,\widehat{\mathbf{x}}$. Due to Definition 6.3 we have

$$\mathbf{G} = \mathbf{V}^{-1}\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{V}. \tag{6.68}$$

Using (6.66), we can write $\mathbf{G}$ as follows:

$$\mathbf{G} = \frac{1}{M}\sum_{k=0}^{M-1}\mathbf{V}^{-1}\boldsymbol{\Omega}^{k}\,\mathbf{V}. \tag{6.69}$$

In the following, we will not constrain ourselves to M-block cyclic graphs and assume that $\mathbf{A}$ is diagonalizable and only the eigenvectors of $\mathbf{A}$ satisfy (6.57) and let eigenvalues be arbitrary. In Section 6.15 we will discuss how this assumption on the eigenvectors can be removed by appropriately generalizing the definition of the decimator $\mathbf{D}$.

Now notice that $\boldsymbol{\Omega}$ is the eigenvector-shift operator for the eigenvectors satisfying (6.57). Therefore, $\boldsymbol{\Omega}^{k}\,\mathbf{V}$ will be the column permuted version of $\mathbf{V}$. Due to (6.57), $\boldsymbol{\Omega}^{k}$ will circularly shift each vector of an eigen-family to the left by k times. Due to our ordering convention on the eigenvectors in (6.60), we have the following:

$$\boldsymbol{\Omega}^{k}\mathbf{V} = \Big[\mathbf{v}_{1,1+k}\cdots\mathbf{v}_{1,M+k}\;\cdots\;\mathbf{v}_{N/M,1+k}\cdots\mathbf{v}_{N/M,M+k}\Big]. \tag{6.70}$$

Notice that this permutation of the columns of $\mathbf{V}$ can also be written with a column permutation matrix. Therefore we have

$$\boldsymbol{\Omega}^{k}\;\mathbf{V} = \mathbf{V}\;\boldsymbol{\Pi}_{k}, \tag{6.71}$$

where

$$\boldsymbol{\Pi}_{k} = \mathbf{I}_{N/M}\;\otimes\;\mathbf{C}_{M}^{k} = \begin{bmatrix} \mathbf{C}_{M}^{k} & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{C}_{M}^{k} \end{bmatrix}, \tag{6.72}$$

where $\mathbf{C}_{M}$ is the size M cyclic matrix defined in (6.1). Using (6.71) and (6.72), the frequency domain operation $\mathbf{G}$ in (6.68) can be written as

$$\mathbf{G} = \mathbf{I}_{N/M}\;\otimes\;\frac{1}{M}\sum_{k=0}^{M-1}\mathbf{C}_{M}^{k}. \tag{6.73}$$

Since the first M powers of cyclic matrix of size M add up to matrix with all 1 entries, this response further simplifies to

$$\mathbf{G} = \mathbf{I}_{N/M} \otimes \frac{1}{M} \mathbf{1}_M \mathbf{1}_M^{\mathrm{H}}, \tag{6.74}$$

where $\mathbf{1}_M$ denotes the column vector of size M with all 1 entries.

To be consistent with double indexing of the eigenvectors, we will stick to that scheme for the frequency components of a graph signal. That is to say,

$$\widehat{\mathbf{x}} = [\widehat{x}_{1,1} \cdots \widehat{x}_{1,M} \; \cdots \; \widehat{x}_{N/M,1} \cdots \widehat{x}_{N/M,M}]^{\mathrm{T}}. \tag{6.75}$$

Due to (6.74), we have the following relation between the graph Fourier transform of the original signal and the graph Fourier transform of the downsampled-then-upsampled signal

$$\widehat{y}_{i,1} = \widehat{y}_{i,2} = \cdots = \widehat{y}_{i,M} = \frac{1}{M} \sum_{j=1}^{M} \widehat{x}_{i,j}, \tag{6.76}$$

for all $1 \leq i \leq N/M$. We state this result in the following theorem.

Theorem 6.12 (Spectrum folding in graph signals)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of a graph. Assume that* $\mathbf{A}$ *is diagonalizable and has the eigenvector structure in* (6.57) *as indexed in* (6.60) *with arbitrary eigenvalues. Let* $\mathbf{x}$ *be a signal on the graph and* $\mathbf{y} = \mathbf{D}^{\mathrm{T}} \mathbf{D}\mathbf{x}$ *where* $\mathbf{D}$ *is as in* (6.8)*. Then, the graph Fourier transforms of* $\mathbf{x}$ *and* $\mathbf{y}$ *are related as:*

$$\widehat{\mathbf{y}} = \frac{1}{M} \left(\mathbf{I}_{N/M} \otimes \mathbf{1}_M \mathbf{1}_M^{\mathrm{T}} \right) \widehat{\mathbf{x}}. \tag{6.77}$$

Thus the DU operation results in the phenomenon described by (6.76) in the frequency domain. This is similar to *aliasing* or *spectral folding* because multiple frequency components of the input overlap into the same frequency component of the output. This is similar to the effect of decimation in classical signal processing [200]. From the folded spectrum (6.76) we cannot in general recover the original signal, which is consistent with the fact that decimation is in general a information-lossy operation. It should be remembered however that the expression (6.76) has been derived only for graphs $\mathbf{A}$ for which the eigenvectors have the restricted structure (6.57).

6.9 Linear Systems on Graphs: Interconnection Between Shift Invariance, Alias-Free Property, and Polynomial Property

The above notion of aliasing or spectrum folding due to the DU operator on a graph can be generalized. Thus consider *any* system $\mathcal{S}$ defined on a diagonalizable graph $\mathbf{A}$, producing output $\mathbf{y} = \mathcal{S}(\mathbf{x})$ in response to an input $\mathbf{x}$. Let $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{y}}$ denote the graph Fourier transforms of $\mathbf{x}$ and $\mathbf{y}$. We say that the *system $\mathcal{S}$ is alias-free* if each component of $\widehat{\mathbf{y}}$ is determined by the corresponding component of $\widehat{\mathbf{x}}$, i.e., $\widehat{y}_i = g_i(\widehat{x}_i)$. In other words, there is no interference between Fourier components. For the special case of *linear systems* on the graph $\mathbf{A}$, this reduces to $\widehat{y}_i = \alpha_i \widehat{x}_i$, where α_i is analogous to frequency response.

In classical signal processing, it is well known that linear shift invariant systems are automatically alias-free. For the case of graph signals this equivalence is not always true as we shall elaborate. It was proved in [152] that shift invariance of a linear system on a graph $\mathbf{A}$ is equivalent to the statement that the system $\mathbf{H}$ be a polynomial (under some conditions, see Theorem 6.13 below). In this section we will see that the shift invariance, alias-free property, and polynomial property do not imply each other in general. Their inter relationship depends on whether the graph $\mathbf{A}$ has distinct eigenvalues or not. These results are elaborated in Theorems 6.14 and 6.15, which we shall prove in this section. For clarity we begin with the following formal definitions.

Definition 6.6 (Shift-invariant filters [152])**.** *Let* $\mathbf{A}$ *be the adjacency matrix of a graph. Let* $\mathbf{H}$ *be a linear system on the graph. It is said that* $\mathbf{H}$ *is* shift-invariant *if it commutes with* $\mathbf{A}$*, that is,* $\mathbf{AH} = \mathbf{HA}$.

Definition 6.7 (Alias-free filters)**.** *Let the graph be such that* $\mathbf{A}$ *is diagonalizable, i.e.,* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ *for some diagonal* $\mathbf{\Lambda}$ *and invertible* $\mathbf{V}$*. Let* $\mathbf{H}$ *be a linear system on* $\mathbf{A}$ *with frequency domain operation* $\widehat{\mathbf{H}} = \mathbf{V}^{-1}\mathbf{H}\mathbf{V}$*. We say* $\mathbf{H}$ *is a alias-free filter on graph* $\mathbf{A}$ *if* $\widehat{\mathbf{H}}$ *is a diagonal matrix. In this case* $\widehat{\mathbf{H}}$ *is called the* frequency response *of the filter* $\mathbf{H}$.

A polynomial filter is always shift invariant because

$$\mathbf{HA} = \left(\sum_{k=0}^{N-1} \alpha_k \mathbf{A}^k\right)\mathbf{A} = \mathbf{A}\left(\sum_{k=0}^{N-1} \alpha_k \mathbf{A}^k\right) = \mathbf{AH}. \tag{6.78}$$

But in general the converse is not true. The following result was proved in [152]:

Theorem 6.13 (Polynomial and shift-invariant graph filters, Theorem 1 in [152])**.** *Let* $\mathbf{A}$ *be the graph adjacency matrix and assume that its characteristic and minimal polynomials are equal. Then a graph filter* $\mathbf{H}$ *is linear and shift-invariant if and only if* $\mathbf{H}$ *is a polynomial on the graph shift* $\mathbf{A}$*.*

We now state and prove the following results.

Theorem 6.14 (Linear systems on diagonalizable graphs)**.** *Let* $\mathbf{H}$ *be a linear system on the graph* $\mathbf{A}$*. Assume* $\mathbf{A}$ *is diagonalizable. Then the following are true:*

1. *If* $\mathbf{H}$ *is a polynomial in* $\mathbf{A}$ *then it is alias free.*
2. *If* $\mathbf{H}$ *is alias-free then it is shift invariant.*

Proof. 1) Let $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ be the eigenvalue decomposition of $\mathbf{A}$. Since $\mathbf{H}$ is polynomial in $\mathbf{A}$ we have $\mathbf{H} = H(\mathbf{A})$ where $H(\cdot)$ is a polynomial. Then $\mathbf{H} = \mathbf{V}H(\mathbf{\Lambda})\mathbf{V}^{-1}$, where $\mathbf{\Lambda}$ is the diagonal matrix consisting of the eigenvalues of $\mathbf{A}$. Notice that $H(\mathbf{\Lambda}) = \mathbf{V}^{-1}\mathbf{H}\mathbf{V}$ is the frequency domain operation of the system, which is a polynomial of a diagonal matrix. Therefore the overall frequency domain operation is a diagonal matrix, hence it is alias-free.

2) Let $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ be the eigenvalue decomposition of $\mathbf{A}$. Assume that $\mathbf{H}$ is alias-free. Then it can be written as $\mathbf{H} = \mathbf{V}\mathbf{Z}\mathbf{V}^{-1}$ for a diagonal $\mathbf{Z}$ due to Definition 6.7. Then, we have $\mathbf{H}\mathbf{A} = \mathbf{V}\mathbf{Z}\mathbf{\Lambda}\mathbf{V}^{-1} = \mathbf{V}\mathbf{\Lambda}\mathbf{Z}\mathbf{V}^{-1} = \mathbf{A}\mathbf{H}$ since diagonal matrices commute. Hence, $\mathbf{H}$ is shift-invariant. □

Theorem 6.15 (Linear systems on graphs with distinct eigenvalues)**.** *Let* $\mathbf{H}$ *be a linear system on the graph* $\mathbf{A}$*. Assume* $\mathbf{A}$ *has distinct eigenvalues (so that it is, in particular, diagonalizable). Then the following statements are equivalent:*

1. $\mathbf{H}$ *is a polynomial in* $\mathbf{A}$*.*
2. $\mathbf{H}$ *is alias-free.*
3. $\mathbf{H}$ *is shift invariant.*

Proof. Since $\mathbf{A}$ is diagonalizable, it follows from Theorem 6.14 that (1) implies (2) and (2) implies (3).

We now prove that (3) implies (2): Assume $\mathbf{H}$ is shift invariant, that is, $\mathbf{AH} = \mathbf{HA}$. Since $\mathbf{A}$ has distinct eigenvalues, this implies the following: $\mathbf{H}$ is also diagonalizable; $\mathbf{A}$ and $\mathbf{H}$ are *simultaneously diagonalizable*. (These two claims follows from Problem 13 on page 56 of [85]). But since $\mathbf{A}$ has distinct eigenvalues, $\mathbf{V}$ is its only diagonalizing matrix (up to a permutation and scaling of columns). So, $\mathbf{V}$ in particular, diagonalizes $\mathbf{H}$, which (by Definition 6.7) shows that $\mathbf{H}$ is alias-free.

We finally prove that (2) implies (1): Assume $\mathbf{H}$ is alias-free, that is, $\mathbf{V}^{-1}\mathbf{HV} = \mathbf{Z}$ is a diagonal matrix with N diagonal elements z_i. Since the eigenvalues λ_i, $1 \leq i \leq N$ of $\mathbf{A}$ are distinct, we can always find a set of N numbers h_i such that the following holds:

$$\begin{bmatrix} 1 & \lambda_1 & \cdots & \lambda_1^{N-1} \\ 1 & \lambda_2 & \cdots & \lambda_2^{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \lambda_N & \cdots & \lambda_N^{N-1} \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{N-1} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix}. \tag{6.79}$$

This is because the matrix on the left, being Vandermonde with distinct λ_i, is invertible. Thus, there exists a polynomial $H(\lambda) = \sum_{k=0}^{N-1} h_k \, \lambda^k$ such that $H(\lambda_i) = z_i$. In matrix notation we can rewrite this as $H(\mathbf{\Lambda}) = \mathbf{Z}$, or

$$\sum_{k=0}^{N-1} h_k \, \mathbf{\Lambda}^k = \mathbf{Z}, \quad \text{i.e.,} \quad \mathbf{V} \sum_{k=0}^{N-1} h_k \, \mathbf{\Lambda}^k \, \mathbf{V}^{-1} = \mathbf{H}, \tag{6.80}$$

or equivalently $\sum_{k=0}^{N-1} h_k \, \mathbf{A}^k = \mathbf{H}$, which proves that $\mathbf{H}$ is a polynomial in $\mathbf{A}$. □

According to Definition 6.1 and 6.6, we can talk about polynomial and shift-invariant filters on a graph with an arbitrary adjacency matrix. However, *the definition of alias-free filters is exclusive to graphs with diagonalizable adjacency matrices*. We intentionally exclude the graphs with non-diagonalizable adjacency matrices due to following reasons. In [153], authors use total variation to quantify the notion of frequency in the graph signals. When the adjacency matrix is not diagonalizable, total variation of a generalized eigenvector inherently depends on the next generalized eigenvector in the Jordan chain that makes it difficult to interpret. Furthermore, when the adjacency matrix is not diagonalizable, even the unit shift element, $\mathbf{A}$, has a non-diagonal frequency domain operation. Hence, relation between the polynomial filtering and aliasing in the case of non-diagonalizable adjacency matrices is out of the scope of this work and deserves an independent study.

In the following we will provide three examples to demonstrate the necessity of distinct eigenvalues for the equivalence of the above mentioned three properties. Let $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\text{-1}}$ be the eigenvalue decomposition of the graph as in (6.5).

1. Let $\mathbf{H}$ be an alias-free filter: $\mathbf{H} = \mathbf{V}\mathbf{Z}\mathbf{V}^{\text{-1}}$ where $\mathbf{Z}$ is a diagonal matrix with *distinct* diagonal entries z_i such that $z_i \neq z_j$ for $i \neq j$. Let λ be a repeated eigenvalue of $\mathbf{A}$ with algebraic multiplicity 2. In order to represent $\mathbf{H}$ as a polynomial, we need to find a polynomial $H(\cdot)$ such that $H(\lambda) = z_i$ and $H(\lambda) = z_j$ for some $i \neq j$. Since z_i's are distinct, such a function does not exist. Hence, $\mathbf{H}$ is alias-free but not polynomial in $\mathbf{A}$.

2. Let λ be an eigenvalue of $\mathbf{A}$ with algebraic multiplicity m [85]. Assume $m > 1$. Hence, $\mathbf{A}$ has repeated eigenvalues. Then we can write $\mathbf{\Lambda}$ as follows (by ordering the eigenvectors):

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda\mathbf{I}_m & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}' \end{bmatrix}, \qquad \mathbf{Z} = \begin{bmatrix} \mathbf{Z}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}_2 \end{bmatrix}. \tag{6.81}$$

   Let $\mathbf{H}$ be such that $\mathbf{H} = \mathbf{V}\mathbf{Z}\mathbf{V}^{\text{-1}}$ with $\mathbf{Z}$ is as in (6.81) where $\mathbf{Z}_2$ is a diagonal but $\mathbf{Z}_1$ is a non-diagonalizable square matrix of size m. Notice that $\mathbf{\Lambda}$ and $\mathbf{Z}$ commute. Hence, $\mathbf{A}$ and $\mathbf{H}$ commute, that is, $\mathbf{H}$ is shift invariant on the graph. But $\mathbf{Z}$, which is the frequency domain operation of $\mathbf{H}$, is not diagonal since $m > 1$ and $\mathbf{Z}_1$ is non-diagonalizable. As a result, $\mathbf{H}$ is shift-invariant but not alias-free.

3. Consider the construction in the previous example (6.81). Since $\mathbf{\Lambda}$ is a diagonal matrix, any polynomial of $\mathbf{\Lambda}$ will be diagonal. That is, no polynomial of $\mathbf{\Lambda}$ is equal to non-diagonal $\mathbf{Z}$. Hence, $\mathbf{H}$ is shift-invariant but not polynomial.

Notice that Theorem 6.13 applies to any graph whether its adjacency matrix is diagonalizable or not. In the case of diagonalizable matrices, the minimal polynomial is equal to characteristic polynomial if and only if the matrix has distinct eigenvalues [85]. Figure 6.15 schematically shows the relation between shift-invariance, alias-free property, and polynomial property of a linear system on a graph.

When the adjacency matrix is the directed cycle $\mathbf{C}_N$, graph signal processing reduces to the classical theory [152]. Since $\mathbf{C}_N$ has distinct eigenvalues in the form of $e^{-j2\pi k/N}$ for $0 \leq k \leq N\text{-}1$, polynomial, alias-free and time-invariant filters are

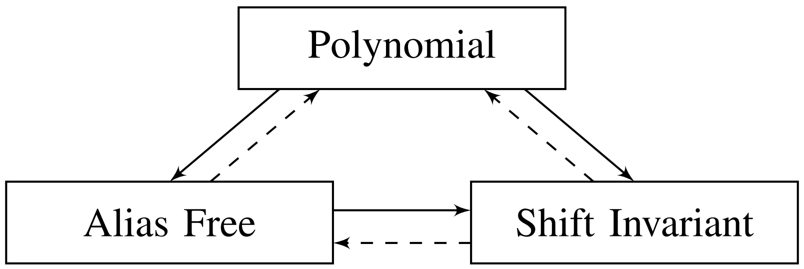


Figure 6.15: Relations between the alias-free, shift invariant and polynomial graph filters. Implications shown with solid lines exist for diagonalizable adjacency matrices whereas broken lines further require all eigenvalues to be distinct. In fact, polynomial filters imply shift invariance even if the adjacency matrix is not diagonalizable.

equivalent to each other. Therefore, relations in Figure 6.15 are consistent with classical signal processing theory but show that our understanding of these properties do not extend to graph case trivially.

6.10 Graph Filter Banks

In previous sections, we found that multirate building blocks defined on graphs satisfy identities similar to classical multirate identities only under certain conditions on the adjacency matrix $\mathbf{A}$. In the case of filter banks, even the simplest maximally decimated filter bank (the lazy filter bank of Figure 6.3) may or may not satisfy perfect reconstruction (unlike in the classical case). These were elaborated in Section 6.2.

In this section we consider filter banks in greater detail. Figure 6.16 shows a maximally decimated graph filter bank where the analysis filters $H_k(\mathbf{A})$ and the synthesis filters $F_k(\mathbf{A})$ are polynomials in $\mathbf{A}$, and $\mathbf{D}$ is as in (6.8). In the classical case, a maximally decimated filter bank is known to be a periodically time-varying system unless aliasing is completely canceled, in which case it becomes a time-invariant system. It is possible to get a somewhat analogous property for filter banks on graphs, but only under some conditions. In this section we first develop these results. We then study a class of filter banks that are analogous to ideal brickwall filters (bandlimited filter banks in the classical case) and show that perfect reconstruction can be achieved under some constraints on the eigen-structure of the graph $\mathbf{A}$.

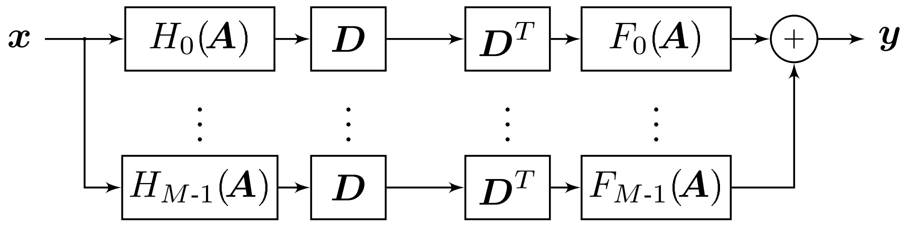


Figure 6.16: An M-channel maximally decimated filter bank on a graph with adjacency matrix $\mathbf{A}$. Here $H_k(\mathbf{A})$ and $F_k(\mathbf{A})$ are polynomials in $\mathbf{A}$ (so they are linear shift-invariant systems). The decimation matrix $\mathbf{D}$ is as in (6.8) with decimation ratio M. Overall response of the filter bank is denoted as $T(\mathbf{A})$, that is, $\mathbf{y} = T(\mathbf{A})\,\mathbf{x}$.

6.10.1 Graph Filter Banks as Periodically Shift-Varying Systems

In classical filter banks where the filters are polynomials in the shift operator z^{-1}, the maximally decimated analysis/synthesis system is a linear periodically shift-variant (LPSV(M)) system (i.e., the system from $\mathbf{x}$ to $\mathbf{y}$ in Figure 6.16 is LPSV). This is always true regardless of what the filter coefficients are. In particular, if the filter coefficients are such that this LPSV(M) system reduces to an LTI system, then the system can be shown to be alias-free (and vice versa) [200]. Furthermore if this LTI system is a pure delay $c\,z^{-n_0}$ then the system has the perfect reconstruction (PR) property. Since our main goal is to get insights into a parallel theory for graph filter banks, we now discuss the role of the periodically shift-varying property for graph filter banks with polynomial filters.

For the filter bank in Figure 6.16, let $T_k(\mathbf{A})$ denote the response of the k^{th} channel. Therefore we have,

$$T_k(\mathbf{A}) = F_k(\mathbf{A})\ \mathbf{D}^{\mathrm{T}}\ \mathbf{D}\ H_k(\mathbf{A}), \tag{6.82}$$

and the overall response from $\mathbf{x}$ to $\mathbf{y}$ is

$$T(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} T_k(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} F_k(\mathbf{A})\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,H_k(\mathbf{A}). \tag{6.83}$$

Notice that $T_k(\mathbf{A})$ is a linear operator. However, the DU operator, $\mathbf{D}^{\mathrm{T}}\mathbf{D}$, does not commute with $\mathbf{A}$ in general, hence response of the k^{th} channel is shift-varying. That is to say,

$$T_k(\mathbf{A})\ \mathbf{A} \neq \mathbf{A}\ T_k(\mathbf{A}), \tag{6.84}$$

for arbitrary analysis and synthesis filters. Therefore we have proved:

Theorem 6.16 (Graph filter banks are shift-varying)**.** *An arbitrary maximally decimated M-channel filter bank on an arbitrary graph is in general a linear but* *shift-varying system (i.e., the mapping from* $\mathbf{x}$ *to* $\mathbf{y}$ *in Figure 6.16 is linear and shift-varying).*

This is similar to classical multirate theory where an arbitrary filter bank is a time-varying system [200]. In fact, filter banks in the classical theory are periodically time-varying systems with period M. This leads to the question: is the filter bank on a graph periodically shift-varying?

In classical theory, when a system relates the input $x(n)$ to the output $y(n)$ in the following way $y(n) = \sum_k \; a_n(k)\; x(n\text{-}k)$, a linear and periodically time-varying system with period M is defined by the defining equation $a_{n+M}(k) = a_n(k)$ in [200]. It can be shown that a linear system satisfies this relation if and only if the following is true: if $x(n)$ produces output $y(n)$, then $x(n{+}M)$ produces output $y(n{+}M)$. Motivated by this, we present the following definition.

Definition 6.8 (Periodically shift-varying system)**.** *A linear system* $\mathbf{H}$ *on a graph* $\mathbf{A}$ *is said to be periodically shift-varying with period M, LPSV(M), if* $\mathbf{A}^M\,\mathbf{H} = \mathbf{H}\,\mathbf{A}^M$. *This reduces to shift-invariance when* $M = 1$.

Using this definition, we state the following result for the periodically shift-varying response of a maximally decimated M-channel filter bank on graphs.

Theorem 6.17 (Periodically shift-varying filter banks)**.** *The graph FB in Figure 6.16 is LPSV(M) for all choices of the polynomial filters* $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$*, if and only if the adjacency matrix of the graph satisfies the noble identity condition in* (6.27).

Proof. The LPSV(M) property, by Definition 6.8, is equivalent to

$$\sum_{k=0}^{M\text{-}1} T_k(\mathbf{A})\;\mathbf{A}^M = \mathbf{A}^M \sum_{k=0}^{M\text{-}1} T_k(\mathbf{A}). \tag{6.85}$$

Since $T_k(\mathbf{A})$ is as in (6.82), this is true for all polynomial filters $H_k(\mathbf{A})$ and $F_k(\mathbf{A})$ if and only if

$$\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{A}^M = \mathbf{A}^M\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}. \tag{6.86}$$

Partition $\mathbf{A}^M$ as in (6.17), and substitute into (6.86). Then the result is $(\mathbf{A}^M)_{1,2} = \mathbf{0}$ and $(\mathbf{A}^M)_{2,1} = \mathbf{0}$, which is the same as the condition (6.27). Conversely, if (6.27) holds it is obvious that (6.86) holds (because $\mathbf{D}^{\mathrm{T}}\,\mathbf{D}$ is as in (6.12)), so the LPSV(M) property (6.85) follows. □

6.11 Graph Filter Banks on $\mathbf{\Omega}$-Graphs

In this section we will construct maximally decimated M-channel filter banks with perfect reconstruction (PR) property on $\mathbf{\Omega}$-graphs. These filter banks are ideal in the sense that each channel has a particular sub-band and there is no overlap between different channels. The key point here is that this construction uses analysis and synthesis filters that are *both* alias-free (diagonal matrices in the frequency domain according to Definition 6.7). If the filters do not have to satisfy this restriction, then the problem of constructing PR filter banks becomes rather trivial. (See Section 6.3.) In fact, one can find low order polynomial filters by first designing unconstrained filters (or, polynomial filters with high degree as in [129, 76]) with PR property, then computing their low order polynomial approximations. This approach results in a filter bank with approximate PR property, with a trade-off between the approximation error and the degree of the filters. However, our approach here is to directly design alias-free filters. Later in Section 6.12 we will study how we can design low order polynomial filters directly.

Remember that $\mathbf{\Omega}$-graphs do not have any constraints on eigenvalues, however, the eigenvectors of $\mathbf{\Omega}$-graphs satisfy the $\mathbf{\Omega}$-structure given in (6.57). In Section 6.15, we will show how this constraint on eigenvectors can be removed by generalizing the definition of the decimator $\mathbf{D}$.

Remember from (6.76) that DU operation, $\mathbf{D}^{\mathrm{T}}\,\mathbf{D}$, results in aliasing for an arbitrary graph signal. Nonetheless, we can still recover the input signal from the output of $\mathbf{D}^{\mathrm{T}}\,\mathbf{D}$ if the input signal has zeros in its graph Fourier transform. To discuss this further, we define *band-limited* signals on graphs as follows:

Definition 6.9 (Band-limited graph signals on $\mathbf{\Omega}$-graphs). *Let* $\mathbf{A}$ *be the adjacency matrix of an* $\mathbf{\Omega}$*-graph with the following eigenvalue decomposition* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$. *A signal* $\mathbf{x}$ *on this* $\mathbf{\Omega}$*-graph said to be* k^{th}*-band-limited when its graph Fourier transform,* $\widehat{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{x}$*, has zeros in the following way:*

$$\widehat{x}_{i,j} = 0, \qquad 1 \le j \le M, \quad j \ne k, \quad 1 \le i \le N/M, \tag{6.87}$$

where we used the double indexing scheme similar to (6.59) *and* (6.60) *to denote the graph Fourier coefficients* $\widehat{x}_{i,j}$.

In the literature, there are different notions and definitions for band-limited graph signals [35, 34, 125, 4, 66]. Under an appropriate re-indexing of the eigenvalues (and the eigenvectors), our notion of k^{th}-band-limited signal is similar to the one in [35] with bandwidth N/M. This is consistent with our purpose of constructing M-channel graph filter banks.

When a graph signal is k^{th}-band-limited according to Definition 6.9, due to (6.76), the output spectrum of DU operation becomes as follows:

$$\widehat{y}_{i,1} = \widehat{y}_{i,2} = \cdots = \widehat{y}_{i,M} = \frac{1}{M}\,\widehat{x}_{i,k}. \tag{6.88}$$

In this case we can recover the input signal from the output of the DU operator since only one of the aliasing frequency components is non-zero. For this purpose, let $\mathbf{F}$ be a linear filter with frequency response $\mathbf{V}^{-1}\mathbf{F}\mathbf{V} = \mathbf{I}_{N/M} \otimes M\,\mathbf{e}_k\,\mathbf{e}_k^{\mathrm{T}}$, where $\mathbf{e}_k$ is the k^{th} element of the standard basis for $\mathbb{C}^M$. Then, consider the following system:

$$\mathbf{z} = \mathbf{F}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{x}. \tag{6.89}$$

Due to (6.74) and the construction of $\mathbf{F}$ above, in the graph Fourier domain (6.89) translates to the following:

$$\widehat{\mathbf{z}} = \left(\mathbf{I}_{N/M} \otimes M\,\mathbf{e}_k\,\mathbf{e}_k^{\mathrm{T}}\right)\left(\mathbf{I}_{N/M} \otimes \frac{1}{M}\,\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\widehat{\mathbf{x}} = \left(\mathbf{I}_{N/M} \otimes \mathbf{e}_k\,\mathbf{1}^{\mathrm{T}}\right)\widehat{\mathbf{x}}. \tag{6.90}$$

Since $\widehat{\mathbf{x}}$ is assumed to be k^{th}-band-limited according to Definition 6.9, we get $\widehat{\mathbf{z}} = \widehat{\mathbf{x}}$. That is, we can reconstruct the original signal from its decimated version using the linear reconstruction filter $\mathbf{F}$.

Notice that the frequency response of $\mathbf{F}$ is a diagonal matrix by its definition. Therefore $\mathbf{F}$ is an alias-free filter due to Definition 6.7. Furthermore, when the graph is assumed to have distinct eigenvalues, $\mathbf{F}$ can be realized as a polynomial filter due to Theorem 6.15. We have therefore proved the following theorem.

Theorem 6.18 (Polynomial interpolation filters for $\mathbf{\Omega}$-graphs)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of an* $\mathbf{\Omega}$*-graph. Let* $\mathbf{x}$ *be a* k^{th}*-band-limited signal on the graph. Then, there exists an interpolation filter* $\mathbf{F}$ *that recovers* $\mathbf{x}$ *from* $\mathbf{D}\mathbf{x}$*. When the eigenvalues are distinct,* $\mathbf{F}$ *can be a polynomial in* $\mathbf{A}$*, that is,* $\mathbf{F} = F(\mathbf{A})$*.*

The recovery of missing samples in graph signals is also discussed in various settings [207, 35, 113].

In the following, we will discuss how we can write an arbitrary full-band signal as a sum of k^{th}-band-limited signals. For this purpose, consider the following identity,

$$\mathbf{I}_N = \sum_{k=1}^{M} \mathbf{I}_{N/M} \otimes \left(\mathbf{e}_k\,\mathbf{e}_k^{\mathrm{T}}\right). \tag{6.91}$$

Then we can write $\widehat{\mathbf{x}} = \sum_{k=1}^{M} \widehat{\mathbf{x}}_k$, where

$$\widehat{\mathbf{x}}_k = \left(\mathbf{I}_{N/M} \otimes \mathbf{e}_k\, \mathbf{e}_k^{\mathrm{T}}\right) \widehat{\mathbf{x}}. \tag{6.92}$$

With the construction in (6.92), $\widehat{\mathbf{x}}_k$ is a k^{th}-band-limited signal. Notice that (6.92) is a frequency domain relation. In the graph signal domain, consider a linear filter $\mathbf{H}_{k\text{-}1}$ with frequency response

$$\widehat{\mathbf{H}}_{k\text{-}1} = \mathbf{V}^{-1}\, \mathbf{H}_{k\text{-}1}\, \mathbf{V} = \mathbf{I}_{N/M} \otimes \left(\mathbf{e}_k\, \mathbf{e}_k^{\mathrm{T}}\right). \tag{6.93}$$

Then, we have $\mathbf{x} = \sum_{k=1}^{M} \mathbf{x}_k$, where $\mathbf{x}_k = \mathbf{H}_{k\text{-}1}\, \mathbf{x}$. Since $\mathbf{x}_k$ is a k^{th}-band-limited graph signal, we can reconstruct it from its decimated version using the interpolation filter discussed in Theorem 6.18. For this purpose, let $\mathbf{F}_{k\text{-}1}$ be a linear filter with frequency response $\mathbf{V}^{-1}\mathbf{F}_{k\text{-}1}\mathbf{V} = \mathbf{I}_{N/M} \otimes M\, \mathbf{e}_k\mathbf{e}_k^{\mathrm{T}}$. We will then have:

$$\mathbf{x}_k = \mathbf{F}_{k\text{-}1}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{x}_k = \mathbf{F}_{k\text{-}1}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{H}_{k\text{-}1}\, \mathbf{x}. \tag{6.94}$$

So we have proved the following theorem.

Theorem 6.19 (PR filter banks on $\mathbf{\Omega}$-graphs)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of an* $\mathbf{\Omega}$*-graph with the following eigenvalue decomposition* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$*. Now consider the maximally decimated filter bank of Figure 6.16 with analysis and synthesis filters as follows:*

$$\mathbf{H}_{k\text{-}1} = \mathbf{V}\left(\mathbf{I}_{N/M} \otimes \mathbf{e}_k\, \mathbf{e}_k^{\mathrm{T}}\right) \mathbf{V}^{-1}, \qquad \mathbf{F}_{k\text{-}1} = M\, \mathbf{H}_{k\text{-}1} \tag{6.95}$$

for $1 \le k \le M$*. This is a perfect reconstruction system, that is,* $T(\mathbf{A})\,\mathbf{x} = \mathbf{x}$ *for all graph signals* $\mathbf{x}$*. When the eigenvalues are distinct,* $\mathbf{H}_k$*'s can be designed to be polynomials in* $\mathbf{A}$*, that is,* $\mathbf{H}_{k\text{-}1} = H_{k\text{-}1}(\mathbf{A})$*.*

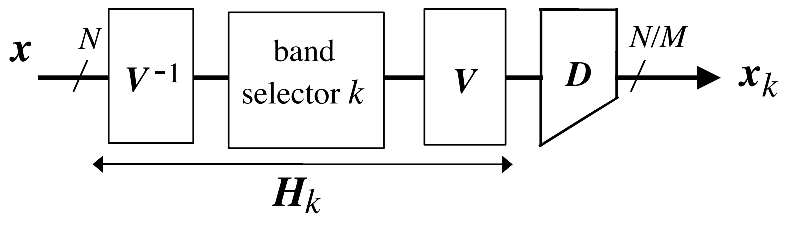
Notice that the frequency response of each filter, (6.95), has N/M nonzero values. These are similar to "ideal" band-limited filters (with bandwidth $2\pi/M$) in classical theory. In classical filter bank theory it is well known that an M-channel maximally decimated filter bank has perfect reconstruction if the filters are ideal "brickwall" filters chosen as

$$H_k(e^{j\omega}) = \begin{cases} 1, & 2\pi k/M \le \omega \le 2\pi(k{+}1)/M, \\ 0, & \text{otherwise,} \end{cases} \tag{6.96}$$

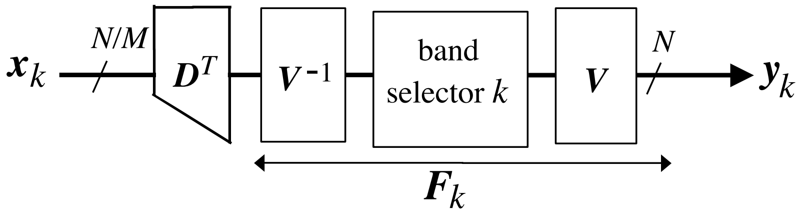
and $F_k(e^{j\omega}) = M\, H_k(e^{j\omega})$. The result of Theorem 6.19 for graph filter banks is analogous to that classical result. Notice that in classical theory, ideal filters have infinite duration impulse responses, whereas the graph filters are polynomials in $\mathbf{A}$ with at most N taps.

Figure 6.17a shows the details of one channel of the brickwall analysis bank, and Figure 6.17b shows the corresponding channel of the synthesis bank. The analysis filters have the form $\mathbf{H}_k = \mathbf{V}\,\mathbf{S}_k\mathbf{V}^{-1}$ where $\mathbf{V}^{-1}$ is the graph Fourier transform matrix and $\mathbf{S}_k = \mathbf{I}_{N/M} \otimes \mathbf{e}_k\mathbf{e}_k^T$ is a diagonal matrix (band selector) which retains N/M outputs of $\mathbf{V}^{-1}$ and sets the rest to zero. For example if $N = 6$ and $M = 3$ the three selector matrices are

$$\mathbf{S}_1 = \begin{bmatrix} 1&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0 \end{bmatrix}, \quad \mathbf{S}_2 = \begin{bmatrix} 0&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&0 \end{bmatrix}, \quad \mathbf{S}_3 = \begin{bmatrix} 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&1 \end{bmatrix}, \quad (6.97)$$



(a)



(b)

Figure 6.17: (a) The k^{th} channel of a graph filter bank, and (b) details of this channel when the filters are brickwall filters as defined in Theorem 6.19.

Once the appropriate outputs of the k^{th} band have been selected, the matrix $\mathbf{V}$ in $\mathbf{H}_k$ is used to convert the subband signal back to the "graph vertex domain." (This is similar to implementing the filtering operation in the frequency domain and taking inverse Fourier transform to come back to time domain.) This graph domain subband signal is then decimated by $\mathbf{D}$. For reconstruction, the synthesis filter $\mathbf{F}_k$ is similarly used. Thus, the implementation of the brick wall filter bank is not merely a matter of using the graph Fourier operator $\mathbf{V}^{-1}$, it also involves band selection, inverse transformation, and decimation. Since $\mathbf{V}^{-1}$ is common to all analysis filters, it contributes to a complexity of N^2 multiplications. The band selector $\mathbf{S}_k$, and the matrices $\mathbf{V}$ and $\mathbf{D}$ can be combined and implemented with $(N/M)^2$ multiplications for each k, so there is a total of N^2/M multiplications for this part. So the decimated analysis bank has complexity of about $N^2 + N^2/M$ which is $O(N^2)$. Once again, by approximating these brickwall filters with polynomial filters with order L we can reduce the complexity to NL. (See Figure 6.7.)

The perfect reconstruction result in Theorem 6.19 is restricted to $\mathbf{\Omega}$-graphs. However this restriction can be removed by applying a similarity transformation to the graph as described later in Section 6.15.

As a final remark, filters in (6.95) are not unique in the sense that we can design different filters and still have $T(\mathbf{A}) = \mathbf{I}$ in the filter bank.

6.12 Graph Filter Banks on M-Block Cyclic Graphs

In Theorem 6.19 aliasing was totally suppressed in each channel by the use of ideal filters (6.95), which explains why the filter polynomials had order N-1. But if we resort to cancellation of aliasing among different channels, it is less restrictive on the filters. While this idea is not easy to develop for arbitrary graphs, the theory can be developed under some further assumptions on the graph, namely that $\mathbf{A}$ be M-block cyclic as we shall see. We will see that such filter banks have many advantages compared to those given by Theorem 6.19, which does not use the M-block cyclic assumption and applies to more general class of $\mathbf{\Omega}$-graphs.

As shown in Theorem 6.8, M-block cyclic graphs satisfy the noble identity condition in (6.27). Therefore, Theorem 6.17 shows that a maximally decimated M-channel filter bank on an M-block cyclic graph is a periodically shift-varying system with period M. This can be interpreted as aliasing when the graph $\mathbf{A}$ is diagonalizable (Theorem 6.15 and Figure 6.15). As in the classical case, it is possible to cancel out aliasing components arising from different channels, and in fact, achieve perfect reconstruction, that is, $T(\mathbf{A}) = \mathbf{A}^n$ for some integer n as we shall see. [2] We begin by proving the following result:

Theorem 6.20 (PR filter banks on M-block cyclic graphs)**.** *Consider the graph*

[2]In analogy with classical filter banks where $T(z) = z^{-n}$ signifies the PR property, we take $T(\mathbf{A}) = \mathbf{A}^n$ to be the PR property. But this makes practical sense only in situations where $\mathbf{A}$ is invertible so that the distortion $\mathbf{A}^n$ can be canceled.

filter bank of Figure 6.16 and assume that the adjacency matrix of the graph is diagonalizable M-block cyclic. With no further restrictions on the graph, the system has perfect reconstruction if and only if

$$\sum_{k=0}^{M\text{-}1} F_k(\lambda)\, H_k(w^l\,\lambda) = M\,\lambda^n\,\delta(l), \tag{6.98}$$

for some n, for all $\lambda \in \mathbb{C}$, and for all l in $0 \le l \le M\text{-}1$, where $\delta(\cdot)$ is the discrete Dirac function.

Proof. The overall response of the system in Figure 6.16 can be written as

$$T(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} F_k(\mathbf{A})\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,H_k(\mathbf{A}) = \frac{1}{M}\sum_{l=0}^{M\text{-}1}\sum_{k=0}^{M\text{-}1} F_k(\mathbf{A})\,\mathbf{\Omega}^l\,H_k(\mathbf{A}), \tag{6.99}$$

where we have used (6.66). For simplicity, define

$$S_l(\mathbf{A}) = \sum_{k=0}^{M\text{-}1} F_k(\mathbf{A})\,\mathbf{\Omega}^l\,H_k(\mathbf{A}). \tag{6.100}$$

Therefore, the response of the system is

$$T(\mathbf{A}) = \frac{1}{M}\left(\underbrace{S_0(\mathbf{A})}_{\text{Polynomial}} + \underbrace{S_1(\mathbf{A}) + \cdots + S_{M\text{-}1}(\mathbf{A})}_{\text{Alias components}}\right). \tag{6.101}$$

Notice that $S_0(\mathbf{A})$ is the sum of products of polynomials in $\mathbf{A}$, therefore it is also a polynomial in $\mathbf{A}$, hence it is alias-free since $\mathbf{A}$ is assumed to be diagonalizable. However, for $l \ge 1$, there exists $\mathbf{\Omega}^l$ term in each $S_l(\mathbf{A})$ in (6.100), which does not commute with $\mathbf{A}$ in general. As a result $S_l(\mathbf{A})$ is not shift-invariant and results in aliasing (Theorem 6.15). Perfect reconstruction $T(\mathbf{A}) = \mathbf{A}^n$ can be achieved by imposing

$$S_0(\mathbf{A}) = M\,\mathbf{A}^n, \qquad \sum_{l=1}^{M\text{-}1} S_l(\mathbf{A}) = \mathbf{0}. \tag{6.102}$$

The second equation above is the alias cancellation condition. Using the eigenvalue decomposition of the adjacency matrix, $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\text{-}1}$, the first condition in (6.102) reduces to

$$\mathbf{V}\left(\sum_{k=0}^{M\text{-}1} F_k(\mathbf{\Lambda})\,H_k(\mathbf{\Lambda})\right)\mathbf{V}^{\text{-}1} = M\,\mathbf{V}\mathbf{\Lambda}^n\mathbf{V}^{\text{-}1}. \tag{6.103}$$

Since $\mathbf{\Lambda}$ is a diagonal matrix consisting of eigenvalues, this can be further simplified to

$$\sum_{k=0}^{M\text{-}1} F_k(\lambda_{i,j})\,H_k(\lambda_{i,j}) = M\,\lambda_{i,j}^n, \tag{6.104}$$

for all eigenvalues, $\lambda_{i,j}$, of the adjacency matrix $\mathbf{A}$.

Now consider the second condition in (6.102). With the eigenvalue decomposition, it reduces to

$$\sum_{l=1}^{M\text{-}1} \sum_{k=0}^{M\text{-}1} F_k(\mathbf{\Lambda})\ \mathbf{V}^{-1}\ \mathbf{\Omega}^l\ \mathbf{V}\ H_k(\mathbf{\Lambda}) = \mathbf{0}. \tag{6.105}$$

Since $\mathbf{A}$ is M-block cyclic, it satisfies the eigenvector structure in (6.57). So we can use (6.71) and re-write (6.105) as

$$\sum_{l=1}^{M\text{-}1} \sum_{k=0}^{M\text{-}1} F_k(\mathbf{\Lambda})\ \mathbf{\Pi}_l\ H_k(\mathbf{\Lambda}) = \mathbf{0}. \tag{6.106}$$

Notice that the permutation matrix $\mathbf{\Pi}_l$ is defined as the l^{th} power of the cyclic matrix of size M ((6.72)). Since the supports of $\mathbf{C}_M^{l_1}$ and $\mathbf{C}_M^{l_2}$ have no common index for different l_1 and l_2, supports of $\mathbf{\Pi}_{l_1}$ and $\mathbf{\Pi}_{l_2}$ will also have no common index. Furthermore, due to $\mathbf{\Lambda}$ being a diagonal matrix, the support of the term $F_k(\mathbf{\Lambda})\,\mathbf{\Pi}_l\,H_k(\mathbf{\Lambda})$ will be the same as the support of $\mathbf{\Pi}_l$. Combining both arguments, we can say that (6.106) holds if and only if the inner sum is zero for each l, that is,

$$\sum_{k=0}^{M\text{-}1} F_k(\mathbf{\Lambda})\ \mathbf{\Pi}_l\ H_k(\mathbf{\Lambda}) = \mathbf{0}, \qquad \forall\, l \in \{1, \cdots, M\text{-}1\}. \tag{6.107}$$

Since $\mathbf{\Lambda}$ is a diagonal matrix with the ordering scheme in (6.59), the permutation $\mathbf{\Pi}_l$ circularly shifts each eigenvalue of an eigenfamily. Hence, (6.107) is equivalent to

$$\sum_{k=0}^{M\text{-}1} F_k(\lambda_{i,j+l})\ H_k(\lambda_{i,j}) = 0, \tag{6.108}$$

for all $1 \leq l \leq M\text{-}1$. Since $\mathbf{A}$ is M-block cyclic, eigenvalues of $\mathbf{A}$ satisfy (6.56). Then (6.108) can be written as

$$\sum_{k=0}^{M\text{-}1} F_k(\lambda_{i,j+l})\ H_k(w^{M\text{-}l}\ \lambda_{i,j+l}) = 0. \tag{6.109}$$

By changing the index variables, (6.109) can be simplified as follows:

$$\sum_{k=0}^{M\text{-}1} F_k(\lambda_{i,j})\ H_k(w^{l}\ \lambda_{i,j}) = 0, \tag{6.110}$$

for all $1 \leq l \leq M\text{-}1$ and for all eigenvalues, $\lambda_{i,j}$, of the adjacency matrix $\mathbf{A}$. Combining (6.104) and (6.110), if the filter bank in Figure 6.16 provides PR, (6.98) should be satisfied for all eigenvalues of $\mathbf{A}$. Since we want PR independent of the graph, (6.98) should be satisfied for all $\lambda \in \mathbb{C}$. Hence, it is a necessary condition.

Conversely, assume that the filters satisfy (6.98), hence (6.104) and (6.110) are satisfied. Since the graph is assumed to be diagonalizable M-block cyclic, (6.104) and (6.110) are equivalent to (6.102). Therefore, the overall response of the filter bank is $\mathbf{A}^n$, where n satisfies (6.104). So the filter bank has the PR property. □

It is quite interesting to observe that the condition (6.98) on the graph filters is the same as the PR condition in classical multirate theory. We state this equivalence in the following theorem.

Theorem 6.21 (PR condition equivalence)**.** *A set of polynomials,* $\{H_k(\lambda), F_k(\lambda)\}$, *provides PR in maximally decimated M-channel FB on all M-block cyclic graphs with diagonalizable adjacency matrix if and only if* $\{H_k(z), F_k(z)\}$ *provides PR in the classical maximally decimated M-channel FB.*

Proof. The PR condition (6.98) is the same as the PR condition for the classical maximally decimated M-channel filter bank [200]. □

At the moment of this writing, we do not have examples of graphs other than M-block cyclic for which such results can be developed. At the expense of restraining the eigenvalues of the adjacency matrix to have the structure in (6.56), Theorem 6.20 offers three significant benefits compared to construction of PR filter banks on $\mathbf{\Omega}$-graphs discussed in Section 6.11.

First of all, (6.98) puts a condition on the filter coefficients *independent* of the graph as long as the graph is M-block cyclic with diagonalizable adjacency matrix. A change in the adjacency matrix does not affect the PR property. As a result, the response of the overall graph filter bank is robust to ambiguities in the adjacency matrix.

Secondly, eigenvalues of the adjacency matrix $\mathbf{A}$ do not need to be distinct since the condition solely depends on the filter coefficients.

Lastly, filter banks on M-block cyclic graphs are legitimate generalization of the classical multirate theory to graph signals due to Theorem 6.21. In order to design PR filter banks on an M-block cyclic graph, we can use any algorithm developed in the classical multirate theory [200, 199]. As an example, consider the following set of polynomials,

$$\begin{aligned} H_0(\lambda) &= 5 + 2\lambda + \lambda^3 + 2\lambda^4 + \lambda^5, & F_0(\lambda) &= 3\lambda^3\text{-}2\lambda^4 + \lambda^5, \\ H_1(\lambda) &= 2 + \lambda + 2\lambda^3 + 4\lambda^4 + 2\lambda^5, & F_1(\lambda) &= \text{-}8\lambda^3 + 5\lambda^4\text{-}2\lambda^5, \\ H_2(\lambda) &= \lambda^3 + 2\lambda^4 + \lambda^5, & F_2(\lambda) &= 1 + 13\lambda^3\text{-}8\lambda^4 + 3\lambda^5. \end{aligned} \tag{6.111}$$

In classical theory, this is a 3-channel PR filter bank [200]. Notice that these polynomials satisfy (6.98) with $n = 5$. Therefore, overall response of the filter bank constructed with the polynomials in (6.111) on any 3-block cyclic graph will be $T(\mathbf{A}) = \mathbf{A}^5$.

For a randomly generated adjacency matrix of a 3-block cyclic graph, channel responses of the filter bank utilizing the filters in (6.111) are shown in Figure 6.18. Response of each channel has non-zero blocks with large values in it (relative to the overall response of the FB), which results in large alias terms. Notice that some blocks of the channel responses cancel out each other exactly so that overall response, $T(\mathbf{A})$, is equal to $\mathbf{A}^5$ (up to numerical precision) as seen from Figure 6.18(f).

6.13 Polyphase Representations

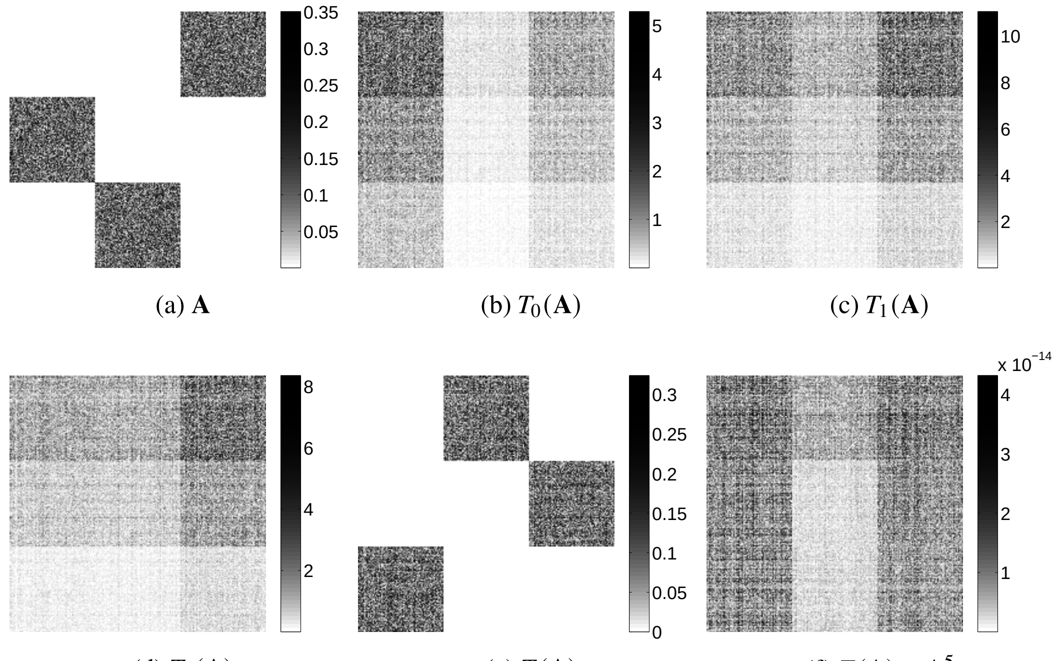
For the polyphase implementation of the filter bank in Figure 6.16, we decompose the analysis filters using Type-1 polyphase structure in (6.31) and synthesis filters with Type-2 decomposition in (6.32). Then we have

$$H_k(\mathbf{A}) = \sum_{l=0}^{M\text{-}1} \mathbf{A}^l \, E_{k,l}(\mathbf{A}^M), \quad F_k(\mathbf{A}) = \sum_{l=0}^{M\text{-}1} \mathbf{A}^{M\text{-}1\text{-}l} \, R_{l,k}(\mathbf{A}^M). \tag{6.112}$$

Using the polyphase implementation of decimation and interpolation filters given in (6.33) and (6.34) (they are described schematically in Figure 6.4 and Figure 6.5, respectively), we can define the polyphase component matrices as follows:

$$\big(\mathbf{E}(\bar{\mathbf{A}})\big)_{i,j} = E_{i\text{-}1,j\text{-}1}(\bar{\mathbf{A}}), \quad \big(\mathbf{R}(\bar{\mathbf{A}})\big)_{i,j} = R_{i\text{-}1,j\text{-}1}(\bar{\mathbf{A}}), \tag{6.113}$$

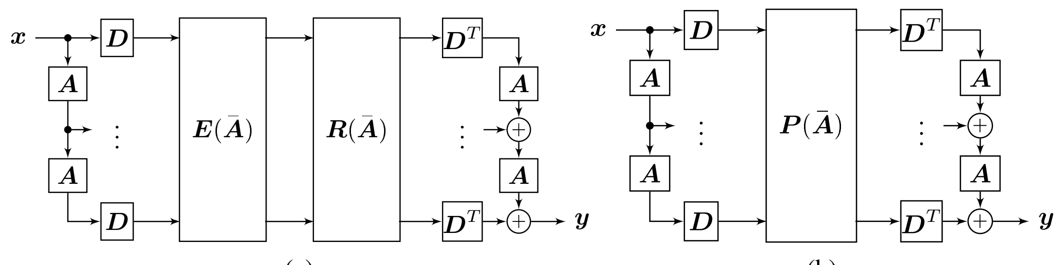
for $1 \leq i, j \leq M$ where $\mathbf{E}(\bar{\mathbf{A}}) \in \mathbb{C}^{N\times N}$ is the polyphase matrix for the analysis filters, $\mathbf{R}(\bar{\mathbf{A}}) \in \mathbb{C}^{N\times N}$ is the polyphase matrix for the synthesis filters, and $\bar{\mathbf{A}}$ is the adjusted shift operator given in (6.28). Notice that each block $(\mathbf{E}(\bar{\mathbf{A}}))_{i,j}$ and $(\mathbf{R}(\bar{\mathbf{A}}))_{i,j}$ is a polynomial in $\bar{\mathbf{A}}$, whereas overall polyphase matrices $\mathbf{E}(\bar{\mathbf{A}})$ and $\mathbf{R}(\bar{\mathbf{A}})$ are not polynomials in $\bar{\mathbf{A}}$.



(a) $\mathbf{A}$ (b) $T_0(\mathbf{A})$ (c) $T_1(\mathbf{A})$

(d) $T_2(\mathbf{A})$ (e) $T(\mathbf{A})$ (f) $T(\mathbf{A}) - \mathbf{A}^5$

Figure 6.18: (a) Adjacency matrix of a 3-block cyclic graph of size 210 with randomly generated complex edge weights. Response of (b) the zeroth channel, (c) the first channel, (d) the second channel of 3-channel FB in Figure 6.16 on graph given in (a). (e) is the overall response of the FB. (f) is the difference between $T(\mathbf{A})$ and $\mathbf{A}^5$. All figures show the element-wise absolute values of the matrices.


$\boldsymbol{x}$
$\boldsymbol{D}$
$\boldsymbol{A}$
$\boldsymbol{E}(\bar{\boldsymbol{A}})$
$\boldsymbol{R}(\bar{\boldsymbol{A}})$
$\boldsymbol{D}^T$
$\boldsymbol{y}$
$\boldsymbol{P}(\bar{\boldsymbol{A}})$

(a) (b)

Figure 6.19: (a) Polyphase representation of the maximally decimated M-channel filter bank in Figure 6.16 on a graph with the adjacency matrix $\mathbf{A}$ that satisfies the noble identity condition (6.27). (b) Combined representation of the polyphase matrices. The decimation matrix $\mathbf{D}$ is as in (6.8) and the polyphase transfer matrix is given in (6.114). $\bar{\mathbf{A}}$ is as in (6.28).

Using the polyphase matrices defined in (6.113), we can implement the filter bank in Figure 6.16 as in Figure 6.19a. This can be redrawn as in Figure 6.19b where $\mathbf{P}(\bar{\mathbf{A}}) = \mathbf{R}(\bar{\mathbf{A}})\,\mathbf{E}(\bar{\mathbf{A}})$. Due to partitioning of the polyphase component matrices in (6.113), we have the following result for the partitions of $\mathbf{P}(\bar{\mathbf{A}})$:

$$(\mathbf{P}(\bar{\mathbf{A}}))_{i,j} = \sum_{k=0}^{M-1} R_{i-1,k}(\bar{\mathbf{A}})\ E_{k,j-1}(\bar{\mathbf{A}}). \tag{6.114}$$

Notice that polyphase transfer matrix provides the polyphase implementation of the filter bank in Figure 6.16, only if the graph satisfies the noble identity condition in (6.27). Summarizing, we have proved:

Theorem 6.22 (Polyphase implementation of a filter bank)**.** *On a graph with the adjacency matrix satisfying the noble identity condition* (6.27)*, the maximally decimated M-channel FB given in Figure 6.16 has the polyphase implementation given in Figure 6.19 where polyphase transfer matrix,* $\mathbf{P}(\bar{\mathbf{A}})$*, is as in* (6.114)*, and* $\bar{\mathbf{A}}$ *is the adjusted shift operator given in* (6.28)*.*

Since M-block cyclic matrices satisfy the noble identity condition (6.27), the polyphase representation of Figure 6.19 is valid for graph filter banks on M-block cyclic graphs. However, M-block cyclic property is not *necessary* for this. The condition (6.27) is enough.

In Section 6.11, we showed that it is possible to construct PR filter banks on $\mathbf{\Omega}$-graphs. In order to talk about polyphase implementation of such filter banks, we need to characterize the set of graph matrices $\mathbf{A}$ that satisfy both the noble identity condition in (6.27) and have the $\mathbf{\Omega}$-structure in (6.57). From Theorem 6.8 and Theorem 6.9, we know that M-block cyclic graphs with diagonalizable adjacency matrices belong to that set. It is interesting to observe that any matrix that belongs to this set is *similar* to an M-block cyclic graph. We state this fact in the following theorem whose proof is given in Section 6.18.3.

Theorem 6.23 (The noble identity condition and the eigenvector structure)**.** *Let* $\mathbf{A}$ *have distinct eigenvalues with the eigenvalue decomposition* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$*. If* $\mathbf{A}$ *satisfies the noble identity condition in* (6.27) *and has the* $\mathbf{\Omega}$*-structure in* (6.57)*, then* $\mathbf{A}$ *is similar to an M-block cyclic matrix. More precisely, there exists a permutation matrix* $\mathbf{\Pi}$ *such that* $(\mathbf{V\Pi})\mathbf{\Lambda}(\mathbf{V\Pi})^{-1}$ *is M-block cyclic.*

6.13.1 Alias-free and PR Property for M-Block Cyclic Graphs

Returning to Figure 6.19 for the polyphase representation of a graph filter bank, we now provide a sufficient condition on $P(\bar{\mathbf{A}})$ for the PR property on M-block cyclic graphs. This is an extension of a similar condition ($\mathbf{P}(z) = \mathbf{I}$) that guarantees PR in classical filter banks [200].

Theorem 6.24 (Sufficiency condition for PR in polyphase filter banks on M-block cyclic graphs)**.** *Consider the polyphase implementation of a maximally decimated M-channel filter bank in Figure 6.19b, and assume that the adjacency matrix of the graph,* $\mathbf{A}$*, is M-block cyclic. The system has PR property if the polyphase transfer matrix has the following form:*

$$\mathbf{P}(\bar{\mathbf{A}}) = \mathbf{I}_M \otimes \bar{\mathbf{A}}^m, \tag{6.115}$$

for some non-negative integer m, and $\bar{\mathbf{A}}$ *is as in* (6.28).

Proof. Assume that $\mathbf{P}(\bar{\mathbf{A}})$ has the form in (6.115). Then, overall response of the filter bank in Figure 6.19b is written as

$$T(\mathbf{A}) = \sum_{k=0}^{M-1} \mathbf{A}^{M-1-k}\, \mathbf{D}^{\mathrm{T}}\, \bar{\mathbf{A}}^m\, \mathbf{D}\, \mathbf{A}^k = \sum_{k=0}^{M-1} \mathbf{A}^{M-1-k}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{A}^{Mm}\, \mathbf{A}^k, \tag{6.116}$$

$$= \left(\sum_{k=0}^{M-1} \mathbf{A}^{M-1-k}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{A}^k \right) \mathbf{A}^{Mm} = \mathbf{A}^{M-1+Mm}, \tag{6.117}$$

where we use the first noble identity (6.25) in (6.116) and the lazy FB PR property (6.30) in (6.117), since M-block cyclic matrices satisfy both of these properties. □

Notice that when we let $m = 0$ in Theorem 6.24, we get $\mathbf{P}(\bar{\mathbf{A}}) = \mathbf{I}_N$ which corresponds to the lazy filter bank structure given in Figure 6.3b. This observation agrees with the result given by Theorem 6.8 for M-block cyclic graphs.

For the most complete characterization of the PR property on M-block cyclic graphs, we provide the following result, whose proof is provided in Section 6.18.4.

Theorem 6.25 (PR polyphase filter banks on M-block cyclic graphs)**.** *Consider the polyphase implementation of a maximally decimated M-channel filter bank in Figure 6.19b, and assume that the adjacency matrix of the graph,* $\mathbf{A}$*, is an invertible M-block cyclic matrix. The system has PR property if and only if the polyphase transfer matrix has the following form:*

$$\mathbf{P}(\bar{\mathbf{A}}) = \left(\mathbf{I}_M \otimes \bar{\mathbf{A}}^m \right) \begin{bmatrix} \mathbf{0} & \mathbf{I}_{M-n} \otimes \mathbf{I}_{N/M} \\ \mathbf{I}_n \otimes \bar{\mathbf{A}} & \mathbf{0} \end{bmatrix}, \tag{6.118}$$

for some integers m, n with $0 \leq n \leq M\text{-}1$. $\bar{\mathbf{A}}$ *is as in* (6.28).

When we waive the perfect reconstruction property and ask for an alias-free response, we get a more relaxed condition on the polyphase transfer matrix. The following theorem states the necessary condition for this case. This is a generalization of the classical pseudocirculant condition for alias cancellation [200] to graph filter banks.

Theorem 6.26 (Alias-free polyphase filter banks on M-block cyclic graphs)**.** *Consider the polyphase implementation of a maximally decimated M-channel filter bank in Figure 6.19b, and assume that the adjacency matrix of the graph,* $\mathbf{A}$*, is M-block cyclic with distinct eigenvalues. The system has alias-free property if and only if the polyphase transfer matrix has the following pseudo-block-circulant form:*

$$\mathbf{P}(\bar{\mathbf{A}}) = \begin{bmatrix} P_0(\bar{\mathbf{A}}) & P_1(\bar{\mathbf{A}}) & \cdots & P_{M\text{-}1}(\bar{\mathbf{A}}) \\ \bar{\mathbf{A}}\,P_{M\text{-}1}(\bar{\mathbf{A}}) & P_0(\bar{\mathbf{A}}) & \cdots & P_{M\text{-}2}(\bar{\mathbf{A}}) \\ \vdots & \ddots & \ddots & \vdots \\ \bar{\mathbf{A}}\,P_1(\bar{\mathbf{A}}) & \bar{\mathbf{A}}\,P_2(\bar{\mathbf{A}}) & \cdots & P_0(\bar{\mathbf{A}}) \end{bmatrix}. \tag{6.119}$$

Proof. Assume that the overall response of the filter bank is alias-free, and the graph has distinct eigenvalues. Then, due to Theorem 6.15, the response of the filter bank is a polynomial filter. By re-indexing the coefficients, we can decompose the polynomial response as follows:

$$T(\mathbf{A}) = \sum_{m=0}^{N/M\text{-}1} \sum_{k=0}^{M\text{-}1} \alpha_{m,k}\ \mathbf{A}^{M\text{-}1+Mm+k}. \tag{6.120}$$

Since $T(\mathbf{A})$ is a linear combination of PR systems, using Theorem 6.25 and linearity, we can write the polyphase transfer matrix that corresponds to (6.120) in the following way:

$$\mathbf{P}(\bar{\mathbf{A}}) = \sum_{m=0}^{N/M\text{-}1} \sum_{k=0}^{M\text{-}1} \alpha_{m,k} \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{I}_{M\text{-}k} \otimes \bar{\mathbf{A}}^{m} \\ \mathbf{I}_k \otimes \bar{\mathbf{A}}^{m+1} & \mathbf{0} \end{bmatrix}}_{\text{pseudo-block-circulant}}. \tag{6.121}$$

By inspection, we can see that the block-matrix in (6.121) is a pseudo-block-circulant matrix. Since a linear combination of pseudo-block-circulant matrices is also a pseudo-block-circulant, the polyphase transfer matrix has the form in (6.119) for some polynomials.

Conversely, if the matrix $\mathbf{P}(\bar{\mathbf{A}})$ has the form in (6.119), then we can decompose it as in (6.121) for some set of $\alpha_{m,k}$. Therefore, due to Theorem 6.25, the overall response of the filter bank is a linear combination of perfect reconstruction systems. It is therefore alias-free. □

6.13.2 Importance of Polyphase Representations for Graph Filter Banks

We know that the polyphase implementation in Figure 6.19 is valid as long as the conditions (6.27) and (6.28) for noble identities are satisfied. Here each polyphase component $E_{k,l}(\bar{\mathbf{A}}) \in \mathbb{C}^{(N/M)\times(N/M)}$ is a polynomial in the matrix $\bar{\mathbf{A}}$:

$$E_{k,l}(\bar{\mathbf{A}}) = e_{k,l}(0)\,\mathbf{I} + e_{k,l}(1)\,\bar{\mathbf{A}} + \cdots + e_{k,l}(K)\,\bar{\mathbf{A}}^K, \tag{6.122}$$

where $\bar{\mathbf{A}} = \mathbf{D}\mathbf{A}^M\mathbf{D}^{\mathrm{T}}$ as in (6.28), and $K \approx L/M$. (L is the order of the filters in Fig 6.16.) Assuming that $\bar{\mathbf{A}}$ is sparse and can be implemented with negligible overhead, the complexity for $E_{k,l}(\bar{\mathbf{A}})$ (implemented similar to Figure 6.7) is about $(L/M)(N/M)$ so that the entire polyphase matrix with M^2 such submatrices has complexity LN, identical to the non polyphase implementation of polynomial filter banks (with each filter implemented as in Figure 6.7). In fact even if $\mathbf{A}$ is sparse, the matrix $\bar{\mathbf{A}} = \mathbf{D}\mathbf{A}^M\mathbf{D}^{\mathrm{T}}$ is in general not. So the polyphase implementation may even have higher complexity than direct implementation of polynomial filters as in Figure 6.7. Thus, while in classical filter banks the polyphase implementation reduces the number of multiplications per unit time, *there is no such advantage in graph filter banks.*

Then the question is, what is the advantage of the polyphase representation of Figure 6.19 for graph filter banks? The answer lies in the *design phase* rather than the complexity of implementation. To explain, suppose we want to optimize the polynomial analysis filters to achieve certain properties of the decimated subband signals $\{\mathbf{x}_0, \mathbf{x}_1, \ldots\}$ (e.g., sparsity), by using some apriori information on the statistics of the graph signal $\mathbf{x}$. If we do this directly by optimizing the multipliers $h_k(n)$ in the structure of Figure 6.7 then it is not easy to constraint these coefficients (during optimization) such that there will exist a perfect reconstruction polynomial synthesis filter bank $\{F_k(\mathbf{A})\}$. But if we optimize the coefficients $e_{k,l}(n)$ of the polynomials (6.122), then it can be shown that as long as the equivalent classical polynomial matrix

$$\mathbf{E}(z) = \begin{bmatrix} E_{0,0}(z) & E_{0,1}(z) & \dots & E_{0,M\text{-}1}(z) \\ E_{1,0}(z) & E_{1,1}(z) & \dots & E_{1,M\text{-}1}(z) \\ \vdots & \vdots & \ddots & \vdots \\ E_{M\text{-}1,0}(z) & E_{M\text{-}1,1}(z) & \dots & E_{M\text{-}1,M\text{-}1}(z) \end{bmatrix} \tag{6.123}$$

has a polynomial inverse $\mathbf{R}(z)$ (i.e., FIR inverse), there will exist a polynomial graph synthesis bank $\{F_k(\mathbf{A})\}$ with perfect reconstruction property. Now, the construction of classical polynomial matrices $\mathbf{E}(z)$ with polynomial inverses $\mathbf{R}(z)$ is a well studied problem in filter bank theory and includes special families such as FIR paraunitary matrices, FIR unimodular matrices and so forth [200]. So, we can take advantage of the results from classical literature to design optimal graph filter banks that suit specific signal statistics, if we use the polyphase representation of Figure 6.19 in the design process. The implementation of each filter in the bank could later be done directly using Figure 6.7, which is more economical.

6.14 Frequency Domain Analysis

In order to quantitatively interpret the amount of fluctuation of a signal, *the total variation* of the signal $\mathbf{x}$ on a graph with the adjacency matrix $\mathbf{A}$ is defined as [153]:

$$\mathrm{S}(\mathbf{x}) = \|\mathbf{x} - \mathbf{A}\mathbf{x}\|_1, \tag{6.124}$$

where it is assumed that the adjacency matrix is normalized such that the maximum eigenvalue has a unit magnitude [153]. With the definition in (6.124), the *frequency* of an eigenvector $\mathbf{v}$ with the corresponding eigenvalue λ becomes

$$\mathrm{S}(\mathbf{v}) = |1 - \lambda|, \tag{6.125}$$

where all the eigenvectors are assumed to be scaled such that they have unit ℓ_1 norm.

When the eigenvalues are *real*, from (6.125) it is clear that two eigenvectors with distinct eigenvalues cannot have the same amount of total variation. Hence, distinct eigenvalues imply distinct total variations, and vice versa. However, for complex eigenvalues this implication does not hold. As a simple example consider two distinct complex eigenvalues $\lambda_1 = (\sqrt{2}\text{-}1)/\sqrt{2}$ and $\lambda_2 = 1/2\text{+}j/2$. Clearly $\lambda_1 \neq \lambda_2$, yet we have $\mathrm{S}(\mathbf{v}_1) = \mathrm{S}(\mathbf{v}_2)$.

The problem with complex eigenvalues arises when we want to describe the frequency domain behavior of a polynomial filter, $H(\lambda)$. How the filter suppresses or amplifies eigenvectors according to their total variation determines the characteristics. When there are complex eigenvalues, a filter may respond differently to eigenvectors with the same total variation. More precisely, we may have $|H(\lambda_1)| \neq |H(\lambda_2)|$ even when $\mathrm{S}(\mathbf{v}_1) = \mathrm{S}(\mathbf{v}_2)$. As a result, we may not even be able to define the behavior of the filter (low-pass, high-pass etc.). Therefore, filters cannot be designed independent of the graph spectrum when the spectrum has complex values.

The question we ask here is as follows: under what conditions can we design filters with meaningful behavior *for all graphs* that satisfy such conditions? One obvious answer is the case of real eigenvalues since $\mathrm{S}(\mathbf{v}_1) = \mathrm{S}(\mathbf{v}_2)$ implies $\lambda_1 = \lambda_2$, hence $|H(\lambda_1)| = |H(\lambda_2)|$. As a result, a filter behaves in the same way for all graphs with real eigenvalues.

For the complex case, we cannot answer the question in the general sense for the time being. Nonetheless, when we constrain the eigenvalues to be on the unit circle, we have the following result.

Theorem 6.27 (Magnitude response of a polynomial filter, and unit-modulus eigenvalues)**.** *Let $H(\lambda)$ be a polynomial filter with* real *coefficients. Then $H(\lambda)$ has a well-defined magnitude response w.r.t. the total variation* for all *graphs with diagonalizable adjacency matrices with unit modulus eigenvalues.*

Proof. Let eigenvalues of the adjacency matrix be on the unit circle. Then we have $\lambda = e^{-j\,\theta}$ for some $0 \leq \theta < 2\pi$. Then, the total variation in (6.125) reduces to

$$\mathrm{S}(\mathbf{v}) = |1 - e^{-j\,\theta}| = 2\,|\sin(\theta/2)|. \tag{6.126}$$

Due to symmetry of (6.126), eigenvectors will have the same total variation *if and only if* the corresponding eigenvalues are conjugate pairs. Assume that $H(\lambda)$ is a polynomial filter with real coefficients. Then, its magnitude response will have the conjugate symmetry, which is to say $|H(\lambda)| = |H(\lambda^*)|$, for all $|\lambda| = 1$. As a result, even though the same total variation may correspond to a conjugate eigenvalue pair, those eigenvalues will be mapped to the same magnitude. Hence, magnitude response of the filter with respect to the total variation will be well-defined and remain the same for all graphs with unit modulus eigenvalues. □

A drawback of (6.126) is the non-linearity of it in terms of the phase angle. We will demonstrate this in the following example. Consider the filter bank coefficients provided in Table 6.5.1 on page 318 of [200]. These analysis filters are optimized for perfect reconstruction in 3-channel filter banks in the classical multirate theory with synthesis filters having the following coefficients $f_{k,l} = h_{k,14\text{-}l}$ [200]. Furthermore, they are designed such that each filter allows only one uniform sub-band to pass. That is, $H_k(z)$ passes frequencies in the range $|\omega| \in [\pi k/3 \quad \pi(k{+}1)/3]$. Their magnitude responses are shown in Figure 6.20a. When all the eigenvalues of $\mathbf{A}$ are on the unit circle, we can quantify the magnitude response of each analysis filter w.r.t. the total variation due to Theorem 6.27. However, the pass bands of $H_0(\mathbf{A})$, $H_1(\mathbf{A})$, and $H_2(\mathbf{A})$ in the total variation are $[0, 1]$, $[1, \sqrt{3}]$, and $[\sqrt{3}, 2]$, respectively. They are visualized in Figure 6.20b.

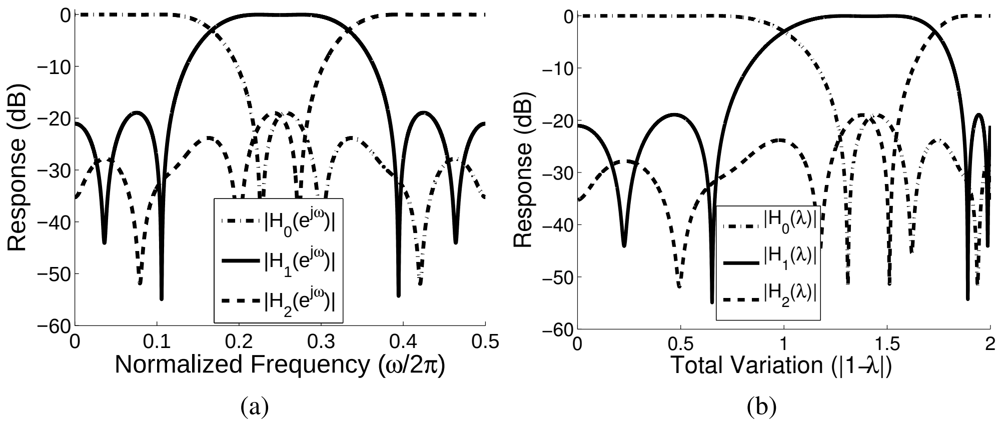


Figure 6.20: Magnitude response of the filters given in Table 6.5.1 of [200]. (a) is the magnitude response in the classical theory. (b) is the magnitude response w.r.t. the total variation in (6.125).

It is important to notice that magnitude responses of the graph filters shown in Figure 6.20b do *not* explicitly depend on the eigenvalues of the graph as long as they are on the unit circle. *Even repeated eigenvalues are tolerated.* Furthermore, magnitude response and the length of the filters are unrelated with the size of the graph. This makes the system robust to ambiguities in the graph. Remember that for a given specific graph with distinct eigenvalues, we can always construct polynomial filters with the desired magnitude response using (6.79). However, those filters are unique to that specific graph and quite sensitive to imperfections in the eigenvalues.

Notice that for a maximally decimated filter bank as in Figure 6.16, having the PR property and having a well-defined magnitude response are two different concepts, and they do not imply each other. To see this, consider the polynomials in (6.111). When the adjacency matrix is diagonalizable M-block cyclic, those polynomials provide PR on the filter bank. Yet, they may have a different frequency behavior w.r.t. the total variation for different M-block cyclic graphs. Conversely, we may select the filters with real coefficients. Then, for graphs with unit modulus eigenvalues, we have a graph independent characteristics due to Theorem 6.27. Yet, we cannot expect them to provide the PR property. Therefore, we reach the following conclusion: even if we assume unit-magnitude eigenvalues in the adjacency matrix, filters given in Table 6.5.1 of [200] do not provide perfect reconstruction on the filter bank. When we further assume that the graph is 3-block cyclic, only then 3-channel filter bank on the graph provides perfect reconstruction with each channel allowing only a sub-band of the total variation spectrum. Moreover, the characteristics of the graph filter bank, PR and magnitude response, will be independent of the actual values of the eigenvalues and the size of the graph, provided that they satisfy the conditions.

The above results show that we can have robust filter banks with practical use for signals on M-block cyclic graphs with eigenvalues having unit magnitude. More importantly, design of such multirate graph systems is not an issue. Due to Theorem 6.21 and Theorem 6.27, any algorithm developed for classical signal processing will serve the purpose. We summarize this observation in the following theorem.

Theorem 6.28 (PR graph filter banks with well-defined magnitude response)**.** *Consider the graph filter bank of Figure 6.16, and assume that the adjacency matrix of the graph is diagonalizable M-block cyclic with unit magnitude eigenvalues. If the analysis and the synthesis filters satisfy* (6.98) *with real polynomial coefficients, then, the filter bank provides perfect reconstruction, and each channel has a well-defined magnitude response w.r.t. total variation spectra.*

As an application of Theorem 6.28, consider the cyclic graph $\mathbf{C}_N$. Due to Fact 6.4, $\mathbf{C}_N$ is an M-block cyclic graph. Furthermore, its eigenvalues are in the form of $\lambda_k = e^{-j2\pi k/N}$ for $0 \leq k \leq N$-1, hence $|\lambda_k| = 1$. As a result, Theorem 6.28 applies to $\mathbf{C}_N$. Remember that, when the adjacency matrix is $\mathbf{C}_N$, graph signal processing reduces to the classical theory [152]. This observation shows that Theorem 6.28 agrees with the classical multirate theory and generalizes it to the graph case with some restrictions on the graph.

6.15 Removing the Need for Ω-Structure

In previous sections, we started with the canonical definition of the decimator as in (6.8) and extended the classical multirate signal processing theory to graphs. Even though we were able to generalize a number of concepts from classical theory to graph signals, many of the results require $\mathbf{A}$ to be an $\mathbf{\Omega}$-graph. In this section we will show that this requirement can be relaxed completely under the mild assumption that $\mathbf{A}$ be diagonalizable. The basic idea is to work with a *similarity-transformed* graph matrix $\widetilde{\mathbf{A}} = \mathbf{Q}\,\mathbf{A}\,\mathbf{Q}^{-1}$ where $\mathbf{Q}$ is chosen such that $\widetilde{\mathbf{A}}$ has the $\mathbf{\Omega}$-structure as in (6.57). We will see that this is always possible. However such a transformation changes the underlying Fourier basis. Therefore, the matrix $\mathbf{Q}$ should be selected carefully, as we will do throughout this section. Given the original graph signal $\mathbf{x}$ we will then define a modified signal

$$\widetilde{\mathbf{x}} = \mathbf{Q}\,\mathbf{x}, \tag{6.127}$$

and use the modified filter bank $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ with canonical decimator (6.8) to process it (see Figure 6.21a). Then the filter bank output $\widetilde{\mathbf{y}}$ is transformed back to $\mathbf{y} = \mathbf{Q}^{-1}\widetilde{\mathbf{y}}$.

The filter bank $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ sandwiched between $\mathbf{Q}$ and $\mathbf{Q}^{-1}$ operates on the modified graph $\widetilde{\mathbf{A}}$, which satisfies the eigenvector structure (6.57). Also, it uses the standard canonical decimator. So, many of the results developed in earlier sections are applicable. We will show that the complete system from $\mathbf{x}$ to $\mathbf{y}$ in Figure 6.21a is equivalent to the graph filter bank system shown in Figure 6.21b, which operates on the original graph. Notice carefully that the canonical decimator and expander have been replaced in this system by a new decimator and expander (to be defined below in (6.130)).

The most important point is that the new filter bank $\{H_k(\widetilde{\mathbf{A}}), F_k(\widehat{\mathbf{A}})\}$ that transforms $\widetilde{\mathbf{x}}$ to $\widetilde{\mathbf{y}}$ and the original filter bank that transforms $\mathbf{x}$ to $\mathbf{y}$ have the following close relationship, as we shall show:

1. The filters $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ are polynomials in $\widetilde{\mathbf{A}}$ if and only if the original filters $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ are polynomials in $\mathbf{A}$.

2. The filter bank $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ is a perfect reconstruction system on the graph $\widetilde{\mathbf{A}}$ if and only if the original filter bank $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ is a perfect reconstruction system on the graph $\mathbf{A}$.

3. Assuming that the diagonalizable graph $\mathbf{A}$ has distinct eigenvalues, the filter bank $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ is alias-free on the graph $\widetilde{\mathbf{A}}$ if and only if the original filter bank $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ is alias-free on the graph $\mathbf{A}$.

4. With the input-output map of the filter bank with generalized decimator and expander denoted as $T_G(\mathbf{A})$ and the transformed map as $T(\widetilde{\mathbf{A}})$, they are related as in Figure 6.21c-6.21d, that is,

$$T_G(\mathbf{A}) = \mathbf{Q}^{-1}\, T(\widetilde{\mathbf{A}})\, \mathbf{Q}. \tag{6.128}$$

Notice that relations in 1, 2, and 3 follow from the following identity, which holds true for *any* invertible $\mathbf{Q}$ and for *any* polynomial $H(\cdot)$ (Theorem 6.2.9 of [86]):

$$H(\mathbf{A}) = \mathbf{Q}^{-1}\, H(\widetilde{\mathbf{A}})\, \mathbf{Q}. \tag{6.129}$$

Here $\widetilde{\mathbf{A}} = \mathbf{Q}\mathbf{A}\,\mathbf{Q}^{-1}$.

The relation in 4 follows from the following theorem.

Theorem 6.29 (Generalized filter banks)**.** *Let* $\mathbf{A}$ *be a diagonalizable adjacency matrix with the eigenvalue decomposition* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$. *Let* $\mathbf{E}$ *be an invertible matrix with the* $\mathbf{\Omega}$*-structure* (6.57) *on its columns. Set the similarity transform as* $\mathbf{Q} = \mathbf{E}\mathbf{V}^{-1}$. *Hence* $\widetilde{\mathbf{A}} = \mathbf{Q}\mathbf{A}\mathbf{Q}^{-1}$. *Define the* generalized decimator $\widetilde{\mathbf{D}}$ *and the* generalized expander $\widetilde{\mathbf{U}}$ *as*

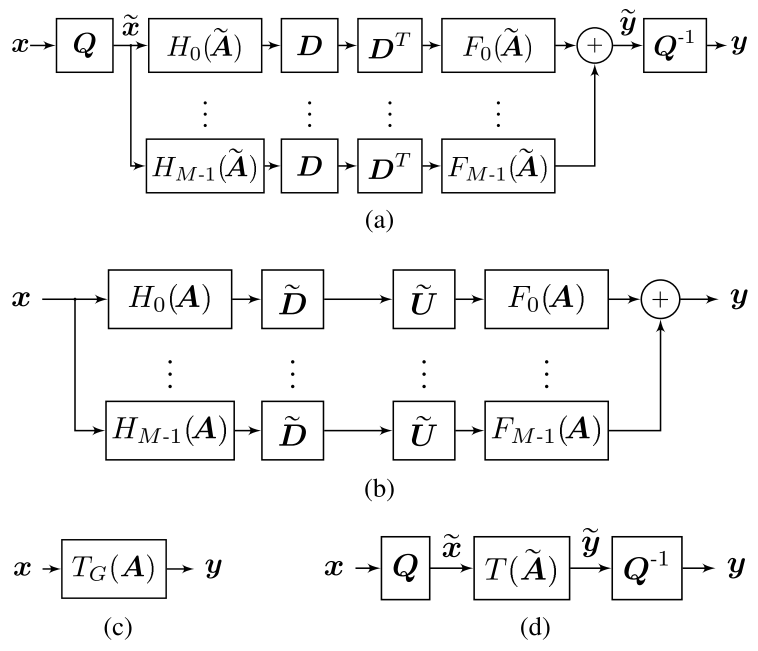
$$\widetilde{\mathbf{D}} = \mathbf{D}\,\mathbf{Q} = \mathbf{D}\,\mathbf{E}\,\mathbf{V}^{-1}, \quad \widetilde{\mathbf{U}} = \mathbf{Q}^{-1}\,\mathbf{D}^{\mathrm{T}} = \mathbf{V}\,\mathbf{E}^{-1}\,\mathbf{D}^{\mathrm{T}}. \tag{6.130}$$

Then, a FB $\{H_k(\widetilde{\mathbf{A}}), F_k(\widetilde{\mathbf{A}})\}$ *on* $\widetilde{\mathbf{A}}$ *that uses the canonical decimator and expander, Figure 6.21a, is equivalent to the generalized FB* $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ *on* $\mathbf{A}$ *that uses generalized decimator and expander, Figure 6.21b. Here the term "equivalent" means that the input-output behaviors are related as in* (6.128).

Proof. Let $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$ be the generalized FB on $\mathbf{A}$ that uses the generalized decimator and expander. Then

$$\begin{aligned} T_G(\mathbf{A}) &= \sum_{k=0}^{M-1} F_k(\mathbf{A})\,\widetilde{\mathbf{U}}\,\widetilde{\mathbf{D}}\,H_k(\mathbf{A}) = \sum_{k=0}^{M-1} \mathbf{Q}^{-1}\,\mathbf{Q}\,F_k(\mathbf{A})\,\mathbf{Q}^{-1}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{Q}\,H_k(\mathbf{A})\,\mathbf{Q}^{-1}\,\mathbf{Q}, \\ &= \mathbf{Q}^{-1} \sum_{k=0}^{M-1} F_k(\widetilde{\mathbf{A}})\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,H_k(\widetilde{\mathbf{A}})\,\mathbf{Q} = \mathbf{Q}^{-1}\,T(\widetilde{\mathbf{A}})\,\mathbf{Q}, \end{aligned} \tag{6.131}$$

where we use (6.129) in (6.131). Notice that the generalized FB implicitly operates on $\widetilde{\mathbf{A}}$, which is an $\mathbf{\Omega}$-graph since $\widetilde{\mathbf{A}} = \mathbf{Q}\,\mathbf{A}\,\mathbf{Q}^{-1} = \mathbf{E}\,\mathbf{\Lambda}\,\mathbf{E}^{-1}$. Hence, the eigenvector condition (6.57) on $\mathbf{A}$ is implicitly satisfied on the generalized FB. □



$\boldsymbol{x}$
$\boldsymbol{Q}$
$\widetilde{\boldsymbol{x}}$
$H_0(\widetilde{\boldsymbol{A}})$
$\boldsymbol{D}$
$\boldsymbol{D}^T$
$F_0(\widetilde{\boldsymbol{A}})$
$+$
$\widetilde{\boldsymbol{y}}$
$\boldsymbol{Q}^{-1}$
$\boldsymbol{y}$
$H_{M-1}(\widetilde{\boldsymbol{A}})$
$F_{M-1}(\widetilde{\boldsymbol{A}})$
(a)
$H_0(\boldsymbol{A})$
$\widetilde{\boldsymbol{D}}$
$\widetilde{\boldsymbol{U}}$
$F_0(\boldsymbol{A})$
$H_{M-1}(\boldsymbol{A})$
$F_{M-1}(\boldsymbol{A})$
(b)
$T_G(\boldsymbol{A})$
(c)
$T(\widetilde{\boldsymbol{A}})$
(d)

Figure 6.21: (a) A FB on the similarity-transformed adjacency matrix $\widetilde{\mathbf{A}}$ sandwiched between $\mathbf{Q}$ and $\mathbf{Q}^{-1}$. (b) The FB with generalized decimator and expander on the original adjacency matrix $\mathbf{A}$. (c) and (d) are input-output equivalent of the systems in terms of $\mathbf{A}$ and $\widetilde{\mathbf{A}}$, respectively. $\mathbf{Q}$ is the similarity transform. $\mathbf{D}$ is as in (6.8). $\widetilde{\mathbf{D}}$ and $\widetilde{\mathbf{U}}$ are as in (6.130). All four systems shown above are equivalent to each other in terms of input-output relations.

The identity in (6.129) basically says that instead of working on the given adjacency matrix, we can use the similarity-transformed adjacency matrix as long as the input graph signal is also transformed accordingly. As an example, consider the generalized FB. The matrix $\mathbf{Q}$ transforms the graph signal $\mathbf{x}$ into $\widetilde{\mathbf{x}}$ as shown in Figure 6.21a. Special cases of this can be found in [152] where a permutation matrix is used and in [126] where a diagonal matrix is used. Here we use it for a different purpose, namely to create a hypothetical system (the system flanked by $\mathbf{Q}$ and $\mathbf{Q}^{-1}$ in Figure 6.21a) that satisfies the eigenvector condition (6.57). This $\boldsymbol{\Omega}$-structure is sufficient (though possibly not necessary) to be able to use some of the filter banks we developed, e.g., the brickwall filter bank of Theorem 6.19, and alias free filter banks of Section 6.10. Thus the similarity transform $\mathbf{Q}$ merely sets the stage for that. As long as the similarity transform $\mathbf{Q}$ is selected properly, $\mathbf{A}$ can be treated *as if it is an* $\boldsymbol{\Omega}$*-graph* even if it is not. For example, consider the spectrum folding phenomena described in Section 6.8. When the decimator and the expander are selected as in (6.130), we can remove the condition on the eigenvectors of the adjacency matrix. We state this result as follows.

Theorem 6.30 (Spectrum folding and generalized decimation)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of a graph with the following eigenvalue decomposition* $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$*. Let* $\mathbf{E}$ *be an invertible matrix with the* $\mathbf{\Omega}$*-structure* (6.57) *on its columns. Define the generalized decimator and expander as in* (6.130)*. Let* $\mathbf{x}$ *be a signal on the graph* $\mathbf{A}$ *and* $\mathbf{y}$ *be the DU version of* $\mathbf{x}$*, that is,* $\mathbf{y} = \widetilde{\mathbf{U}}\widetilde{\mathbf{D}}\mathbf{x}$*. Then, graph Fourier transform of* $\mathbf{x}$ *and* $\mathbf{y}$ *are related as*

$$\widehat{\mathbf{y}} = \frac{1}{M}\left(\mathbf{I}_{N/M} \otimes \mathbf{1}_M \mathbf{1}_M^{\mathrm{T}}\right)\widehat{\mathbf{x}}, \tag{6.132}$$

which is nothing but the spectrum folding phenomena.

Proof. The graph Fourier transforms of $\mathbf{x}$ and $\mathbf{y}$ are given as $\widehat{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{x}$ and $\widehat{\mathbf{y}} = \mathbf{V}^{-1}\mathbf{y}$, respectively. Therefore we have

$$\mathbf{V}\,\widehat{\mathbf{y}} = \widetilde{\mathbf{U}}\,\widetilde{\mathbf{D}}\,\mathbf{V}\,\widehat{\mathbf{x}}, \tag{6.133}$$

that is,

$$\widehat{\mathbf{y}} = \mathbf{V}^{-1}\,\mathbf{V}\,\mathbf{E}^{-1}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{E}\,\mathbf{V}^{-1}\,\mathbf{V}\,\widehat{\mathbf{x}} = \mathbf{E}^{-1}\,\mathbf{D}^{\mathrm{T}}\,\mathbf{D}\,\mathbf{E}\,\widehat{\mathbf{x}} = 1/M\left(\mathbf{I}_{N/M} \otimes \mathbf{1}_M\mathbf{1}_M^{\mathrm{T}}\right)\widehat{\mathbf{x}}, \tag{6.134}$$

which follows from (6.68)-(6.74) since $\mathbf{E}$ has the $\mathbf{\Omega}$-structure. □

Even though results of Theorem 6.12 and Theorem 6.30 appear to be similar, the difference is that Theorem 6.12 applies to diagonalizable $\mathbf{\Omega}$-graphs, whereas Theorem 6.30 applies to *any* graph with diagonalizable adjacency matrix. Notice that Theorem 6.30 includes Theorem 6.12: when the eigenvectors of $\mathbf{A}$ have the $\mathbf{\Omega}$-structure, simply select $\mathbf{E} = \mathbf{V}$, the decimator and the expander then reduce to canonical form $\widetilde{\mathbf{D}} = \mathbf{D}$ and $\widetilde{\mathbf{U}} = \mathbf{D}^{\mathrm{T}}$.

The application of a similarity transform $\mathbf{Q}\mathbf{A}\mathbf{Q}^{-1}$ is therefore useful whenever the $\mathbf{\Omega}$-structure is necessary. For arbitrary eigenvectors, we need to use a similarity matrix $\mathbf{Q}$ to adjust them to the form (6.57), which, in turn, affects the choice of decimation matrix: namely the non-canonical form (6.130) that depends on $\mathbf{Q}$ has to be used. On the other hand, this technique does not alter the eigenvalues of $\mathbf{A}$. Remember from (6.79) that coefficients of a polynomial filter and its frequency response are related through the eigenvalues of the graph. Therefore, whatever $\mathbf{Q}$ we choose, a polynomial filter on $\mathbf{A}$, $H(\mathbf{A})$, and the same polynomial filter on $\widetilde{\mathbf{A}}$, $H(\widetilde{\mathbf{A}})$, have the same frequency response in their corresponding graph Fourier domains. In summary, for the desired multirate graph signal processing, the coefficients of polynomial filters should be designed according to the graph spectrum (eigenvalues of $\mathbf{A}$); the decimator and the expander should be designed according to eigenvectors of the graph.

The generalized decimator and expander in (6.130) have two important properties. Firstly, given a graph, they are not unique: we can select $\mathbf{E}$ arbitrarily as long as its columns have the $\boldsymbol{\Omega}$-property, and it is invertible. In fact, $\mathbf{E}$ can be selected as a properly permuted version of the inverse DFT matrix of size N. This follows from the fact that $\mathbf{C}_N$ is an M-block cyclic matrix for any M that divides N. Secondly, the generalized decimator *does* depend on the eigenspaces of the adjacency matrix due to presence of $\mathbf{V}$ in (6.130). Therefore, we cannot talk about a "universal" decimator, and expander, which can remove the eigenvector constraint in (6.57) for all graphs.

It might appear that the use of $\mathbf{Q}$ and $\mathbf{Q}^{-1}$ involves additional complexity of the order of N^2 in Figure 6.21a (where N is the size of the graph). This can be avoided if we implement Figure 6.21b which is equivalent. In this implementation the simple decimator $\mathbf{D}$ is replaced with $\widetilde{\mathbf{D}}$. Now, $\widetilde{\mathbf{D}}$ is an $(N/M)\times N$ matrix with possibly non-zero entries everywhere, and there are M such matrices in the figure. This gives the impression that there is additional computational overhead of N^2 multiplications. But note that $\widetilde{\mathbf{D}}$ is not unique; it is defined as $\widetilde{\mathbf{D}} = \mathbf{D}\,\mathbf{E}\,\mathbf{V}^{-1}$ where $\mathbf{E}$ is an arbitrary matrix with the $\boldsymbol{\Omega}$-structure. The degrees of freedom in $\mathbf{E}$ can be exploited to make $\widetilde{\mathbf{D}}$ relatively sparse to reduce the complexity. In fact $\widetilde{\mathbf{D}}$ can have the form $\widetilde{\mathbf{D}} = [\mathbf{I}_{N/M} \;\; \mathbf{X}]$ as shown next.

Assume that $\mathbf{E}$ has the $\boldsymbol{\Omega}$-structure on its columns. Then it can be written as

$$\mathbf{E} = \begin{bmatrix} \mathbf{E}_1\ (\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}}) \\ \vdots \\ \mathbf{E}_M\ (\mathbf{I}_{N/M} \otimes \mathbf{f}_M^{\mathrm{H}}) \end{bmatrix}, \tag{6.135}$$

for some $\mathbf{E}_k$ where $\mathbf{E}_k \in \mathbb{C}^{(N/M)\times(N/M)}$ and $\mathbf{f}_k$ is the k^{th} column of the DFT matrix of size M. Let $\mathbf{R} = \mathbf{V}^{-1}$ and write it as

$$\mathbf{R} = [\mathbf{R}_1 \ \cdots \ \mathbf{R}_M], \tag{6.136}$$

where $\mathbf{R}_k \in \mathbb{C}^{N\times N/M}$. Then we have the following for the generalized decimator

$$\widetilde{\mathbf{D}} = \mathbf{D}\,\mathbf{E}\,\mathbf{R} = \mathbf{E}_1\ (\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}})\,\mathbf{R} = [\mathbf{E}_1\ (\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}})\mathbf{R}_1 \ \cdots \ \mathbf{E}_1\ (\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}})\mathbf{R}_M]. \tag{6.137}$$

Then, select $\mathbf{E}_1$ as follows:

$$\mathbf{E}_1 = \left(\left(\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}} \right) \mathbf{R}_1 \right)^{-1}. \tag{6.138}$$

As a result we get the decimator in the form of $\widetilde{\mathbf{D}} = [\mathbf{I}_{N/M} \ \ \mathbf{X}]$ where $\mathbf{X}$ depends on $\mathbf{R}_k$'s. This proves the claim. In this construction the decimator and expander have the form

$$\widetilde{\mathbf{D}} = \mathbf{E}_1 \left(\mathbf{I}_{N/M} \otimes \mathbf{f}_1^{\mathrm{H}} \right) \mathbf{V}^{-1}. \tag{6.139}$$

$$\widetilde{\mathbf{U}} = \mathbf{V} \left(\mathbf{I}_{N/M} \otimes \mathbf{f}_1 \right) \mathbf{E}_1^{-1}. \tag{6.140}$$

At the time of this writing, we do not know how to reduce the complexities of both $\widetilde{\mathbf{D}}$ and $\widetilde{\mathbf{U}}$ simultaneously.

The similarity transform in (6.129) changes the eigenvectors, and hence the graph Fourier basis that supports the filter bank. So the transform matrix $\mathbf{Q}$ should be selected carefully, otherwise the new Fourier basis may be of no use. Notice that Figure 6.21a and 6.21b are equivalent, hence the similarity transform can be integrated into the decimator and the expander. At this point, notice the proposed decimator in (6.139) and the expander in (6.140). They explicitly *depend* on the Fourier basis of the original graph. As a result, the idea of a similarity transform reduces to a carefully designed decimator, where the decimator in (6.139) takes the Fourier basis of the original graph into account and decimates the signal accordingly. The same interpretation is valid for the expander in (6.140) as well. Moreover, the filters in Figure 6.21b are still polynomials in the original graph, hence the original graph Fourier basis can still be used to diagonalize the filter responses.

6.16 Examples

In this section, we will implement a 3-channel brickwall filter bank for the famous Minnesota road graph [76, 129, 126]. With due thanks to the authors of [129] and [76], we use the data publicly available in [128, 75]. This graph has 2642 nodes in total where 2 nodes are disconnected to the rest of the graph. Since a road graph is expected to be connected, we disregard those two nodes. See Figure 6.22 for the visual representation of the graph. The adjacency matrix is extremely sparse with only 0.1% non-zero entries. Since the edges are represented with 1's, the unit shift, $\mathbf{Ax}$, merely requires addition and no multiplication at all.

It is clear from Figure 6.22 that the graph is not 3-block cyclic. Therefore, we cannot apply Theorem 6.19 directly. However, we can use the generalized decimator and

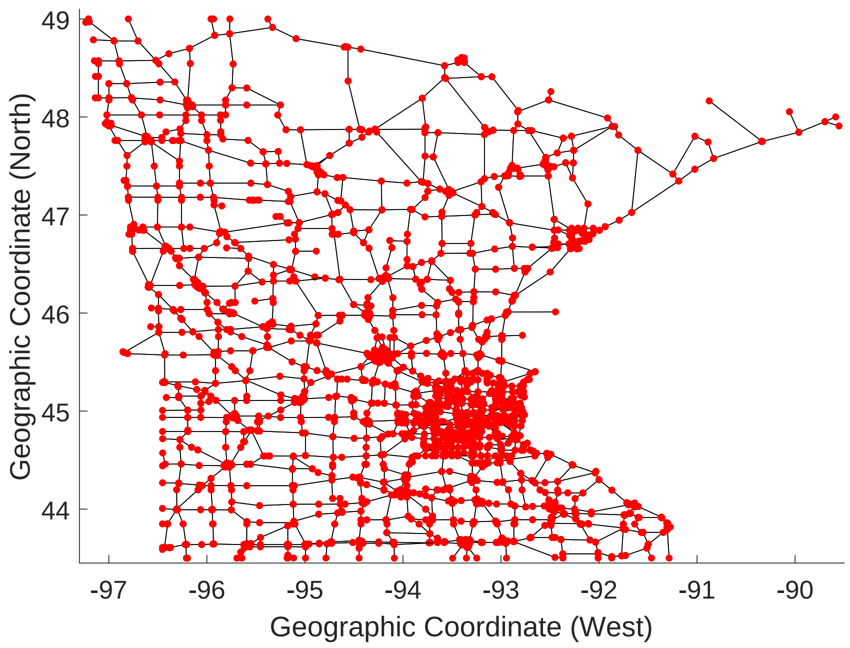


Figure 6.22: Minnesota traffic graph which has $N = 2640$ nodes, and 3302 undirected unweighted edges (courtesy of [129, 76]).

expander as explained in Section 6.15. Furthermore, we select the graph Laplacian to be the unit shift element and find the graph Fourier basis, $\mathbf{V}$, as the eigenvectors of the graph Laplacian. This selection (the Laplacian but not the adjacency matrix) is consistent with the development, since we do not put any specific meaning to the unit shift operator. We precisely use (6.139) and (6.140) to construct the generalized decimator and expander, respectively. We construct the analysis and synthesis filters as in (6.95). We denote the reconstructed output of the k^{th} channel as $\mathbf{y}_k = \mathbf{F}_k \, \widetilde{\mathbf{U}} \, \widetilde{\mathbf{D}} \, \mathbf{H}_k \, \mathbf{x}$ for $0 \leq k \leq 2$. (These are indicated in Figure 6.17b).

In the first example, we consider the signal used in [129, 126] as the filter bank input. A visual representation of this signal is given in Figure 6.23a. The reconstructed outputs from subbands, $\mathbf{y}_0$, $\mathbf{y}_1$, $\mathbf{y}_2$, are given in Figure 6.23b, 6.23c, and 6.23d, respectively. We observe that $\mathbf{y}_0$ is a smooth approximation of the original signal since it is the output of the low-pass channel. The remaining two channels give information about nodes where the discontinuity (high-frequency content) in the signal occurs.

In the second example, we select a smoother signal defined as $x_i = \exp(\text{-}0.5 \; d_i^2)$ where d_i denotes the geographic distance between the i^{th} node and the point [-93.5 45]. This signal, which is used as the filter bank input, is visualized in

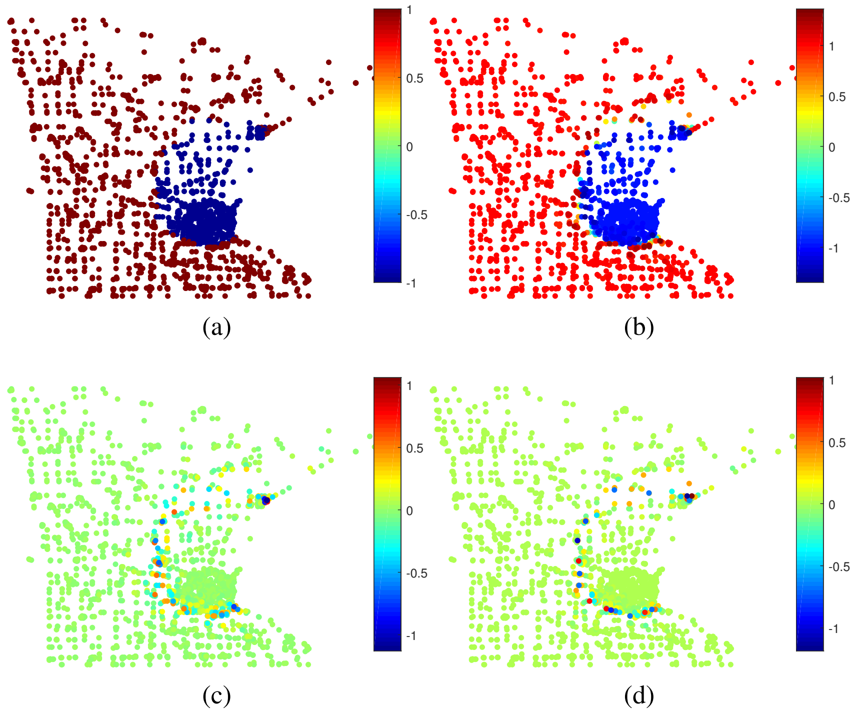


Figure 6.23: (a) Signal consisting of only 1's and -1's as given in [126], output of (b) channel-0, (c) channel-1, (d) channel-2.

Figure 6.24a. The reconstructed outputs from the channels $\mathbf{y}_0$, $\mathbf{y}_1$, and $\mathbf{y}_2$ are given in Figure 6.24b, 6.24c, and 6.24d, respectively. Similar to the previous example, the low-pass channel captures most of the energy. Even though the remaining two channels identify the nodes where change in the signal is located, outputs are not as strong as the previous example. This is due to smoother character of the signal.

The Laplacian of the Minnesota traffic graph has repeated eigenvalues. Furthermore, the distinct eigenvalues are closely spaced. As a result, we cannot implement $\{\mathbf{H}_k\}$'s as polynomials, as low order polynomial approximations result in poor performance. Hence, each filtering operation requires $N^2 \approx 7 \cdot 10^7$ multiplications. In the following, in order to obtain filters with low complexity, we will replace the brick-wall filters, $\{\mathbf{H}_k\}$, with their hard-thresholded counter-parts, $\{\mathbf{H}'_k\}$, that are constructed as follows:

$$(\mathbf{H}'_k)_{i,j} = \begin{cases} (\mathbf{H}_k)_{i,j}, & |(\mathbf{H}_k)_{i,j}| \geq \gamma \\ 0, & \text{otherwise} \end{cases} \tag{6.141}$$

for some threshold γ. It is clear that this non-linear operation on the filters compromises the PR property of the filter bank. However, it is interesting to investigate

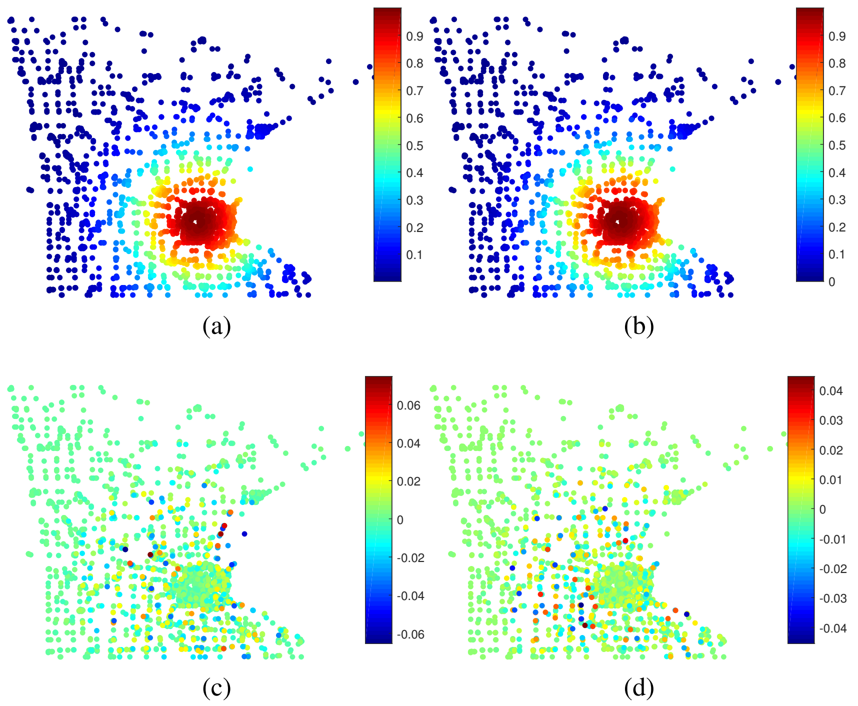


Figure 6.24: (a) Signal with exponential decay due to geographic distance, output of (b) channel-0, (c) channel-1, (d) channel-2.

the trade-off between the reconstruction error and the efficiency of the thresholded filters. For this purpose, we define the average fraction of non-zero elements of the filters as $\frac{1}{M\,N^2}\sum_{k=0}^{M\text{-}1}\left\|\mathbf{H}_k'\right\|_0$ where $\|\cdot\|_0$ counts the number of non-zeros and define the reconstruction error as $\|\mathbf{x}-\mathbf{x}'\|_2^2/\|\mathbf{x}\|_2^2$ where $\mathbf{x}'$ is the output of the filter bank with filters in (6.141). By sweeping over different thresholds, γ, we obtain the result in Figure 6.25. Here $\mathbf{x}_a$ denotes the signal in Figure 6.23a, $\mathbf{x}_b$ denotes the signal in Figure 6.24a, and $\mathbf{x}_c$ is a signal with i.i.d. Gaussian entries. These are the inputs to the filter bank in the three cases. It is very interesting to observe that when we tolerate 1% error in the reconstruction, we can sparsify the filters significantly: we only need 7.6%, 12.8% and 3.5% of the non-zero values of the actual filters, respectively. This is a huge saving, due to N^2 being very large. We do not expect hard-thresholding to be optimal, and we are not suggesting this as a filter design approach. This example merely shows that by replacing the PR property with near-PR property, very efficient filters can be designed for M-channels.

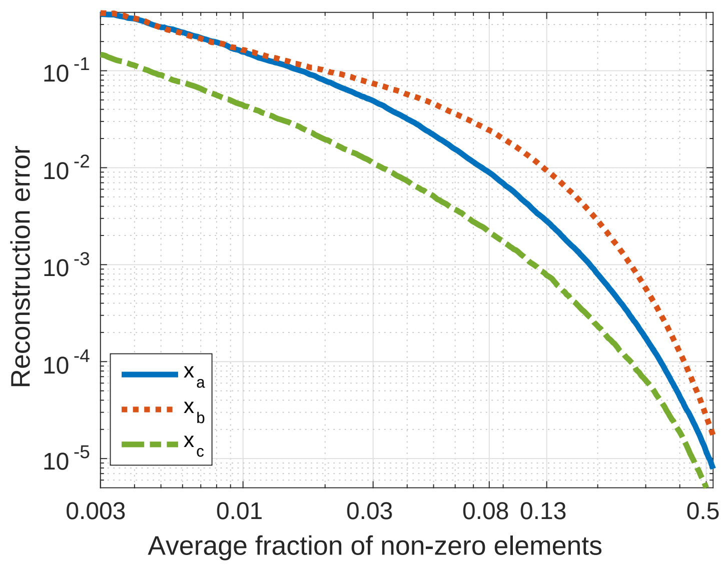


Figure 6.25: Trade-off between the efficiency of the filters and the reconstruction error. Trade-off depends on the input signal under consideration.

6.17 Concluding Remarks

In this chapter we first developed fundamental building blocks for multirate signal processing on graphs by drawing a parallel with classical multirate systems. We started with the canonical definition of the decimator and identified the corresponding expander. We then defined noble identities for graph multirate DSP. Contrary to the classical case, we showed that a certain structure needs to be imposed on the graph to establish these identities. We then studied some graphs that satisfy the conditions and defined M-block cyclic graphs in this context. The unique eigenstructure of such graphs was also shown, and the concept of spectrum folding in such graphs was thereby established. Finally we showed that alias-free systems, polynomial systems, and shift-invariant systems on graphs do not imply each other for arbitrary graphs, and established conditions under which these three concepts are equivalent.

We also extended the theory of M-channel maximally decimated filter banks, from classical time domain to the domain of graphs. There are several important aspects in which these filter banks differ from classical filter banks. We found that perfect reconstruction (PR) can be achieved with polynomial filters for graphs which satisfy certain eigenstructure conditions. Furthermore for M-block cyclic graphs, PR filter banks can be built by starting from any classical filter bank. We also developed polyphase representations for such filter banks. Even though many of the results were developed for graphs with a certain eigenstructure, it was shown in Section 6.15 that the eigenvector structure (6.57) can be relaxed, and only the eigenvalue structure (6.56) remains to be satisfied.

This also brings up some practical questions which we have not been able to address here: what are practical examples of graphs which satisfy the eigenvalue constraints? How do these filter banks perform for practical graph signals, and how do they compare with alternative ways of processing these graph signals? For example, imagine we build a compression system (akin to a subband coder) based on the PR graph filter bank with a certain order for the polynomial filters $\{H_k(\mathbf{A}), F_k(\mathbf{A})\}$. How does this compare with a brute-force compression system that performs a large DFT or a graph Fourier transform on the entire graph signal $\mathbf{x}$, performs optimal bit allocation, and reconstructs with the inverse transform? Many such practical questions remain to be addressed. In this theoretically intense chapter it has not been possible to address these important practical issues, but we plan to explore these aspects in future work.

6.18 Appendices

6.18.1 Supplementary Theorems

We now present theorems that reveal block-wise properties of M-block cyclic graphs. These theorems provide familiarity with M-block cyclic graphs.

Remember that using block-wise representation, the M-block cyclic matrix is written as

$$(\mathbf{A})_{i,j} = \mathbf{A}_j \ \delta(j\text{-}i\text{+}1). \tag{6.142}$$

For the sake of simplicity, we assume the following index convention

$$\mathbf{A}_{M+j} = \mathbf{A}_j \tag{6.143}$$

for the blocks of an M-block cyclic matrix. Then we have the following properties.

Theorem 6.31. *Let* $\mathbf{A}$ *be an M-block cyclic matrix. Then,* $(\mathbf{A}^r)_{i,j} = \mathbf{A}_{j+r\text{-}1} \cdots \mathbf{A}_j \ \delta(j\text{-}i\text{+}r)$.

Proof. Write the second power as

$$(\mathbf{A}^2)_{i,j} = \sum_{k=1}^{M} (\mathbf{A})_{i,k} \ (\mathbf{A})_{k,j} = \sum_{k=1}^{M} \mathbf{A}_k \ \delta(k\text{-}i\text{+}1) \ \mathbf{A}_j \ \delta(j\text{-}k\text{+}1) = \mathbf{A}_{j+1} \ \mathbf{A}_j \ \delta(j\text{-}i\text{+}2).$$

Then, write the third power as

$$(\mathbf{A}^3)_{i,j} = \sum_{k=1}^{M} (\mathbf{A})_{i,k}\ (\mathbf{A}^2)_{k,j} = \sum_{k=1}^{M} \mathbf{A}_k\ \delta(k\text{-}i\text{+}1)\ \mathbf{A}_{j+1}\mathbf{A}_j\ \delta(j\text{-}k\text{+}2)$$
$$= \mathbf{A}_{j+2}\ \mathbf{A}_{j+1}\ \mathbf{A}_j\ \delta(j\text{-}i\text{+}3). \tag{6.144}$$

Iterating the above steps r times, the claimed property is proved. □

Theorem 6.32. *Let* $\mathbf{A}$ *be an invertible* M*-block cyclic matrix. Then,* $(\mathbf{A}^{\text{-}r})_{i,j} = \left(\mathbf{A}_{i+r\text{-}1} \cdots \mathbf{A}_i\right)^{\text{-}1}\ \delta(i\text{-}j\text{+}r)$.

Proof.

$$(\mathbf{I}_N)_{i,j} = \sum_{k=1}^{M} (\mathbf{A}^r)_{i,k} (\mathbf{A}^{\text{-}r})_{k,j} = \sum_{k=1}^{M} \mathbf{A}_{k+r\text{-}1} \cdots \mathbf{A}_k \delta(k\text{-}i\text{+}r) \left(\mathbf{A}_{k+r\text{-}1} \cdots \mathbf{A}_k\right)^{\text{-}1}\ \delta(k\text{-}j\text{+}r)$$
$$= \mathbf{A}_{i\text{-}1} \cdots \mathbf{A}_{i\text{-}r} \left(\mathbf{A}_{j\text{-}1} \cdots \mathbf{A}_{j\text{-}r}\right)^{\text{-}1} \sum_{k=1}^{M} \delta(k\text{-}i\text{+}r)\delta(k\text{-}j\text{+}r)$$
$$= \mathbf{A}_{i\text{-}1} \cdots \mathbf{A}_{i\text{-}r} \left(\mathbf{A}_{j\text{-}1} \cdots \mathbf{A}_{j\text{-}r}\right)^{\text{-}1} \delta(j\text{-}i) = \mathbf{I}_{N/M}\ \delta(j\text{-}i). \tag{6.145}$$

□

Theorem 6.33. *Let* $\mathbf{V} \in \mathbb{C}^{N \times N}$ *and have the structure in* (6.57) *in its columns.* $\mathbf{V}$ *is invertible if and only if*

$$\left[(\mathbf{v}_{1,1})_j\ \cdots\ (\mathbf{v}_{N/M,1})_j \right] \in \mathbb{M}^{N/M} \tag{6.146}$$

is invertible for all $1 \leq j \leq M$*, where* $(\cdot)_j$ *denotes the* j^{th} *block of size* N/M*.*

Proof. Using the indexing scheme in (6.60) and relation in (6.57), we can write $\mathbf{V}$ as follows:

$$\mathbf{V} = [\mathbf{v}_{1,1}\quad \cdots \quad \mathbf{\Omega}^{M\text{-}1}\mathbf{v}_{1,1}\ \cdots \quad \mathbf{v}_{N/M,1}\quad \cdots \quad \mathbf{\Omega}^{M\text{-}1}\mathbf{v}_{N/M,1}]. \tag{6.147}$$

Let $\boldsymbol{\alpha} \in \mathbb{C}^N$ be a vector indexed as

$$\boldsymbol{\alpha} = [\alpha_{1,1}\ \cdots\ \alpha_{1,M}\ \cdots\ \alpha_{N/M,1}\ \cdots\ \alpha_{N/M,M}]^{\mathrm{T}}. \tag{6.148}$$

Assume $\mathbf{V}$ has a null-space. Then $\mathbf{V}\boldsymbol{\alpha} = \mathbf{0}$ for some $\boldsymbol{\alpha} \neq \mathbf{0}$. Notice that due to definition of $\mathbf{\Omega}$ in (6.50), we have the following relation:

$$\mathbf{\Omega}\ \mathbf{v} = \left[w^0\ (\mathbf{v})_1^{\mathrm{T}} \qquad w^{\text{-}1}\ (\mathbf{v})_2^{\mathrm{T}} \qquad \cdots \qquad w^{\text{-}(M\text{-}1)}\ (\mathbf{v})_M^{\mathrm{T}} \right]^{\mathrm{T}}, \tag{6.149}$$

where $(\mathbf{v})_i$ denotes the i^{th} block of $\mathbf{v}$ of size N/M. Using the block relation given in (6.149) and indexing in (6.148), we can write $\mathbf{V}\boldsymbol{\alpha}$ as follows:

$$\mathbf{V}\boldsymbol{\alpha} = \begin{bmatrix} \sum_{l=1}^{N/M} (\mathbf{v}_{l,1})_1 \sum_{n=1}^{M} \alpha_{l,n}\, w^{-0(n-1)} \\ \sum_{l=1}^{N/M} (\mathbf{v}_{l,1})_2 \sum_{n=1}^{M} \alpha_{l,n}\, w^{-(n-1)} \\ \vdots \\ \sum_{l=1}^{N/M} (\mathbf{v}_{l,1})_M \sum_{n=1}^{M} \alpha_{l,n}\, w^{-(M-1)(n-1)} \end{bmatrix} = \mathbf{0}_M. \tag{6.150}$$

Define $f_l(k) = \sum_{n=1}^{M} \alpha_{l,n}\, w^{-(k-1)(n-1)}$, that is, $f_l(k)$ is the k^{th} value of inverse DFT of the sequence $[\alpha_{l,1} \cdots \alpha_{l,M}]$. Then we have the following:

$$\left[(\mathbf{v}_{1,1})_j \ \cdots \ (\mathbf{v}_{N/M,1})_j\right] \left[f_1(j) \ \cdots \ f_{N/M}(j)\right]^{\mathrm{T}} = \mathbf{0}_{N/M}, \tag{6.151}$$

for all $1 \leq j \leq M$. Due to the DFT relation, not all $f_l(k)$ are zero since $\boldsymbol{\alpha} \neq \mathbf{0}$. Therefore, there exists j such that $[f_1(j) \cdots f_{N/M}(j)] \neq \mathbf{0}$ but (6.151) is zero. Hence, $[(\mathbf{v}_{1,1})_j \cdots (\mathbf{v}_{N/M,1})_j]$ has a null-space.

Conversely, assume that $[(\mathbf{v}_{1,1})_j \cdots (\mathbf{v}_{N/M,1})_j]$ has a null-space for some j. Therefore we have $[f_1(j) \cdots f_{N/M}(j)] \neq \mathbf{0}$ but (6.151) is satisfied. By selecting remaining $f_l(k)$ to be zero, we have $\mathbf{V}\boldsymbol{\alpha} = \mathbf{0}$ but $\boldsymbol{\alpha} \neq \mathbf{0}$, that is $\mathbf{V}$ has a null-space.

Therefore, $\mathbf{V}$ has a null space if and only if (6.146) has a null space for some $1 \leq j \leq M$. That is to say, $\mathbf{V}$ does not have a null space if and only if (6.146) does not have a null space for all $1 \leq j \leq M$, which is equivalent to the statement of the theorem. □

Corollary 6.3. *Let* $\mathbf{A}$ *be an* M*-block cyclic matrix.* $\mathbf{A}$ *is invertible if and only if* $\mathbf{A}_j$ *are invertible for* $1 \leq j \leq M$.

Proof. Let $r = 1$ in Theorem 6.32. Therefore, inverse of $\mathbf{A}$ consists of inverse of $\mathbf{A}_j$'s as blocks. Existence of inverse of $\mathbf{A}$ implies existence of inverses of $\mathbf{A}_j$'s and vice versa. □

Corollary 6.4. *Let* $\mathbf{A}$ *be an* M*-block cyclic matrix. Then,* $\mathbf{A}^M$ *is a block diagonal matrix.*

Proof. Let $r = M$ in Theorem 6.31. Then, $(\mathbf{A}^M)_{i,j} = \mathbf{A}_{j+M-1} \cdots \mathbf{A}_j\ \delta(j-i+M)$. Therefore $(\mathbf{A}^M)_{i,j} = \mathbf{0}$ when $i \neq j$ since $\delta(\cdot)$ is periodic with M. □

Corollary 6.5. *Let* $\mathbf{A}$ *be an* M*-block cyclic matrix. Then adjusted shift operator* $\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}^M\,\mathbf{D}^{\mathrm{T}}$ *has the form* $\bar{\mathbf{A}} = \mathbf{A}_M \cdots \mathbf{A}_1$.

Proof. Remember from Theorems 6.1 and 6.2, the adjusted shift operator is defined as $\bar{\mathbf{A}} = (\mathbf{A}^M)_{1,1}$. In Theorem 6.31, set $r = M$, $i = 1$ and $j = 1$. Then we have $\bar{\mathbf{A}} = \mathbf{A}_M \cdots \mathbf{A}_1$. □

Corollary 6.6. *Let* $\mathbf{A}$ *be an* M*-block cyclic matrix. Then,* $\mathbf{A}^{-r}\,\mathbf{D}^{\mathrm{T}}$ *is a block-column vector with only one non-zero block at index* $M+1-r$*. More precisely,* $(\mathbf{A}^{-r}\,\mathbf{D}^{\mathrm{T}})_i = \left(\mathbf{A}_M \cdots \mathbf{A}_{M+1-r}\right)^{-1} \delta(i-1+r)$ *for* $1 \le i \le M$ *and* $r \ge 0$.

Proof. Remember that $(\mathbf{D}^{\mathrm{T}})_k = \mathbf{I}_{N/M}\ \delta(k-1)$. Therefore,

$$\begin{aligned}(\mathbf{A}^{-r}\,\mathbf{D}^{\mathrm{T}})_i &= \sum_{k=1}^{M} (\mathbf{A}^{-r})_{i,k} (\mathbf{D}^{\mathrm{T}})_k = \sum_{k=1}^{M} \left(\mathbf{A}_{i+r-1} \cdots \mathbf{A}_i\right)^{-1} \delta(i-k+r)\, \mathbf{I}_{N/M}\, \delta(k-1) \\ &= \left(\mathbf{A}_M \cdots \mathbf{A}_{M+1-r}\right)^{-1} \delta(i-1+r) \qquad (6.152)\end{aligned}$$

□

Corollary 6.7. *Let* $\mathbf{A}$ *be an* M*-block cyclic matrix. Then,* $\mathbf{D}\,\mathbf{A}^r$ *is a block-row vector with only one non-zero block at index* $M+1-r$*. More precisely,* $(\mathbf{D}\,\mathbf{A}^r)_i = \mathbf{A}_M \cdots \mathbf{A}_{M+1-r}\ \delta(i-1+r)$ *for* $1 \le i \le M$ *and* $r \ge 0$.

Proof. Remember that $(\mathbf{D})_k = \mathbf{I}_{N/M}\ \delta(k-1)$. Therefore,

$$\begin{aligned}(\mathbf{D}\,\mathbf{A}^r)_i &= \sum_{k=1}^{M} (\mathbf{D})_k (\mathbf{A}^r)_{k,i} = \sum_{k=1}^{M} \mathbf{I}_{N/M}\ \delta(k-1) \mathbf{A}_{i+r-1} \cdots \mathbf{A}_i\, \delta(i-k+r) \\ &= \mathbf{A}_M \cdots \mathbf{A}_{M+1-r}\ \delta(i-1+r) \qquad (6.153)\end{aligned}$$

□

6.18.2 Proof of Theorem 6.10

Before starting to prove Theorem 6.10, we state the following lemma, which will be useful in the proof.

Lemma 6.1. *Let* $\mathbf{A}$ *be written as:*

$$\mathbf{A} = \sum_{k=0}^{M-1} w^k\, \boldsymbol{\Omega}^k\, \mathbf{v}\, \mathbf{u}\, \boldsymbol{\Omega}^{-k}, \qquad (6.154)$$

where w *is in* (6.49) *and* $\boldsymbol{\Omega}$ *is in* (6.50)*. Then* $\mathbf{A}$ *is a balanced* M*-block cyclic matrix for any column vector* $\mathbf{v}$ *and row vector* $\mathbf{u}$.

Proof. Let $\mathbf{v} = [(\mathbf{v})_1^{\mathrm{T}} \ \cdots \ (\mathbf{v})_M^{\mathrm{T}}]^{\mathrm{T}}$ and $\mathbf{u} = [(\mathbf{u})_1 \ \cdots \ (\mathbf{u})_M]$, where $(\mathbf{v})_i$ and $(\mathbf{u})_i$ are the i^{th} block of size N/M of $\mathbf{v}$ and $\mathbf{u}$, respectively. Then we have

$$(\mathbf{A})_{l,s} = \sum_{k=0}^{M\text{-}1} w^k \, w^{\text{-}(l\text{-}1)k} \, \mathbf{v}_l \, \mathbf{u}_s \, w^{(s\text{-}1)k} = \mathbf{v}_l \, \mathbf{u}_s \sum_{k=0}^{M\text{-}1} w^{(s\text{-}l+1)k} = \mathbf{v}_l \, \mathbf{u}_s \, M \, \delta(s\text{-}l+1). \tag{6.155}$$

Hence we have $(\mathbf{A})_{l,s} = \mathbf{v}_l \, \mathbf{u}_s \, M \, \delta(s\text{-}l+1)$, which is the definition of a balanced M-block cyclic matrix due to (6.47). □

Theorem 6.9 proves that structure in (6.56) and (6.57) of exist if the matrix is diagonalizable and M-block cyclic.

For the reverse direction, assume that structure in (6.56) and (6.57) exists with the indexing scheme in (6.59) and (6.60). To be consistent with this indexing, let $\mathbf{r}_{i,j} \in \mathbb{C}^{1 \times N}$ denote the rows of $\mathbf{V}^{\text{-}1}$. That is,

$$\mathbf{V}^{\text{-}1} = \begin{bmatrix} \mathbf{r}_{1,1} \\ \mathbf{r}_{1,M} \\ \vdots \\ \mathbf{r}_{N/M,M} \\ \mathbf{r}_{N/M,M} \end{bmatrix}. \tag{6.156}$$

Then, we have the following relation between the rows of $\mathbf{V}^{\text{-}1}$

$$\mathbf{r}_{i,\,j+k} = \mathbf{r}_{i,\,j} \, \boldsymbol{\Omega}^{\text{-}k} \tag{6.157}$$

for all $1 \leq i \leq N/M$ and $1 \leq j \leq M$. To see this, remember $\mathbf{V}^{\text{-}1}\mathbf{V} = \mathbf{I}$. That is $\mathbf{r}_{i,1} \, \mathbf{v}_{j,a} = \delta(j\text{-}i)\delta(a\text{-}1)$, or we can say that $\mathbf{r}_{i,1} \, \boldsymbol{\Omega}^{a\text{-}1}\mathbf{v}_{j,1} = \delta(j\text{-}i)\delta(a\text{-}1)$. Letting $a = k\text{-}l+1$, we have $\mathbf{r}_{i,1}\boldsymbol{\Omega}^{\text{-}l+1} \, \boldsymbol{\Omega}^{k\text{-}1}\mathbf{v}_{j,1} = \delta(j\text{-}i)\delta(k\text{-}l)$. When we assume that (6.157) is correct, we get $\mathbf{r}_{i,l} \, \mathbf{v}_{j,k} = \delta(j\text{-}i)\delta(k\text{-}l)$, which also means $\mathbf{V}^{\text{-}1}\mathbf{V} = \mathbf{I}$. Since inverse of a matrix is unique, rows of $\mathbf{V}^{\text{-}1}$ satisfy (6.157).

Since $\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{\text{-}1}$ gives the matrix $\mathbf{A}$, we have the following decomposition for $\mathbf{A}$:

$$\mathbf{A} = \sum_{l=1}^{N/M} \sum_{k=1}^{M} \lambda_{l,k} \, \mathbf{v}_{l,k} \, \mathbf{r}_{l,k} = \sum_{l=1}^{N/M} \lambda_{l,1} \sum_{k=0}^{M\text{-}1} w^k \, \boldsymbol{\Omega}^k \, \mathbf{v}_{l,1} \, \mathbf{r}_{l,1} \, \boldsymbol{\Omega}^{\text{-}k}. \tag{6.158}$$

Due to Lemma 6.1 from above, inner summation over k produces an M-block cyclic matrix. Therefore $\mathbf{A}$ is a weighted sum of M-block cyclic matrices, which is also an M-block cyclic matrix.

6.18.3 Proof of Theorem 6.23

Let $\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{-1}$ be the eigenvalue decomposition of the matrix $\mathbf{A}$. Assume that eigenvectors in $\mathbf{V}$ are indexed as in (6.60) and eigenvalues in $\mathbf{\Lambda}$ are indexed as in (6.59). Assume $\mathbf{V}$ has the structure in (6.57) and let eigenvalues be arbitrary but distinct. Let $\mathbf{r}_{i,j} \in \mathbb{C}^{1\times N}$ denote the rows of $\mathbf{V}^{-1}$. That is,

$$\mathbf{V}^{-1} = \begin{bmatrix} \mathbf{r}_{1,1} \\ \mathbf{r}_{1,M} \\ \vdots \\ \mathbf{r}_{N/M,M} \\ \mathbf{r}_{N/M,M} \end{bmatrix}. \tag{6.159}$$

Then, we have the following relation between the rows of $\mathbf{V}^{-1}$

$$\mathbf{r}_{i,j+k} = \mathbf{r}_{i,j}\,\mathbf{\Omega}^{-k} \tag{6.160}$$

for all $1 \leq i \leq N/M$ and $1 \leq j \leq M$. To see this, remember $\mathbf{V}^{-1}\mathbf{V} = \mathbf{I}$. That is $\mathbf{r}_{i,1}\,\mathbf{v}_{j,a} = \delta(j\text{-}i)\delta(a\text{-}1)$, or we can say that $\mathbf{r}_{i,1}\,\mathbf{\Omega}^{a-1}\mathbf{v}_{j,1} = \delta(j\text{-}i)\delta(a\text{-}1)$. Letting $a = k\text{-}l\text{+}1$, we have $\mathbf{r}_{i,1}\mathbf{\Omega}^{-l+1}\,\mathbf{\Omega}^{k-1}\mathbf{v}_{j,1} = \delta(j\text{-}i)\delta(k\text{-}l)$. When we assume that (6.160) is correct, we get $\mathbf{r}_{i,l}\,\mathbf{v}_{j,k} = \delta(j\text{-}i)\delta(k\text{-}l)$, which also means $\mathbf{V}^{-1}\mathbf{V} = \mathbf{I}$. Since inverse of a matrix is unique, rows of $\mathbf{V}^{-1}$ satisfy (6.159).

The M^{th} power of $\mathbf{A}$ can be written as $\mathbf{A}^M = \mathbf{V}\,\mathbf{\Lambda}^M\,\mathbf{V}^{-1}$. Therefore, we can expand $\mathbf{A}^M$ as follows:

$$\mathbf{A}^M = \sum_{l=1}^{N/M}\sum_{k=1}^{M}\lambda_{l,k}^M\,\mathbf{v}_{l,k}\,\mathbf{r}_{l,k} = \sum_{l=1}^{N/M}\sum_{k=1}^{M}\lambda_{l,k}^M\,\mathbf{\Omega}^{k-1}\,\mathbf{v}_{l,1}\,\mathbf{r}_{l,1}\,\mathbf{\Omega}^{-k+1}, \tag{6.161}$$

where $\mathbf{r}_{l,k}$'s denote the rows of $\mathbf{V}^{-1}$, are indexed as in (6.159) and have the property in (6.160). With this decomposition, $(i,j)^{th}$ block of $\mathbf{A}^M$ can be written as

$$\begin{aligned}(\mathbf{A}^M)_{i,j} &= \sum_{l=1}^{N/M}\sum_{k=1}^{M}\lambda_{l,k}^M\,w^{-(i-1)(k-1)}\,(\mathbf{v}_{l,1})_i\,(\mathbf{r}_{l,1})_j\,w^{(j-1)(k-1)}, \\ &= \sum_{l=1}^{N/M}(\mathbf{v}_{l,1})_i\,(\mathbf{r}_{l,1})_j\sum_{k=1}^{M}\lambda_{l,k}^M\,w^{-(i-j)(k-1)}. \end{aligned} \tag{6.162}$$

Further assume that $\mathbf{A}^M$ satisfies (6.27). Then we have $(\mathbf{A}^M)_{1,j} = (\mathbf{A}^M)_{i,1} = \mathbf{0}$ for all $2 \leq i,j \leq M$. Therefore, our purpose is to find $\lambda_{l,k}$ such that (6.162) satisfies the following:

$$(\mathbf{A}^M)_{1,j} = \sum_{l=1}^{N/M}(\mathbf{v}_{l,1})_1\,(\mathbf{r}_{l,1})_j\sum_{k=1}^{M}\lambda_{l,k}^M\,w^{(j-1)(k-1)} = \mathbf{0}, \tag{6.163}$$

for $2 \leq j \leq M$. Define $f_l(j)$ as follows:

$$f_l(j) = \sum_{k=1}^{M} \lambda_{l,k}^{M} \, w^{(j\text{-}1)(k\text{-}1)}. \tag{6.164}$$

Notice that $f_l(j)$ corresponds to j^{th} value of the DFT of the sequence $[\lambda_{l,1}^{M} \, \cdots \, \lambda_{l,M}^{M}]$. Then we can write (6.163) as

$$[(\mathbf{v}_{1,1})_1 \quad \cdots \quad (\mathbf{v}_{N/M,1})_1] \begin{bmatrix} f_1(j) & & \\ & \ddots & \\ & & f_{N/M}(j) \end{bmatrix} \begin{bmatrix} (\mathbf{r}_{1,1})_j \\ \vdots \\ (\mathbf{r}_{N/M,1})_j \end{bmatrix} = \mathbf{0},$$

for $2 \leq j \leq M$. Since the inverse of $\mathbf{V}$ exists by its definition, above matrices do not have null-spaces due to Theorem 6.33 in Section 6.18.1. Therefore, the only solution to the above equation is $f_l(j) = 0$ for $2 \leq j \leq M$ and for all $1 \leq l \leq N/M$. Furthermore, we can solve for $\lambda_{l,k}^{M}$ by taking inverse DFT of $f_l(j)$. Since $f_l(1)$ is allowed to have an arbitrary value, we get the following solution:

$$\lambda_{l,k}^{M} = c_l, \tag{6.165}$$

for some $c_l \in \mathbb{C}$ for $1 \leq k \leq M$. Notice that (6.165) also yields $(\mathbf{A}^M)_{i,1} = \mathbf{0}$ for $2 \leq i \leq M$. Since eigenvalues are assumed to be distinct, we get the following solution set:

$$\lambda_{l,k} = c_l \, w^{S(k)}, \tag{6.166}$$

where $S(k) \in \{1, \cdots, M\}$ and $S(k_1) \neq S(k_2)$ for $k_1 \neq k_2$ for some c_l. Notice that when $S(k) = k$, we get the eigenvalue structure in (6.56), yet this is not the only solution. Therefore the eigenvalue structure of an M-block cyclic matrix in (6.56) implies (6.166), whereas (6.166) does not imply (6.56) since the later one requires a particular ordering in the eigenvalues. More precisely, the only difference between $\mathbf{A}$ and an M-block cyclic matrix is the order of the eigenvalues associated with the eigenvectors. Using a permutation matrix $\mathbf{\Pi}$, we can re-order the eigenvalues in the following way $\mathbf{\Pi\Lambda\Pi}^{\text{-}1}$. Since we can select $\mathbf{\Pi}$ to satisfy the specific ordering required by (6.56), we can construct an M-block cyclic matrix as $\mathbf{V\Pi\Lambda\Pi}^{\text{-}1}\mathbf{V}^{\text{-}1}$.

As a result, $\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{\text{-}1}$ may not be an M-block cyclic matrix, but $\mathbf{V\Pi\Lambda\Pi}^{\text{-}1}\mathbf{V}^{\text{-}1}$ is. Hence, we conclude that $\mathbf{A}$ is *similar* to an M-block cyclic matrix.

6.18.4 Proof of Theorem 6.25

For the sake of simplicity, in the following we will use $\mathbf{P}_{i,j}$ to denote the $(i, j)^{th}$ block of the matrix $\mathbf{P}(\bar{\mathbf{A}})$. That is, $\mathbf{P}_{i,j} = (\mathbf{P}(\bar{\mathbf{A}}))_{i,j} \in \mathbb{C}^{(N/M)\times(N/M)}$.

Assuming that $\mathbf{A}$ is invertible, overall response of the filter bank in Figure 6.19b is written as

$$T(\mathbf{A}) = \sum_{i,j=1}^{M} \mathbf{A}^{M\text{-}i}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\, \mathbf{A}^{j\text{-}1} = \mathbf{A}^{M\text{-}1}\, T'(\mathbf{A}), \tag{6.167}$$

where we define

$$T'(\mathbf{A}) = \sum_{i,j=1}^{M} \mathbf{A}^{\text{-}(i\text{-}1)}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\, \mathbf{A}^{j\text{-}1}. \tag{6.168}$$

Now consider the $(s,k)^{th}$ block of both sides of (6.167) where $1 \le s,k \le M$. We have

$$\begin{aligned}\big(T'(\mathbf{A})\big)_{s,k} &= \left(\sum_{i,j=1}^{M} \mathbf{A}^{\text{-}(i\text{-}1)}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\, \mathbf{A}^{j\text{-}1}\right)_{s,k} = \sum_{i,j=1}^{M} \left(\mathbf{A}^{\text{-}(i\text{-}1)}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\, \mathbf{A}^{j\text{-}1}\right)_{s,k} \\ &= \sum_{i,j,l,r=1}^{M} \big(\mathbf{A}^{\text{-}(i\text{-}1)}\big)_{s,l} \big(\mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\big)_{l,r} \big(\mathbf{A}^{j\text{-}1}\big)_{r,k}. \end{aligned} \tag{6.169}$$

Due to definition of $\mathbf{D}$ from (6.8), we have

$$\big(\mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\big)_{l,r} = \mathbf{P}_{i,j}\, \delta(l\text{-}1)\, \delta(r\text{-}1), \tag{6.170}$$

that is, only the $(1,1)^{th}$ block of $\mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}$ is non-zero. Therefore we can simplify (6.169) as follows:

$$\big(T'(\mathbf{A})\big)_{s,k} = \sum_{i,j=1}^{M} \big(\mathbf{A}^{\text{-}(i\text{-}1)}\big)_{s,1}\, \mathbf{P}_{i,j}\, \big(\mathbf{A}^{j\text{-}1}\big)_{1,k}. \tag{6.171}$$

Now assume that the filter bank provides perfect reconstruction, that is, overall response of the filter bank is

$$T(\mathbf{A}) = \mathbf{A}^{M\text{-}1+Mm+n}, \tag{6.172}$$

for some integer $0 \le n \le M\text{-}1$ and integer m. Then we have

$$T'(\mathbf{A}) = \mathbf{A}^{Mm+n}. \tag{6.173}$$

Using Theorem 6.31, left hand side of (6.171) can be expanded as follows:

$$\big(T'(\mathbf{A})\big)_{s,k} = \Big(\mathbf{A}_{k+Mm+n\text{-}1} \cdots \mathbf{A}_{k}\Big)\, \delta(k\text{-}s\text{+}n). \tag{6.174}$$

Using Theorem 6.31 and 6.32, right hand side of (6.171) can be expanded as follows:

$$\sum_{i,j=1}^{M} (\mathbf{A}_{s+i\text{-}2} \cdots \mathbf{A}_{s})^{\text{-}1}\, \delta(s\text{+}i\text{-}2)\, \mathbf{P}_{i,j}\, (\mathbf{A}_{k+j\text{-}2} \cdots \mathbf{A}_{k})\, \delta(k\text{+}j\text{-}2). \tag{6.175}$$

In order to have a perfect reconstruction system, we need (6.174) and (6.175) to be equal to each other. In (6.175) notice that, for a given (s, k), there is only one (i, j) pair that have a non-zero contribution due to presence of $\delta(\cdot)$ terms. For a given (s, k), we need to consider 4 different cases.

Case 1: $s = 1$, $k = 1$. In this case, only non-zero term in (6.175) is when $i = 1$ and $j = 1$. When we equate (6.174) and (6.175), we get

$$\left(\mathbf{A}_{Mm+n} \cdots \mathbf{A}_1\right) \delta(n) = \mathbf{P}_{1,1}. \tag{6.176}$$

This is equivalent to have

$$\mathbf{P}_{1,1} = \bar{\mathbf{A}}^m \, \delta(n), \tag{6.177}$$

where we used the fact that $\bar{\mathbf{A}} = \mathbf{D}\,\mathbf{A}\,\mathbf{D}^{\mathrm{T}} = \mathbf{A}_M \cdots \mathbf{A}_1$ for M-block cyclic matrix and $\mathbf{A}_{M+j} = \mathbf{A}_j$ for any j for the blocks of an M-block cyclic matrix as in (6.143).

Case 2: $s \geq 2$, $k = 1$. In this case, only non-zero term in (6.175) is when $i = M$+2-s and $j = 1$. When we equate (6.174) and (6.175), we get

$$\left(\mathbf{A}_{Mm+n} \cdots \mathbf{A}_1\right) \delta(1\text{-}s\text{+}n) = \left(\mathbf{A}_M \cdots \mathbf{A}_s\right)^{-1} \mathbf{P}_{M+2-s,1}. \tag{6.178}$$

In (6.178), the only non-zero solution happens when $s = n+1$. However, $n = 0$ results in $s = 1$ which falls outside of the scope of this case. Hence we get the following for $n \geq 1$:

$$\begin{aligned}\mathbf{P}_{M+1-n,1} &= (\mathbf{A}_M \cdots \mathbf{A}_{n+1}) \left(\mathbf{A}_{Mm+n} \cdots \mathbf{A}_1\right) = (\mathbf{A}_{Mm+M} \cdots \mathbf{A}_{Mm+n+1})\left(\mathbf{A}_{Mm+n} \cdots \mathbf{A}_1\right) \\ &= \left(\mathbf{A}_{M(m+1)} \cdots \mathbf{A}_1\right) = \bar{\mathbf{A}}^{m+1}.\end{aligned}$$

Hence we have

$$\mathbf{P}_{M+1-n,1} = \bar{\mathbf{A}}^{m+1}, \qquad M\text{-}1 \geq n \geq 1. \tag{6.179}$$

Case 3: $s = 1$, $k \geq 2$. In this case, only non-zero term in (6.175) is when $i = 1$ and $j = M$+2-k. When we equate (6.174) and (6.175), we get

$$\left(\mathbf{A}_{k+Mm+n-1} \cdots \mathbf{A}_k\right) \delta(k\text{-}1\text{+}n) = \mathbf{P}_{1,M+2-k} \, (\mathbf{A}_M \cdots \mathbf{A}_k). \tag{6.180}$$

In (6.180), the only non-zero solution happens when $k = M$+1-n. However, $n = 0$ results in $k = M+1$ which falls outside of the range of k. Hence we get the following for $n \geq 1$:

$$\begin{aligned}\mathbf{P}_{1,n+1} &= \left(\mathbf{A}_{M(m+1)} \cdots \mathbf{A}_{M+1-n}\right)(\mathbf{A}_M \cdots \mathbf{A}_{M+1-n})^{-1} \\ &= \left(\mathbf{A}_{M(m+1)} \cdots \mathbf{A}_{M+1}\right) = \left(\mathbf{A}_{Mm} \cdots \mathbf{A}_1\right) = \bar{\mathbf{A}}^m.\end{aligned} \tag{6.181}$$

Even though this case excludes $n = 0$, from Case 1, we have $\mathbf{P}_{1,1} = \bar{\mathbf{A}}^m$ for $n = 0$. Therefore we can assume that (6.181) is valid for all n. Hence we have,

$$\mathbf{P}_{1,1+n} = \bar{\mathbf{A}}^m, \qquad M\text{-}1 \geq n \geq 0. \tag{6.182}$$

Case 4: $s \geq 2$, $k \geq 2$. In this case, only non-zero term in (6.175) is when $i = M\text{+}2\text{-}s$ and $j = M\text{+}2\text{-}k$. When we equate (6.174) and (6.175), we get

$$\left(\mathbf{A}_{k+Mm+n\text{-}1}\cdots\mathbf{A}_k\right)\delta(k\text{-}s\text{+}n) = \left(\mathbf{A}_M\cdots\mathbf{A}_s\right)^{\text{-}1}\mathbf{P}_{M+2\text{-}s,M+2\text{-}k}\left(\mathbf{A}_M\cdots\mathbf{A}_k\right). \tag{6.183}$$

In (6.183), notice that the index s and the index k are inter-related unlike the previous cases. Therefore, we need to consider two special sub-cases:

Case 4.1: $s \geq 2$, $M\text{-}n \geq k \geq 2$. In (6.183), the only non-zero solution happens when $s = k\text{+}n$. Hence we get

$$\begin{aligned}\mathbf{P}_{M+2\text{-}k\text{-}n,M+2\text{-}k} &= (\mathbf{A}_M\cdots\mathbf{A}_{k+n})\left(\mathbf{A}_{k+Mm+n\text{-}1}\cdots\mathbf{A}_k\right)(\mathbf{A}_M\cdots\mathbf{A}_k)^{\text{-}1}\\ &= (\mathbf{A}_{Mm+M}\cdots\mathbf{A}_{Mm+k+n})\left(\mathbf{A}_{k+Mm+n\text{-}1}\cdots\mathbf{A}_k\right)(\mathbf{A}_M\cdots\mathbf{A}_k)^{\text{-}1}\\ &= (\mathbf{A}_{M(m+1)}\cdots\mathbf{A}_{M+1}) = \bar{\mathbf{A}}^m.\end{aligned} \tag{6.184}$$

With a proper change of index variables we have

$$\mathbf{P}_{k,k+n} = \bar{\mathbf{A}}^m, \qquad 2 \leq k \leq M\text{-}n. \tag{6.185}$$

Case 4.2: $s \geq 2$, $M \geq k \geq M\text{-}n\text{+}2$. In (6.183), the only non-zero solution happens when $s = k\text{+}n\text{-}M$. Notice that in this case we exclude $k = M\text{-}n\text{+}1$ since it results in $s = 1$ and it has already been investigated in Case 1 and 3. Hence we get

$$\begin{aligned}\mathbf{P}_{2M+2\text{-}k\text{-}n,M+2\text{-}k} &= (\mathbf{A}_M\cdots\mathbf{A}_{k+n\text{-}M})\left(\mathbf{A}_{k+Mm+n\text{-}1}\cdots\mathbf{A}_k\right)(\mathbf{A}_M\cdots\mathbf{A}_k)^{\text{-}1}\\ &= (\mathbf{A}_{M(m+2)}\cdots\mathbf{A}_{Mm+k+n})\left(\mathbf{A}_{k+Mm+n\text{-}1}\cdots\mathbf{A}_k\right)(\mathbf{A}_M\cdots\mathbf{A}_k)^{\text{-}1}\\ &= (\mathbf{A}_{M(m+2)}\cdots\mathbf{A}_{M+1}) = \bar{\mathbf{A}}^{m+1}.\end{aligned} \tag{6.186}$$

With a proper change of index variables we have

$$\mathbf{P}_{k,k+n\text{-}M} = \bar{\mathbf{A}}^{m+1}, \qquad M\text{-}n\text{+}2 \leq k \leq M. \tag{6.187}$$

When we combine Case 4.1 in (6.185) with Case 3 in (6.182) and combine Case 4.2 in (6.187) with Case 2 in (6.179), we get the following relation for the blocks of the polyphase transfer matrix

$$\begin{aligned}\mathbf{P}_{k,\,k+n} &= \bar{\mathbf{A}}^m, \qquad && 1 \leq k \leq M\text{-}n,\\ \mathbf{P}_{k,\,k+n\text{-}M} &= \bar{\mathbf{A}}^{m+1}, \qquad && M\text{-}n\text{+}1 \leq k \leq M.\end{aligned} \tag{6.188}$$

Notice that (6.188) gives all the non-zero blocks of the polyphase transfer matrix. Blocks that are not specified in (6.188) are zero. As a result, we can represent (6.188) in a compact manner as follows:

$$P(\bar{\mathbf{A}}) = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{M-n} \otimes \bar{\mathbf{A}}^{m} \\ \mathbf{I}_{n} \otimes \bar{\mathbf{A}}^{m+1} & \mathbf{0} \end{bmatrix}. \qquad (6.189)$$

The given form of $P(\bar{\mathbf{A}})$ in (6.118) is achieved when the Kronecker product with $\bar{\mathbf{A}}^{m}$ in (6.189) is carried out. As a result, for an invertible M-block cyclic $\mathbf{A}$, when the overall response of the filter bank is $\mathbf{A}^{M-1+Mm+n}$ for some integer m and integer $0 \leq n \leq M$-1, then $P(\bar{\mathbf{A}})$ necessarily have the form in (6.118).

For the converse direction, assume that the polyphase transfer matrix has the form in (6.118), which is equivalent to having (6.188). Then, we can write the overall response of the filter bank as

$$T(\mathbf{A}) = \sum_{i,j=1}^{M} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{P}_{i,j}\, \mathbf{D}\, \mathbf{A}^{j-1},$$

$$= \sum_{i=1}^{M-n} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \bar{\mathbf{A}}^{m}\, \mathbf{D}\, \mathbf{A}^{i+n-1} + \sum_{i=M-n+1}^{M} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \bar{\mathbf{A}}^{m+1}\, \mathbf{D}\, \mathbf{A}^{i+n-M-1}, \qquad (6.190)$$

$$= \sum_{i=1}^{M-n} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{A}^{Mm}\, \mathbf{A}^{i+n-1} + \sum_{i=M-n+1}^{M} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{A}^{M(m+1)}\, \mathbf{A}^{i+n-M-1}, \qquad (6.191)$$

$$= \sum_{i=1}^{M} \mathbf{A}^{M-i}\, \mathbf{D}^{\mathrm{T}}\, \mathbf{D}\, \mathbf{A}^{i-1} \mathbf{A}^{Mm+n} = \mathbf{A}^{M-1+Mm+n}, \qquad (6.192)$$

where we use (6.188) in (6.190), and the first noble identity (6.25) in (6.191). The last equation (6.192) follows from the fact that M-block cyclic matrices provide perfect reconstruction in lazy filter banks. Hence, the polyphase matrix in (6.189) implies perfect reconstruction.

Chapter 7

UNCERTAINTY PRINCIPLES AND SPARSE EIGENVECTORS OF GRAPHS

7.1 Introduction

An essential concept in signal analysis is the uncertainty principle, which states that a signal cannot be arbitrarily localized in both time and frequency simultaneously [65]. In classical signal processing, the uncertainty principle is useful to design filters that are maximally localized in time for a given frequency spread, or vice versa. Due to its importance, some authors extended this principle to signals defined on graphs [2, 195, 104, 143]. Details of these approaches are elaborated in Section 7.1.3.

Similar to [2, 195, 104, 143], this chapter studies the concept of uncertainty for graph signals. However, unlike earlier methods, and motivated by [54, 59, 60], we introduce a non-local measure for uncertainty, based on the notion of sparsity of the signal in the graph domain and frequency domain. We show that a nonzero graph signal and its corresponding GFT cannot be arbitrarily sparse simultaneously, and we provide a lower bound for the total number of nonzero elements. We further provide the optimal selection of the GFB (that minimizes the uncertainty bound) when the graph operator has repeated eigenvalues. In order to find signals that achieve the derived lower bound, we consider sparse eigenvectors of the graph operator. A detailed outline and the contributions of this chapter are given below.

7.1.1 Outline and Contributions of This Chapter

Broadly speaking, results presented in this chapter can be divided into three main parts. In the first part (Sections 7.2 and 7.3), we propose *discrete* and *non-local* uncertainty principles that depend on the max-norm of the graph Fourier basis. These results follow from more general theory of sparse representations studied in [54, 59, 60]. We consider the identity matrix and the GFB as a pair of bases to represent a graph signal and interpret the methodologies in [60] in the context of graph signal processing. Subsequently, the study in [143] has obtained very similar interpretations based on the same theory. Apart from the overlapping results [143] generalizes these results to the case of frames. Then, it focuses on local uncertainty principles where bounds depend on a particular portion of the graph. More detailed comparison with [143] is provided in Section 7.1.3.

Our main contributions are presented in the remaining sections: in the second part of our results (Section 7.4) we discuss that given a graph, max-norm of the graph Fourier basis is not unique in the presence of the repeated eigenvalues of the selected graph operator. Since this non-uniqueness greatly affects the interpretation of the relations between the graph structure and the uncertainty, we formulate a problem to select eigenvectors (GFB) such that the max-norm of the graph Fourier basis is maximized (or, equivalently the uncertainty bounds considered in Section 7.2 are minimized). We solve this problem analytically (Theorem 7.6) and provide an algorithmic routine to obtain a set of eigenvectors that gives a graph Fourier basis with the maximum possible max-norm.

In the third part of our results (Section 7.5) we focus on the sparse eigenvectors of graphs in order to find signals that achieve the proposed uncertainty bounds. We study the relation between sparsity of eigenvectors and graph topology, and their effects on the max-norm of the GFB. We provide the necessary and sufficient conditions for the existence of 1-sparse and 2-sparse eigenvectors of the graph Laplacian (Theorems 7.7 and 7.8). We then show that existence of a 2-sparse eigenvector implies *very low and attainable uncertainty bounds* (Theorem 7.11). Finally in Section 7.7 we apply our results to real world graph examples. Interestingly, uncertainty bounds for these graphs are very low. We precisely explain why this is the case, and find the signals that achieve these bounds.

The content of this chapter is mainly drawn from [187], and parts of it have been presented in [169, 183].

7.1.2 Notation

In this chapter, the definition of the graph Fourier basis is not fixed. Eigenvectors of either the graph Laplacian or the adjacency matrix itself can be set to be the graph Fourier domain. We will use $\mathbf{V}_A$ and $\mathbf{V}_L$ to explicitly denote the eigenvectors of the adjacency matrix and the graph Laplacian, respectively. Assuming $\mathbf{A}$ is diagonalizable, we precisely have

$$\mathbf{A} = \mathbf{V}_A \, \mathbf{\Lambda}_A \, \mathbf{V}_A^{-1}, \qquad\qquad \mathbf{L} = \mathbf{V}_L \, \mathbf{\Lambda}_L \, \mathbf{V}_L^{-1}, \tag{7.1}$$

where $\mathbf{\Lambda}_A$ and $\mathbf{\Lambda}_L$ are diagonal eigenvalue matrices. We will always assume that eigenvectors are normalized to have a unit ℓ_2-norm. When the graph is undirected $\mathbf{V}_A$ and $\mathbf{V}_L$ can be selected to be unitary.

For a matrix $\mathbf{A}$ and two index sets $\mathcal{S}$ and $\mathcal{K}$, $\mathbf{A}$ denotes the matrix with columns of $\mathbf{A}$ indexed by $\mathcal{S}$ and $\mathbf{A}_{\mathcal{K},\mathcal{S}}$ denotes the matrix with columns indexed by S and rows indexed by K. For a vector $\mathbf{x}$, elements of $\mathbf{x}$ indexed by $\mathcal{S}$ will be denoted by $\mathbf{x}_{\mathcal{S}}$. The index set $\bar{\mathcal{S}}$ is defined as $\bar{\mathcal{S}} = \{1, \ldots, N\} \setminus \mathcal{S}$, where $\setminus$ stands for the set difference operator. We will use $\otimes$ to denote Kronecker product of matrices. Dimension of the right null space of a matrix $\mathbf{A}$ is denoted by nullity$(\mathbf{A})$.

7.1.3 Related Work

To the best to our knowledge, there are mainly four studies that consider uncertainty principles for signals defined over graphs [2, 195, 104, 143]. The main theme in these studies (including this one) is to define a "measure" of the signal in the vertex domain and the graph Fourier domain. In particular, the study in [2] considers the following for unweighted graphs:

$$\Delta^2_{g,u_0}(\mathbf{x}) = \frac{\mathbf{x}^{\mathrm{H}}\,\mathbf{P}^2_{u_0}\,\mathbf{x}}{\|\mathbf{x}\|_2^2}, \qquad \Delta^2_s(\mathbf{x}) = \frac{\mathbf{x}^{\mathrm{H}}\,\mathcal{L}\,\mathbf{x}}{\|\mathbf{x}\|_2^2}, \tag{7.2}$$

where $\Delta^2_{g,u_0}(\mathbf{x})$ is referred to as the vertex spread (with $\mathbf{P}_{u_0}$ being the diagonal distance matrix with respect to the node u_0), and $\Delta^2_s(\mathbf{x})$ is referred to as the spectral spread of the signal $\mathbf{x}$. This approach is extended to weighted graphs in [140]. The study in [195] works on an alternative definition and focuses on the following:

$$\alpha = \|\mathbf{D}_S\,\mathbf{x}\|_2 \;/\; \|\mathbf{x}\|_2, \qquad \beta = \|\mathbf{B}_F\,\mathbf{x}\|_2 \;/\; \|\mathbf{x}\|_2, \tag{7.3}$$

where α^2 and β^2 represent the amount of energy confined in the vertex set S and the frequency set F (with $\mathbf{D}_S$ and $\mathbf{B}_F$ being corresponding projection matrices), respectively. The study in [17, 104] considers the smoothness of the signal in both domains: using the difference operator on the graph, $\mathbf{D}_r$, the interplay between $\|\mathbf{D}_r\,\mathbf{x}\|_2^2$ and $\|\mathbf{D}_r\,\widehat{\mathbf{x}}\|_2^2$ is studied.

The main observation of these studies is that measures in both domains cannot be arbitrarily small simultaneously: a signal "limited" in one domain cannot be "limited" in the other domain. These trade-offs are then considered as uncertainty principles. Depending on their corresponding definitions, they characterize these uncertainty curves theoretically and study the signals that achieve them with equality.

More recently, the study in [143] takes a non-local perspective where vertex and spectral measures are defined with ℓ_p norms. Using the fact that the identity matrix and the GFB form a pair of bases to represent graph signals, [143] proposes some uncertainty principles where the results are based on theory of sparse representations studied in [60, 59, 54, 145]. These results show that the "graph Fourier coherence" is a fundamental quantity for the uncertainty principles of interest. Notice that we specifically consider the case of ℓ_0 and ℓ_1 in this chapter and obtain very similar results where we use the term "max-norm of GFB" instead of coherence. This overlap is a direct consequence of the influence of the sparse representation theory in [54, 59, 60]. In the rest, [143] focuses on the representation of graph signals using frames and proposes some uncertainty results based on these representations. It later considers local uncertainty principles where bounds depend on a region of the graph.

In order to find signals that achieve the proposed uncertainty bounds we consider sparse eigenvectors (2-sparse in particular) of the graph Laplacian. In this context, the study in [11] reveals a specific graph structure (referred to as motif-doubling) that results in sparse eigenvectors of both the adjacency matrix and the graph Laplacian. The structure of motif-doubling gives rise to sparse eigenvectors with even number of non-zero entries. However, this structure is only sufficient to have sparse eigenvectors, whereas our condition on 2-sparse eigenvectors is necessary and sufficient. A detailed comparison with [11] will be provided in Section 7.5.2.

7.2 Discrete Non-Local Spreads

Inspired by [54, 59, 60], we will study the concept of uncertainty from a *discrete* and *non-local* perspective. We will consider graph spread of a signal as the total number of nonzero elements in the signal. Similarly, spectral spread of a signal will be defined as the number of nonzero elements in the GFT of the signal. Hence, we have the following definitions:

Definition 7.1 (ℓ_0-based spread on vertex domain)**.** *Given a nonzero signal* $\mathbf{x}$ *on a graph with GFT* $\mathbf{F}$*, the "spread" of the signal on vertex domain is defined as* $||\mathbf{x}||_0$.

Definition 7.2 (ℓ_0-based spread on Fourier domain)**.** *Given a nonzero signal* $\mathbf{x}$ *on a graph with GFT* $\mathbf{F}$*, the "spread" of the signal in the Fourier domain is defined as* $||\mathbf{Fx}||_0$.

The definition of the spectral spread depends on the selected GFT $\mathbf{F}$. Whether it is based on the adjacency matrix, the graph Laplacian, or something else, for a given graph, the GFB $\mathbf{V}$, hence $\mathbf{F} = \mathbf{V}^{-1}$, may *not* be unique. This is due to the fact that the selected graph operator may have *repeated eigenvalues*. In such a case, one can select different bases to span the corresponding eigenspaces resulting in different GFB matrices. In order to avoid this ambiguity we assume that, not just the graph itself, but also the associated GFT is given in Definition 7.2. It should be noted that in the case of repeated eigenvalues selection of the GFB is not a simple task and it requires attention. We will address this problem in Sections 7.4 and 7.5, where we discuss optimal selection of the GFB in order to minimize the spectral spread of signals.

For a nonzero graph signal notice that its vertex domain spread and spectral spread have to be at least 1, however, they may *not* achieve this bound simultaneously. As a simple motivational example, consider the directed cyclic graph of N vertices, whose graph Fourier Transform corresponds to DFT of size N [152]. If the signal $\mathbf{x}$ is an impulse, then $\|\mathbf{x}\|_0 = 1$, but $\|\mathbf{F}\mathbf{x}\|_0 = N$. On the contrary, if the signal is a constant, then $\|\mathbf{x}\|_0 = N$, but $\|\mathbf{F}\mathbf{x}\|_0 = 1$. As a result we ask the following question: *Given the GFT* $\mathbf{F}$*, what is the minimum total number of nonzero elements in a graph signal and its corresponding Fourier Transform?* For this purpose, we consider the following two definitions of uncertainty:

$$s_0(\mathbf{x}) = \big(\|\mathbf{x}\|_0 + \|\mathbf{F}\mathbf{x}\|_0\big)/2, \qquad p_0(\mathbf{x}) = \sqrt{\|\mathbf{x}\|_0 \, \|\mathbf{F}\mathbf{x}\|_0}, \tag{7.4}$$

where $s_0(\mathbf{x})$ and $p_0(\mathbf{x})$ are referred to as *additive* and *multiplicative* uncertainty of the signal $\mathbf{x}$, respectively. The definitions in (7.4) have the following two important properties: 1) They are scale invariant, that is $s_0(\alpha\,\mathbf{x}) = s_0(\mathbf{x})$ and $p_0(\alpha\,\mathbf{x}) = p_0(\mathbf{x})$ for all $\alpha \neq 0$. This is a useful property since uncertainty of a signal is expected to be scale-invariant. 2) Both can take only a discrete, finite set of values, namely, $s_0(\mathbf{x})$ takes half integer values (i.e., $k/2$ for integer k) and $p_0(\mathbf{x})$ takes values in the form of $p_0(\mathbf{x}) = \sqrt{k}$ for some integer $k \geq 1$. As a final remark, notice that the AM-GM inequality dictates the following relation:

$$s_0(\mathbf{x}) \geq p_0(\mathbf{x}), \qquad \forall\, \mathbf{x} \in \mathbb{C}^N, \tag{7.5}$$

with equality if and only if $\|\mathbf{x}\|_0 = \|\mathbf{F}\mathbf{x}\|_0$.

The main purpose of this section is to find the lowest value that $s_0(\mathbf{x})$ can attain. More precisely the following problem will be considered:

$$s_0^\star = \min_{\mathbf{x}\neq\mathbf{0}} s_0(\mathbf{x}), \tag{7.6}$$

where $s_0^\star$ is called as the uncertainty bound since a signal and its corresponding graph Fourier Transform cannot be arbitrarily sparse simultaneously, that is, $s_0(\mathbf{x}) \geq s_0^\star \quad \forall \mathbf{x} \neq \mathbf{0}$.

Due to the combinatoric nature of the problem in (7.6), no closed form solution for $s_0^\star$ is available for an arbitrary $\mathbf{F}$. Nevertheless, the theory of sparse representations provides useful bounds for the problem. Motivated by Theorem 2.1 in [60], the following theorem (whose proof is presented in Appendix 7.9.1) provides a lower bound for $p_0(\mathbf{x})$:

Theorem 7.1 (Multiplicative uncertainty principle)**.** *For a graph with GFT* $\mathbf{F}$*, the multiplicative uncertainty of a nonzero signal* $\mathbf{x}$ *is lower bounded as follows:*

$$p_0(\mathbf{x}) \geq \left(\|\mathbf{F}^{-1}\|_2 \; \|\mathbf{F}\|_{\max}\right)^{-1}, \tag{7.7}$$

where $\|\cdot\|_2$ *and* $\|\cdot\|_{\max}$ *are described in Section 1.1.*

Corollary 7.1 (Additive uncertainty principle)**.** *For a graph with GFT* $\mathbf{F}$*, the additive uncertainty of a nonzero signal* $\mathbf{x}$ *is lower bounded as follows:*

$$s_0(\mathbf{x}) \geq \left(\|\mathbf{F}^{-1}\|_2 \; \|\mathbf{F}\|_{\max}\right)^{-1}, \tag{7.8}$$

where $\|\cdot\|_2$ *and* $\|\cdot\|_{\max}$ *are described in Section 1.1.*

Proof. From (7.5) we have $s_0(\mathbf{x}) \geq p_0(\mathbf{x})$. Therefore, any lower bound for $p_0(\mathbf{x})$ is also a lower bound for $s_0(\mathbf{x})$. □

When the GFT of interest is unitary, Theorem 7.1 and Corollary 7.1 reduce to the following corollaries:

Corollary 7.2 (Weak ℓ_0 uncertainty)**.** *For any graph with* unitary *graph Fourier basis* $\mathbf{V}$*, the following holds true:*

$$p_0(\mathbf{x}) \geq \|\mathbf{V}\|_{\max}^{-1}. \tag{7.9}$$

Proof. Let $\mathbf{V}$ be unitary in Theorem 7.1. Then we have $\mathbf{F} = \mathbf{V}^{\mathrm{H}}$, hence $\|\mathbf{F}\|_{\max} = \|\mathbf{V}\|_{\max}$. Furthermore, $\|\mathbf{V}\|_2 = \|\mathbf{F}^{-1}\|_2 = 1$. □

This corollary is important since for undirected graphs, the adjacency matrix and the graph Laplacian are symmetric which result in a unitary graph Fourier basis. The corollary also implies the following (from AM-GM inequality).

Corollary 7.3 (Strong ℓ_0 uncertainty)**.** *For any graph with* unitary *graph Fourier basis* $\mathbf{V}$*, the following holds true:*

$$s_0(\mathbf{x}) \geq \|\mathbf{V}\|_{\max}^{-1}. \tag{7.10}$$

In Section 7.7, we will provide graph examples on which the inequality in (7.10) is satisfied with equality.

Even though (7.10) is a lower bound for the uncertainty, it does not say anything about the signal that achieves the bound. Furthermore, it does not say whether *there is* a signal that achieves the lower bound or not. In order to understand the existence of such signals, we provide the following result.

Theorem 7.2 (Existence of signals)**.** *Let* $\mathbf{V}$ *be a unitary Fourier basis of the graph. There exists a signal* $\mathbf{x}$ *on the graph that achieves the strong* ℓ_0 *uncertainty bound (satisfies* (7.10) *with equality) if and only if there exist index sets* $\mathcal{K}$ *and* $\mathcal{S}$ *with* $|\mathcal{K}| = |\mathcal{S}| = \|\mathbf{V}\|_{\max}^{-1}$ *such that* $\text{nullity}\left((\mathbf{V}_{\mathcal{S},\bar{\mathcal{K}}})^{\mathrm{H}}\right) > 0$*. Furthermore, a signal that achieves the bound is given as*

$$\mathbf{x}_{\mathcal{S}} \in \text{null}\left((\mathbf{V}_{\mathcal{S},\bar{\mathcal{K}}})^{\mathrm{H}}\right), \qquad \mathbf{x}_{\bar{\mathcal{S}}} = \mathbf{0}. \tag{7.11}$$

Proof. Assume that $\mathbf{x}$ achieves the strong ℓ_0 bound, that is,

$$s_0(\mathbf{x}) = \frac{\|\mathbf{x}\|_0 + \|\mathbf{F}\mathbf{x}\|_0}{2} = \|\mathbf{V}\|_{\max}^{-1}. \tag{7.12}$$

Then we have $s_0(\mathbf{x}) = p_0(\mathbf{x})$, which implies

$$\|\mathbf{x}\|_0 = \|\mathbf{F}\mathbf{x}\|_0 = \|\mathbf{V}\|_{\max}^{-1}, \tag{7.13}$$

since $s_0(\mathbf{x}) = p_0(\mathbf{x})$ if and only if $\|\mathbf{x}\|_0 = \|\mathbf{F}\mathbf{x}\|_0$ due to AM-GM inequality. Let S denote the support of $\mathbf{x}$ and K denote the support of $\widehat{\mathbf{x}}$. Then $\mathbf{F}\mathbf{x} = \mathbf{F}_S\,\mathbf{x}_S$. Since $\widehat{\mathbf{x}}$ is zero outside of its support we have $\mathbf{F}_{\bar{K},S}\,\mathbf{x}_S = \mathbf{0}$, which means that $nullity(\mathbf{F}_{\bar{K},S}) = nullity\left((\mathbf{V}_{S,\bar{K}})^{\mathrm{H}}\right) > 0$ for some S and K with with $|S| = |K| = \|\mathbf{V}\|_{\max}^{-1}$.

Conversely, assume that nullity $\left((\mathbf{V}_{\mathcal{S},\bar{\mathcal{K}}})^{\mathrm{H}}\right) > 0$ for some $\mathcal{S}$ and $\mathcal{K}$ with $|\mathcal{S}| = |\mathcal{K}| = \|\mathbf{V}\|_{\max}^{-1}$. Then select $\mathbf{x}_{\mathcal{S}}$ as $\mathbf{x}_{\mathcal{S}} \in \text{null}\left((\mathbf{V}_{\mathcal{S},\bar{\mathcal{K}}})^{\mathrm{H}}\right)$ and $\mathbf{x}_{\bar{\mathcal{S}}}$ to be $\mathbf{0}$. Hence, $\|\mathbf{x}\|_0 = |\mathcal{S}|$. Since $\mathcal{S}$ is the support of $\mathbf{x}$, we have $\mathbf{F}_{\mathcal{S}}\,\mathbf{x}_{\mathcal{S}} = \mathbf{F}\,\mathbf{x}$. Furthermore,

$$\|\mathbf{F}_{\mathcal{S}}\,\mathbf{x}_{\mathcal{S}}\|_0 = \|\mathbf{F}_{\mathcal{K},\mathcal{S}}\,\mathbf{x}_{\mathcal{S}}\|_0 + \|\mathbf{F}_{\bar{\mathcal{K}},\mathcal{S}}\,\mathbf{x}_{\mathcal{S}}\|_0 = |K|, \tag{7.14}$$

since $\mathbf{x}_{\mathcal{S}} \in \text{null}(\mathbf{F}_{\bar{\mathcal{K}},\mathcal{S}})$. As a result, $s_0(\mathbf{x}) = \|\mathbf{V}\|_{\max}^{-1}$. $\square$

It should be noted that $\|\mathbf{V}\|_{\max}^{-1}$ may not be an integer in general. In this case, we cannot find index sets of size $\|\mathbf{V}\|_{\max}^{-1}$ since the size of an index set is an integer.

In this case, Theorem 7.2 tells us that there is no signal that achieves the bound in (7.10) with equality.

As an immediate example, the normalized inverse DFT matrix is a unitary graph Fourier basis for circulant graphs [56, 58]. Notice that the normalized DFT matrix of size N has $\|\mathbf{V}\|_{\max}^{-1} = \sqrt{N}$. Thus, circulant graphs have the following two results. 1) From Corollary 7.2, we have $p_0(\mathbf{x}) \geq \sqrt{N}$. This is a well-known uncertainty result given in [54], and the bound is known to be tight for all N. 2) From Corollary 7.3, we have $s_0(\mathbf{x}) \geq \sqrt{N}$. When N is a perfect square "picket fence" signal is known to achieve this bound [60]. Details of these results will be elaborated in Section 7.7.1.

Even though Theorem 7.2 is useful to characterize signals that achieve the bound in (7.10), it has a major drawback in terms of practical usability. Finding the index sets requires a combinatorial search over all possible sets of size $\|\mathbf{V}\|_{\max}^{-1}$, which is *not* computationally efficient.

7.3 Uncertainty Based on ℓ_1 Norm

In the previous section, we defined the spread of a signal as the total number of nonzero elements in the signal. (see Definitions 7.1 and 7.2.) In this section we will consider a "smoother" measure by replacing the ℓ_0 with ℓ_1 norm.

Imitating (7.4), we can define an ℓ_1 based additive uncertainty as $s_1(\mathbf{x}) = \big(\|\mathbf{x}\|_1 + \|\mathbf{F}\mathbf{x}\|_1\big)/2$. With this definition we have $s_1(\alpha\,\mathbf{x}) = |\alpha| s_1(\mathbf{x})$, that is, $s_1(\mathbf{x})$ is *not* scale invariant. The problem is that a nonzero signal can have arbitrarily small uncertainty, which is an undesired property. One way to impose the scale invariance is to use a normalization as follows:

$$s_1(\mathbf{x}) = \frac{\|\mathbf{x}\|_1 + \|\mathbf{F}\mathbf{x}\|_1}{2\,\|\mathbf{x}\|_p}. \tag{7.15}$$

For any nonzero p, $s_1(\mathbf{x})$ in (7.15) has the property of $s_1(\alpha\mathbf{x}) = s_1(\mathbf{x})$ for $\alpha \neq 0$, hence it can be used as an uncertainty measure. However, it should be noted that characteristics of $s_1(\mathbf{x})$ depend on the selected ℓ_p norm. In the following, we will consider the case of $p = 2$, and define the ℓ_1 based uncertainty measures as follows:

$$s_1(\mathbf{x}) = \frac{\|\mathbf{x}\|_1 + \|\mathbf{F}\mathbf{x}\|_1}{2\,\|\mathbf{x}\|_2}, \quad p_1(\mathbf{x}) = \frac{\sqrt{\|\mathbf{x}\|_1\ \|\mathbf{F}\mathbf{x}\|_1}}{\|\mathbf{x}\|_2}, \tag{7.16}$$

where $s_1(\mathbf{x})$ and $p_1(\mathbf{x})$ are referred to as ℓ_1 based additive and multiplicative uncertainty of the signal $\mathbf{x}$, respectively.

The main reason for considering the case of $p = 2$ is that such a selection has strong connections with the ℓ_0 based uncertainty measures discussed in Section 7.2. These relations will be elaborated at the end of this section (see Theorem 7.4).

As done in Section 7.2, motivated by Theorem 2.1 in [60], a lower bound for $p_1(\mathbf{x})$ can be obtained as follows (whose proof is presented in Appendix 7.9.2):

Theorem 7.3. *For a graph with GFT* $\mathbf{F}$, *ℓ_1-based multiplicative uncertainty of a nonzero signal* $\mathbf{x}$ *is lower bounded as follows:*

$$p_1(\mathbf{x}) \geq \left(\|\mathbf{F}^{-1}\|_2 \; \|\mathbf{F}\|_{\max}^{1/2} \right)^{-1}, \tag{7.17}$$

where $\|\cdot\|_2$ *and* $\|\cdot\|_{\max}$ *are described in Section 1.1.*

The reader should carefully notice the presence of the square root in (7.17), which was not there in (7.7).

When the GFT of interest is unitary Theorem 7.3 reduces to the following corollary.

Corollary 7.4 (Weak ℓ_1 uncertainty)**.** *For any graph with* unitary *Fourier basis* $\mathbf{V}$,

$$p_1(\mathbf{x}) \geq \|\mathbf{V}\|_{\max}^{-1/2}. \tag{7.18}$$

Proof. In Theorem 7.3, when $\mathbf{V}$ is unitary, we have $\|\mathbf{V}\|_2 = \|\mathbf{F}^{-1}\|_2 = 1$. Furthermore, $\mathbf{F} = \mathbf{V}^{\mathrm{H}}$, hence $\|\mathbf{F}\|_{\max} = \|\mathbf{V}\|_{\max}$. □

We can finally provide a lower bound for the ℓ_1-based additive uncertainty as follows:

$$s_1(\mathbf{x}) \geq \|\mathbf{V}\|_{\max}^{-1/2}, \tag{7.19}$$

where the inequality follows from $s_1(\mathbf{x}) \geq p_1(\mathbf{x})$ (AM-GM inequality). Hence, any lower bound for $p_1(\mathbf{x})$ (Corollary 7.4) is a lower bound for $s_1(\mathbf{x})$.

It is quite interesting to observe that the term $\|\mathbf{V}\|_{\max}$ appears in the lower bound for both ℓ_0-based and ℓ_1-based uncertainty definitions. It should be noted that $1/\sqrt{N} \leq \|\mathbf{V}\|_{\max} \leq 1$ for any matrix $\mathbf{V}$ since $\mathbf{V}$ is assumed to have columns with unit ℓ_2 norm. Therefore, $\|\mathbf{V}\|_{\max}^{-1} \geq \|\mathbf{V}\|_{\max}^{-1/2}$ always holds true. This shows that the lower bound given by (7.19) is always less than the bound in (7.10). In fact, not just the bounds but ℓ_0 and ℓ_1 based uncertainties are also related with each other. The following theorem (whose proof is presented in Appendix 7.9.3) establishes this relation.

Theorem 7.4. *Assume that graph of interest has a unitary GFB* $\mathbf{V}$*. Then, we have the following inequality*

$$p_0(\mathbf{x}) \geq \Big(p_1(\mathbf{x})\Big)^2, \tag{7.20}$$

for all signals defined on the graph.

Combining the result of Theorem 7.4, Corollary 7.4 and (7.5), we have the following relations between the aforementioned uncertainty definitions

$$s_0(\mathbf{x}) \geq p_0(\mathbf{x}) \geq \Big(p_1(\mathbf{x})\Big)^2 \geq \|\mathbf{V}\|_{\max}^{-1} \tag{7.21}$$

for a unitary graph Fourier basis. It should be noted that ℓ_0-based additive uncertainty has the strongest result. Namely, if a signal achieves the ℓ_0-based additive uncertainty bound ($s_0(\mathbf{x}) = \|\mathbf{V}\|_{\max}^{-1}$), then the signal also achieves the bounds given in Corollaries 7.2 and 7.4. (This is due to (7.21).) However, the converse is not true. For this reason, in the rest of the chapter, we will focus on $s_0(\mathbf{x})$ as the uncertainty measure.

7.4 Case of Repeated Eigenvalues

Definitions 7.1 and 7.2 are generic in the sense that one can choose any suitable graph Fourier basis. Most common selections are eigenvectors of the adjacency matrix [152], the graph Laplacian and normalized Laplacian [160]. Even after one decides on which of these should be used, there still is a significant point that requires attention: the possibility of *repeated eigenvalues*. This is mostly the case for unweighted graphs or graphs with integer edge weights (see Section 7.7). The relation between eigenvalue multiplicity of a graph and the topology of the graph is an interesting problem. Interested reader may refer to [118] for some results. More on this topic can be found in [73, 16, 44, 124].

In the following we will use eigenvectors of the graph Laplacian as the graph Fourier basis. However, the discussion is also valid for the adjacency matrix and the normalized graph Laplacian. *Further, we assume that the graph Laplacian, the adjacency matrix and the normalized graph Laplacian are diagonalizable matrices.* Hence, geometric and algebraic multiplicity of an eigenvalue are the same. This justifies the use of the term "multiplicity" without specifying which one.

Let λ_i be an eigenvalue of the graph Laplacian with multiplicity N_i. The corresponding eigenspace $\mathcal{S}_i$ is then defined as $\mathcal{S}_i = \text{null}(\mathbf{L} - \lambda_i \mathbf{I})$, where $\mathcal{S}_i$ is an N_i dimensional sub-space of $\mathbb{C}^N$. When λ_i is a repeated eigenvalue ($N_i > 1$), any vector in $\mathcal{S}_i$ is an eigenvector with eigenvalue λ_i. Therefore, by selecting different set of eigenvectors, one can come up with a different graph Fourier basis. Hence, *the graph Fourier basis for the graph Laplacian is not unique*.

Selection of the graph Fourier basis may affect the proposed uncertainty principles significantly. For example, consider the complete graph of N nodes. It is possible to select $\mathbf{V}$ such that either $||\mathbf{V}||_{\max}^{-1} = \sqrt{N/(N\text{-}1)}$, or $||\mathbf{V}||_{\max}^{-1} = \sqrt{N}$. The former decreases with N and approaches unity for large N, whereas the latter increases with N unboundedly. Therefore, interpretations of the proposed uncertainty bounds differ greatly depending on the selected basis. It should be noted that the complete graph of size N is an extreme example since it has an N-1 dimensional eigenspace corresponding to eigenvalue N. Nevertheless it shows the importance of the selection of the graph Fourier basis.

Since our definition of uncertainty depends on the selected graph Fourier basis, in the following, we will mainly discuss two different approaches for the selection of the eigenvectors. This section (Section 7.4) will study the first approach where we select the GFB in a way that the lower bound in Corollary 7.3 is minimized. In the next section (Section 7.5), we will consider the second approach where we look for the sparsest eigenvectors. We will also relate these two approaches in Section 7.5.

7.4.1 Minimizing the Lower Bound

In Corollary 7.3, we showed that the average number of nonzero elements in a graph signal and its graph Fourier transform is lower bounded by $||\mathbf{V}||_{\max}^{-1}$ where $\mathbf{V}$ is assumed to be unitary. When the graph of interest has repeated eigenvalues, one can select different set of eigenvectors which results in different values for $||\mathbf{V}||_{\max}^{-1}$. In this section, our purpose is to select eigenvectors of the given graph Laplacian $\mathbf{L}$ such that lower bound in Corollary 7.3 is *minimized* (or equivalently $||\mathbf{V}||_{\max}$ is maximized). We precisely define this problem as follows:

$$\max_{\mathbf{V}} \ ||\mathbf{V}||_{\max} \quad \text{s.t.} \quad \mathbf{L} = \mathbf{V}\,\mathbf{\Lambda}_L\,\mathbf{V}^{\mathrm{H}}, \tag{7.22}$$

where $\mathbf{\Lambda}_L$ is a diagonal matrix of eigenvalues of $\mathbf{L}$.

As a result of the graph Laplacian being a symmetric matrix, eigenspaces, $\mathcal{S}_i$, of $\mathbf{L}$ are orthogonal to each other. Hence, (7.22) is a decoupled problem in the sense that we can focus on individual eigenspaces rather than finding all eigenvectors of $\mathbf{L}$. To be more precise, assume that the graph Laplacian has K distinct eigenvalues with the corresponding eigenspaces $\mathcal{S}_i$ for $1 \leq i \leq K$. Then, we can write $\mathbf{V}$ as follows:

$$\mathbf{V} = [\mathbf{V}_1 \ \mathbf{V}_2 \ \cdots \ \mathbf{V}_K], \tag{7.23}$$

where each $\mathbf{V}_i$ has the dimension $\mathbf{V}_i \in \mathbb{C}^{N\times N_i}$ (N_i is the multiplicity of the corresponding eigenvalue), spans the eigenspace $\mathcal{S}_i$, and it is unitary $\mathbf{V}_i^{\mathrm{H}}\,\mathbf{V}_i = \mathbf{I}_{N_i}$ for $1 \leq i \leq K$. We also have $\mathbf{V}_i^{\mathrm{H}}\,\mathbf{V}_j = \mathbf{0}$ for $i \neq j$ (orthogonality of eigenspaces), and $\|\mathbf{V}\|_{\max} = \max\limits_{1\leq i\leq K} \|\mathbf{V}_i\|_{\max}$. Hence, we can write (7.22) as

$$\max_{1\leq i\leq K} \quad \max_{\mathbf{V}_i} \|\mathbf{V}_i\|_{\max} \quad \text{s.t.} \quad \begin{aligned} &\mathcal{S}_i = \mathrm{null}(\mathbf{L} - \lambda_i\mathbf{I}),\\ &\mathrm{span}(\mathbf{V}_i) = \mathcal{S}_i,\\ &\mathbf{V}_i^{\mathrm{H}}\,\mathbf{V}_i = \mathbf{I}_{N_i}, \end{aligned} \tag{7.24}$$

where λ_i is the i^{th} distinct eigenvalue of $\mathbf{L}$.

It is important to notice that inner maximization in (7.24) can be solved independently for each eigenspace. For this purpose, we have the following definition:

Definition 7.3 (Max-max norm of a subspace)**.** *Let $\mathcal{S}$ be an M-dimensional subspace of $\mathbb{C}^N$. The max-max norm of $\mathcal{S}$, $m(\mathcal{S})$, is defined as*

$$m(\mathcal{S}) = \max_{\mathbf{U}\in\mathbb{C}^{N\times M}} \|\mathbf{U}\|_{\max} \quad s.t. \quad \mathrm{span}(\mathbf{U}) = \mathcal{S}, \quad \mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}.$$

That is, $m(\mathcal{S})$ is the maximum of max-norm of matrices with orthonormal columns that span the given sub-space $\mathcal{S}$.

Notice that *any* element of *any* unitary basis that spans $\mathcal{S}$ is always less than (or equal to) $m(\mathcal{S})$ in absolute sense. In the following theorem (whose proof is presented in Appendix 7.9.4), we will provide a closed form solution for the max-max norm of a sub-space.

Theorem 7.5. *Let $\mathcal{S}$ be a M-dimensional subspace of $\mathbb{C}^N$. Let $\mathbf{U} \in \mathbb{C}^{N\times M}$ be* any *matrix with* $\mathrm{span}(\mathbf{U}) = \mathcal{S}$*, and $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$. Then max-max norm of sub-space $\mathcal{S}$ is*

$$m(S) = \max_{1\leq j\leq N} \sqrt{\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)_{j,j}}, \tag{7.25}$$

where $(\cdot)_{j,j}$ denotes the j^{th} diagonal entry.

Finally, we state the maximized objective function value in (7.22) in the following theorem.

Theorem 7.6 (Maximum Max-Norm of GFB)**.** *Assume the graph Laplacian,* $\mathbf{L}$*, has K distinct eigenvalues. Let N_i denote the multiplicity, and $\mathcal{S}_i$ denote the eigenspace* *of the eigenvalue λ_i for $1 \leq i \leq K$. Let $\mathbf{U}_i \in \mathbb{C}^{N \times N_i}$ be* any *matrix with $\mathbf{U}_i^{\mathrm{H}} \mathbf{U}_i = \mathbf{I}$ and $span(\mathbf{U}_i) = \mathcal{S}_i$. Then we have the following:*

$$\max_{\substack{1 \leq j \leq N \\ 1 \leq i \leq K}} \sqrt{\left(\mathbf{U}_i \, \mathbf{U}_i^{\mathrm{H}}\right)_{j,j}} = \max_{\mathbf{V}} \ \|\mathbf{V}\|_{\max} \quad s.t. \quad \mathbf{L} = \mathbf{V} \boldsymbol{\Lambda}_L \mathbf{V}^{\mathrm{H}},$$

where $\boldsymbol{\Lambda}_L$ is a diagonal matrix of eigenvalues of $\mathbf{L}$.

Proof. Follows from equivalence between (7.22) and (7.24), Definition 7.3 and Theorem 7.5. $\square$

Theorem 7.6 only provides the value of the maximized objective function in (7.22), which is useful to find the minimum lower bound given by Corollary 7.3. In fact, we can explicitly construct the set of eigenvectors that result in the maximum max-norm. For this purpose, consider again the proof of Theorem 7.5. Notice that it is a *constructive* proof, which can be translated into an algorithm as follows. Given the graph Laplacian, one can take the eigenvalue decomposition and obtain $\mathbf{L} = \mathbf{V} \boldsymbol{\Lambda}_L \mathbf{V}^{\mathrm{H}}$ with a proper ordering of eigenvectors such that $\mathbf{V} = [\mathbf{V}_1 \cdots \mathbf{V}_K]$ and columns of each $\mathbf{V}_i \in \mathbb{C}^{N \times N_i}$ belong to the same eigenspace. Then, we utilize the following three steps for each $\mathbf{V}_i$:

1. Let $\mathbf{v}_{i,j}^{\mathrm{H}}$ denote the j^{th} row of $\mathbf{V}_i$, and let $j^\star$ be such that $j^\star = \arg \max_{1 \leq j \leq N} \|\mathbf{v}_{i,j}^{\mathrm{H}}\|_2$.
2. Let $\mathbf{X}_i = [\mathbf{x}_{i,1} \quad \mathbf{x}_{i,2} \quad \cdots \quad \mathbf{x}_{i,N_i}] \in \mathbb{C}^{N_i \times N_i}$, and select $\mathbf{x}_{i,1} = \mathbf{v}_{i,j^\star} / \|\mathbf{v}_{i,j^\star}\|_2$. Remaining columns of $\mathbf{X}_i$ can be selected *arbitrarily* such that $\mathbf{X}_i^{\mathrm{H}} \, \mathbf{X}_i = \mathbf{I}_{N_i}$ holds true.
3. Compute $\mathbf{V}_i^\star = \mathbf{V}_i \, \mathbf{X}_i$.

Then, the set of eigenvectors that has the maximum max-norm can be constructed as $\mathbf{V}^\star = [\mathbf{V}_1^\star \quad \cdots \quad \mathbf{V}_K^\star]$.

Notice that $\mathbf{V}_i^\star$ is just a different unitary basis for the eigenspace spanned by $\mathbf{V}_i$. However, unlike $\|\mathbf{V}_i\|_{\max}$, $\|\mathbf{V}_i^\star\|_{\max}$ is guaranteed to achieve the max-max norm of the corresponding eigenspace (Theorem 7.5). As a result, $\mathbf{V}^\star$ is a solution to (7.22) since it has the largest max-norm among all possible selections of the eigenvector matrices.

7.4.2 Numerical Problems

Even though Theorem 7.6 provides an efficient way to select the eigenvectors such that lower bound in Corollary 7.3, $\|\mathbf{V}\|_{\max}^{-1}$, is minimized, there is an important numerical issue. In order to utilize Theorem 7.6, we need to group the eigenvectors that belong to the same eigenspace, which requires a equality check between the corresponding eigenvalues. However, it is not possible to distinguish two values if they are closer than the *precision* of the numerical system. As a particular example consider a (undirected, unweighted) path graph of size N. Eigenvalues of the adjacency matrix of this graph are given as $\lambda_k = 2\cos(\pi k/(N{+}1))$ for $1 \leq k \leq N$ [29]. One can show that $|\lambda_1\text{-}\lambda_2| \leq \epsilon$, when the size of the graph is $N \geq \sqrt{3}\,\pi\,\epsilon^{-1/2}$. Hence, for any numerical precision ϵ, there exists a graph of size N such that λ_1 and λ_2 cannot be distinguished from each other. Study in [50] has observed similar numerical problems as well.

7.5 Sparse Eigenvectors of Graphs

After the definition of additive uncertainty given in (7.4), the ultimate purpose of this chapter is to find a solution to (7.6) in order to find signals that are sparse *both* in the vertex domain and the Fourier domain of a given graph. Unfortunately, solution to (7.6) is not straightforward due to its combinatorial nature. We have provided lower bounds for the solution to (7.6) in Corollaries 7.1 and 7.3. In Section 7.4 we studied the optimal selection of the graph Fourier basis in order to minimize the lower bound given by Corollary 7.3. However, these approaches have two downsides 1) Even though there are examples where the bound given by Corollary 7.3 is tight (see Sec 7.7), it may not be the case for an arbitrary graph. 2) Even if the solution to (7.6) is known, aforementioned results are unable to find the signals that achieve the minima (except for Theorem 7.2, which requires an exhaustive search to find a bound achieving signal). In this section, in order to overcome these downsides, we will consider additive uncertainty of graph Fourier basis elements.

Without losing any generality we will use orthogonal eigenvectors of the graph Laplacian as the graph Fourier basis. That is, $\mathbf{V}$ is the GFB where $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}_L\mathbf{V}^{\mathrm{H}}$. Then, assuming a predefined ordering of eigenvectors, the i^{th} column of $\mathbf{V}$, denoted by $\mathbf{v}_i$, will be the i^{th} element of GFB. Since GFT is defined as $\mathbf{F} = \mathbf{V}^{-1}$, we have $\|\mathbf{F}\mathbf{v}_i\|_0 = 1$. Then, the additive uncertainty of a graph signal that is an element of GFB is given as

$$s_0(\mathbf{v}_i) = \big(\|\mathbf{v}_i\|_0 + 1\big)/2. \tag{7.26}$$

Notice that quantity in (7.26) is directly related to the sparsity of the GFB element. If $\mathbf{v}_i$ itself is a sparse vector, then we have a *direct evidence* of a signal that is sparse in the vertex domain and graph Fourier domain simultaneously. Furthermore, additive uncertainty of sparse eigenvectors may achieve (or, come close to) the bound given in Corollary 7.3. Therefore, our aim in this section, is to find sparse eigenvectors of graphs. However, it should be noted that a signal that achieves the minima of the additive uncertainty may *not* be an element of the GFB. Therefore, if the GFB elements, $\mathbf{v}_i$, are dense, we cannot reach any conclusion using (7.26). In Section 7.7.1 we will provide examples in this regard.

7.5.1 Sparse Vectors in an Eigenspace

When the graph Laplacian has repeated eigenvalues, eigenvectors are not unique and they form a sub-space. In Section 7.4.1, we discussed how eigenvectors can be selected so that *the lower bound* for $s_0(\mathbf{x})$ is minimized. In this section we will try to select eigenvectors such that $s_0(\mathbf{v}_i)$ in (7.26) is minimized. To be more precise, assume that $\mathbf{L}$ has K ($K \leq N$) distinct eigenvalues with corresponding eigenspaces $\mathcal{S}_i$ for $1 \leq i \leq K$. Then we consider the following problem:

$$\min_{\mathbf{v}} \ \|\mathbf{v}\|_0 \quad \text{s.t.} \quad \mathbf{v} \in \mathcal{S}_i, \quad \|\mathbf{v}\|_2 = 1 \tag{7.27}$$

for each eigenspace of $\mathbf{L}$.

The problem in (7.27) is precisely defined as "The Null Space Problem" in [40], and shown to be NP-Hard. Interested reader is referred to [41, 18, 70] for computational approaches to solution of (7.27). Apart from these, the study in [144] proposes an iterative algorithm in order to find an approximate solution to (7.27) via ℓ_1 relaxation.

Unlike the case of minimizing the lower bound in Corollary 7.3 (see Theorem 7.6), selection of the sparsest eigenvector is not computationally tractable due to NP completeness of the problem in (7.27). However, it is quite interesting that the max-max norm of a subspace (given in Definition 7.3) provides a lower bound for the problem in (7.27). In the following we will precisely establish this relation.

Let $\mathbf{x} \in \mathbb{C}^N$ be a vector with unit ℓ_2 norm, $\|\mathbf{x}\|_2 = 1$. Then the infinity norm of $\mathbf{x}$ can be bounded as $1 \geq \|\mathbf{x}\|_\infty \geq 1/\sqrt{N}$. In fact, if we further know that $\mathbf{x}$ has L nonzero elements ($L \leq N$), we can improve the lower bound as $\|\mathbf{x}\|_\infty \geq 1/\sqrt{L}$. Therefore, we have the following inequality:

$$\|\mathbf{x}\|_0 \geq \|\mathbf{x}\|_\infty^{-2} \qquad \forall \mathbf{x} \quad \text{s.t} \quad \|\mathbf{x}\|_2 = 1, \tag{7.28}$$

where equality is achieved when $|x_i| = 1/\sqrt{L}$ for all i's in the support of $\mathbf{x}$.

Using the inequality in (7.28) we can obtain a lower bound for (7.27) as follows:

$$\min_{\substack{\mathbf{x}\in\mathcal{S}\\ \|\mathbf{x}\|_2=1}} \|\mathbf{x}\|_0 \geq \min_{\substack{\mathbf{x}\in\mathcal{S}\\ \|\mathbf{x}\|_2=1}} \|\mathbf{x}\|_\infty^{-2} = \left(\max_{\substack{\mathbf{x}\in\mathcal{S}\\ \|\mathbf{x}\|_2=1}} \|\mathbf{x}\|_\infty\right)^{-2}, \tag{7.29}$$

where we use the fact that $\|\mathbf{x}\|_\infty$ is strictly positive, finite, and bounded away from zero so that $\|\mathbf{x}\|_\infty^{-1}$ is finite.

In the following we will show that the maximization problem in the right hand side of (7.29) is equivalent to definition of max-max norm of the subspace $\mathcal{S}$. Remember from Definition 7.3 that max-max norm is defined as

$$m(\mathcal{S}) = \max_{\mathbf{U}\in\mathbb{C}^{N\times M}} \|\mathbf{U}\|_{\max} \qquad \text{s.t.} \qquad \text{span}(\mathbf{U}) = \mathcal{S}, \quad \mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}. \tag{7.30}$$

Assume that $\mathbf{U}^\star = [\mathbf{u}_1^\star \cdots \mathbf{u}_M^\star]$ is a solution to the problem in (7.30). Then we have $m(\mathcal{S}) = \|\mathbf{U}^\star\|_{\max} = \max_i \|\mathbf{u}_i^\star\|_\infty$. Further, $\mathbf{u}_i^\star \in \mathcal{S}$, and $\|\mathbf{u}_i^\star\|_2 = 1$. Hence we have $m(\mathcal{S}) \leq \max_{\mathbf{x}} \|\mathbf{x}\|_\infty,\ \mathbf{x} \in \mathcal{S},\ \|\mathbf{x}\|_2 = 1$.

Now assume that $\mathbf{x}^\star$ is a solution to right hand side of (7.29). Then consider the matrix $\mathbf{U} = [\mathbf{x}^\star\ \mathbf{u}_2 \cdots \mathbf{u}_M]$ by selecting $\mathbf{u}_i$'s such that $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$ and $\text{span}(\mathbf{U}) = \mathcal{S}$. Hence $\mathbf{U}$ is in the feasible set of the problem in (7.30). Therefore, $m(\mathcal{S}) \geq \|\mathbf{U}\|_{\max} \geq \|\mathbf{x}^\star\|_\infty = \max_{\mathbf{x}} \|\mathbf{x}\|_\infty,\ \mathbf{x} \in \mathcal{S},\ \|\mathbf{x}\|_2 = 1$.

As a result, we conclude that maximization on the right-hand side of (7.29) is equivalent to $m(\mathcal{S})$, and provide the following lower bound for the solution of the problem in (7.27)

$$\Big(m(\mathcal{S})\Big)^{-2} \leq \min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{s.t.} \quad \mathbf{x} \in \mathcal{S}, \quad \|\mathbf{x}\|_2 = 1. \tag{7.31}$$

The two main points of this section can be summarized as follows:

1) The search for a sparse eigenvector in a specific eigenspace is an inherently difficult problem. Even though numerical techniques that approximate the solution exist [144], closed form solutions are not available in general. In this aspect, computation of sparse eigenvectors differs from the max-norm approach discussed in Section 7.4, where we provided closed form solutions by focusing on individual eigenspaces.

2) Although we do not have a closed form solution for the sparsest vector in a given eigenspace, we can provide a lower bound for the total number of nonzero elements

as in (7.31). The inequality in (7.31) is especially useful when we want to show that an eigenspace does *not* have sparse vectors. We will use this inequality in Section 7.7.1 to formally show that an undirected cycle graph does not have sparse eigenvectors.

7.5.2 Algebraic Characterization of Sparse Eigenvectors

In the previous section we mentioned that finding the sparsest vector in an eigenspace is a difficult problem. Therefore, when looking for sparse eigenvectors, we should consider the graph (Laplacian) as a whole rather than focusing on each eigenspace individually. Furthermore, in Section 7.4.2 we mentioned that characterization of eigenspaces of graphs suffers from numerical precision when the graph is large (relative to the numerical precision of the system). This is a significant problem especially when a large-scale real-world data is of interest. Therefore, we need a way to characterize the sparse eigenvectors of graphs without using *numerical* techniques. The purpose of this section is to find these sparse eigenvectors *algebraically*.

In the case of disconnected graphs we have a straightforward result. Assume that the graph is *undirected* but *non-negatively weighted* and consists of D disconnected components with sizes C_i. Then the adjacency matrix and the graph Laplacian can be written in the following form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & \\ & \ddots & \\ & & \mathbf{A}_D \end{bmatrix}, \qquad \mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & & \\ & \ddots & \\ & & \mathbf{L}_D \end{bmatrix}, \tag{7.32}$$

where $\mathbf{A}_i \in \mathbb{R}^{C_i \times C_i}$ and $\mathbf{L}_i \in \mathbb{R}^{C_i \times C_i}$ are the adjacency matrix and the graph Laplacian of the i^{th} component, respectively. Due to block-diagonal form of $\mathbf{A}$ and $\mathbf{L}$, corresponding eigenvectors can be selected to be block sparse. Therefore, there exists an eigenvector that has *at most* C_i nonzero elements for each $1 \leq i \leq D$. It is important to note that the converse of this result is not true: if a graph has a sparse eigenvector, it does *not* imply that the graph is disconnected. As a counter example consider Theorem 7.8, which proves that a connected graph can have a sparse eigenvector. Examples of such graphs will be provided at the end of this section. However, 1-sparse eigenvectors are exemption in this regard. That is, a graph has a 1-sparse eigenvector if and only if it has an isolated node. This result (whose proof is presented in Appendix 7.9.5) is stated as follows.

Theorem 7.7 (Isolated nodes of a graph)**.** *Assume that the graph of interest is* undirected *but* non-negatively weighted*. Then, the following statements are equivalent:*

1) The graph has an isolated node.

2) The graph Laplacian has a 1*-sparse eigenvector.*

3) The GFB can *be selected such that there exists a nonzero signal that achieves* $s_0(\mathbf{x}) = 1$*, i.e.,* $\|\mathbf{x}\|_0 = \|\mathbf{F}\mathbf{x}\|_0 = 1$.

Now we provide the characterization theorem for 2-sparse eigenvectors of graphs, whose proof is provided in Appendix 7.9.6. Recall that a graph is said to be connected if there is a path between any pair of nodes.

Theorem 7.8 (2-sparse eigenvectors of a connected graph)**.** *Let* $\mathbf{A}$ *denote the adjacency matrix of an* undirected *and* connected *graph with* $a_{i,j} \geq 0$ *being the weight of the edge between nodes* i *and* j*. Then, there exist nodes* i *and* j *such that*

$$a_{i,k} = a_{j,k} \quad \forall\, k \in \{1, \cdots, N\} \setminus \{i, j\}, \tag{7.33}$$

if and only if *the graph Laplacian,* $\mathbf{L}$*, has a* 2*-sparse eigenvector with nonzero eigenvalue* $\lambda = d_i + a_{i,j}$*. When the graph is unweighted,* (7.33) *can be stated as*

$$\mathcal{N}(i) \setminus \{j\} = \mathcal{N}(j) \setminus \{i\}, \tag{7.34}$$

where $\mathcal{N}(i)$ *is the set of nodes that are adjacent to node* i*.*

Note that a 2-sparse eigenvector can be assumed to have values 1 and -1 on the nodes with the property (7.33). (See the proof in Appendix 7.9.6.) This result is especially useful for Theorem 7.11.

As a historical note we would like to mention that existence of a 2-sparse eigenvector was first presented in [62, Theorem 5.10] for the case of unweighted graphs. We also note that the study [62] had a different motivation, and it did not explore the connections between sparse eigenvectors and uncertainty bounds on graphs. Later, the study in [117] mentioned the specific structure of 2-sparse eigenvectors (see [117, Example 3.7]) *without focusing on its sparsity*, and the study called such an eigenvector as *a Faria vector* in reference to the author of the paper in [62]. In short, 2-sparse eigenvectors are also known as Faria vectors in graph theory.

Similar to Theorem 7.8, the study in [11] also reveals a specific graph structure that results in sparse eigenvectors of both the adjacency matrix and the graph Laplacian. In particular, it considers the case when a graph has two copies of the same subgraph (referred to as “motif doubling” in [11]). That is, there are two disjoint subsets (of size K) of nodes, S_1 and S_2, such that the induced sub-graphs on S_1 and S_2 are the same, there is no edge between S_1 and S_2, and S_2 is connected to the rest of the graph in the same way S_1 is. In this case the adjacency matrix (and the graph Laplacian) can be shown to have $(2K)$-sparse eigenvectors (Theorem 2.2 of [11]). In the case of $K = 1$ (each subset has only one node), this motif doubling property reduces to the condition in (7.34) with $a_{i,j} = 0$.

However, it is important to note that the condition in Theorem 7.8 is more general than the one in [11] due to the following two reasons: 1) the construction in [11] provides only a *sufficient* condition, whereas Theorem 7.8 gives the necessary and sufficient condition to have a 2-sparse eigenvector. 2) The motif doubling idea in [11] specifically considers the case when $a_{i,j} = 0$, whereas Theorem 7.8 is applicable to the case of $a_{i,j} \neq 0$ as well. In the general case of K, it is straightforward to find sufficient conditions for a K-sparse eigenvector to exist: the motif doubling in [11] and "the same neighborhood structure" (to be discussed after Theorem 7.9) are two such examples. On the other hand, it is difficult to reveal the necessary conditions for K-sparse eigenvectors to exist.

Now we provide the characterization theorem for 3-sparse eigenvectors of *unweighted* graphs, whose proof is presented in Appendix 7.9.7. We later show the connection between 3 and 2-sparse eigenvectors of unweighted graphs.

Theorem 7.9 (3-sparse eigenvectors of a connected graph)**.** *Let* **A** *denote the adjacency matrix of an undirected, unweighted, and connected graph. There exist nodes i, j, and k such that*

$$\mathcal{N}(i) \setminus \{j, k\} = \mathcal{N}(j) \setminus \{i, k\} = \mathcal{N}(k) \setminus \{i, j\}, \tag{7.35}$$

if and only if the graph Laplacian, **L**, *has a 3-sparse eigenvector with non-zero eigenvalue.*

There are two remarks regarding Theorems 7.8 and 7.9. 1) The existence of sparse eigenvectors does *not* depend on the size and the global structure of the graph. Existence of nodes with the properties in (7.33) or (7.35) directly implies the claimed results. 2) The sparse eigenvectors are localized on the graph. If the nodes have the properties in (7.33) or (7.35), they must have at least one common neighbor. (This follows from the fact that the graph is connected). Hence, non-zero elements of the eigenvector are *at most* 2 *hops away from each other.*

Similarity between the conditions (7.34) and (7.35) encourages us to pursue a more general condition on a (connected) graph so that an eigenvector with an arbitrary number of non-zero elements exists. In fact, such a generalization is possible only as a sufficient condition. However, finding a necessary condition is not easy. The main reason is that it is possible to combine sparser eigenvectors in a given eigenspace in order to achieve less sparse ones. To see this, let K be an arbitrary sparsity $K \geq 4$. It can be written as $K = 2m + 3n$ for some integer m and n. Hence, if there exist m 2-sparse and n 3-sparse eigenvectors (with the same eigenvalue and disjoint supports), a linear combination of these 2 and 3-sparse eigenvectors yields a K-sparse eigenvector. Furthermore, m and n are not unique for a given K in general. In short, a K-sparse eigenvector might exist for various different reasons, which makes it difficult to find a necessary condition for a K-sparse eigenvector to exist. In particular, consider the Minnesota road graph (to be studied in Section 7.7.4). It has four orthogonal 2-sparse eigenvectors with eigenvalue 1. (See Figure 7.3a-7.3d.) Since these 2-sparse eigenvectors are in the same eigenspace, any linear combination of these is also an eigenvector. Furthermore, it is apparent from Figure 7.3a-7.3d that these 2-sparse eigenvectors have disjoint supports. As a result, one can find a 6-sparse eigenvector via a linear combination of three 2-sparse eigenvectors. However, a 6-sparse eigenvector could have been the result of a combination of two 3-sparse eigenvectors (with the same eigenvalue and disjoint supports) as well. This empirically shows that a necessary condition is not easy to obtain for an arbitrary sparsity. Also notice that one can find 4, 6, and 8-sparse eigenvectors via linear combinations of the 2-sparse eigenvectors of Figure 7.3a-7.3d. Unlike the 2-sparse ones, these 4, 6, and 8-sparse eigenvectors are not localized (in terms of number of hops) on the graph. Hence, *a K-sparse eigenvector may not be localized on the graph.*

It is interesting to observe that the condition for 3-sparse eigenvectors is more strict than the condition for 2-sparse eigenvectors for unweighted graphs. We formally state this result as follows (whose proof is presented in Appendix 7.9.8):

Theorem 7.10 (3-sparse implies 2-sparse)**.** *If the Laplacian of an undirected, unweighted and connected graph has a 3-sparse eigenvector, then it has a 2-sparse eigenvector.*

It is important to note that the result of Theorem 7.10 is specific to 2 and 3-sparse eigenvectors and cannot be generalized to arbitrary sparsity. As a simple counter-example, consider the Minnesota road graph (Section 7.7.4). It has 2-sparse and 4-sparse eigenvectors, but it does *not* have a 3-sparse eigenvector.

7.6 Sparse Eigenvectors and Uncertainty Bounds

As discussed in Section 7.2, the additive uncertainty of a signal $\mathbf{x}$ in (7.4) can take only half integer values for nonzero signals, that is $s_0(\mathbf{x}) \in \{1,\ 3/2,\ 2,\ \cdots,\ N\}$. It should be noted that Theorem 7.7 precisely characterizes the case when $s_0(\mathbf{x})$ takes its possible minimum value. A nonzero signal has $s_0(\mathbf{x}) = 1$ *if and only if* the graph has an isolated node. This result is especially useful to conclude that a nonzero signal on a *connected graph*, which does not have any isolated node, *cannot* achieve $s_0(\mathbf{x}) = 1$. Therefore, we consider the next attainable case for connected graphs, that is, $s_0(\mathbf{x}) = 3/2$. This happens under two circumstances

1. $\|\mathbf{x}\|_0 = 1$, $\|\mathbf{Fx}\|_0 = 2$: the signal $\mathbf{x}$ is an impulse on the vertex domain, and it has a 2-sparse GFT.
2. $\|\mathbf{x}\|_0 = 2$, $\|\mathbf{Fx}\|_0 = 1$: the signal $\mathbf{x}$ is a 2-sparse eigenvector of the graph Laplacian.

We note that Theorem 7.8 precisely characterizes the second case. Therefore, given a connected graph, existence of a pair of nodes that satisfy (7.33) implies that $s_0(\mathbf{x}) \geq 3/2$ for all nonzero signals on the graph. Furthermore, the bound is tight, and the signal that achieves the bound is known. This result is formally stated as follows.

Theorem 7.11 (Uncertainty bound for connected graphs)**.** *For an* undirected*,* connected*, and* non-negatively weighted *graph, assume that there exist nodes i and j satisfying the condition in* (7.33)*. Then, the GFB with respect to the graph Laplacian* can be *selected such that*

$$s_0(\mathbf{x}) \geq 3/2 \qquad \forall \mathbf{x} \neq \mathbf{0}. \tag{7.36}$$

Furthermore, the signal achieving this bound, $s_0(\mathbf{x}^\star) = 3/2$*, is given as* $x_i^\star = \text{-}x_j^\star = 1$ *and zero everywhere else.*

Proof. For a simple and connected graph, Theorem 7.7 says that there is no signal such that $s_0(\mathbf{x}) = 1$. Since $s_0(\mathbf{x})$ can take only half integers in $[1,\ N]$, $s_0(\mathbf{x}) \neq 1$ implies that $s_0(\mathbf{x}) \geq 3/2$ for any nonzero signal $\mathbf{x}$.

Furthermore, if there is a pair of nodes satisfying (7.33), Theorem 7.8 says that the graph Laplacian has a 2-sparse eigenvector. Let $\mathbf{v}$ denote this eigenvector. Then we have $v_i = -v_j = 1$ and zero everywhere else. Notice that GFB with respect to the graph Laplacian *can be* selected such that $\mathbf{v}$ is an element of GFB. In this case we have $\|\mathbf{v}\|_0 = 2$ and $\|\mathbf{F}\mathbf{v}\|_0 = 1$, hence $s_0(\mathbf{v}) = 3/2$. This shows that the lower bound in (7.36) is tight and attainable. □

There are four remarks regarding Theorem 7.11.

1) The tightness of the bound given in (7.36) does *not* depend on the size and the global structure of the graph. Existence of a pair of nodes with (7.33) directly implies this result.

2) The signal that achieves the bound is localized on the graph. Notice that if two nodes have the property in (7.33), they must have at least one common neighbor. (This follows from the fact that the graph is connected.) As a result, nonzero elements of the signal that achieves the bound in (7.36) are at most 2 hops away from each other. However, localization property is unique to 2-sparse eigenvectors. An eigenvector with an arbitrary level of sparsity may not be localized on the graph.

3) Due to Corollary 7.3, the inequality $s_0(\mathbf{x}) \geq \|\mathbf{V}\|_{\max}^{-1}$ is always true. When $3/2$ is the smallest attainable value for $s_0(\mathbf{x})$, we have $3/2 \geq \|\mathbf{V}\|_{\max}^{-1}$. Therefore, existence of a pair of nodes with (7.33) proves that there exists a GFB $\mathbf{V}$ such that $\|\mathbf{V}\|_{\max} \geq 2/3$. In fact, using Corollary 7.2 this result can be slightly improved to $\|\mathbf{V}\|_{\max} \geq 1/\sqrt{2}$.

4) In the case of repeated eigenvalues of the graph Laplacian, the GFB is not unique. However, uncertainty bounds *depend* on the selection of the GFB. When there are repeated eigenvalues GFB should be selected properly (sparse eigenvector should be an element of GFB) in order to have (7.36). This point will be numerically demonstrated in Section 7.7.1.

In the following we will provide three classical graph examples that satisfy, or do not satisfy, the condition in (7.34). Notice that these graphs are simple (undirected, unweighted, free from self-loops) and connected.

Complete Graph, K_N

A complete graph on N nodes has an edge between any two nodes. Figure 7.1a provides a visual representation of K_8. Let i, j and k be three arbitrary nodes of a complete graph. Then, we have

$$\mathcal{N}(i) \setminus \{j, k\} = \mathcal{N}(j) \setminus \{i, k\} = \mathcal{N}(k) \setminus \{i, j\} = \{1, \cdots, N\} \setminus \{i, j, k\}, \quad (7.37)$$

which shows that an unweighted complete graph of an arbitrary size ($N \geq 3$) has a 3-sparse eigenvector, which in particular implies that it has a 2-sparse eigenvector as well (Theorem 7.10).

Complete Bi-Partite Graph, $K_{N,M}$

A complete bi-partite graph of size $N + M$ is a bi-partite graph (one color having N nodes, and other color having M nodes) such that every node of a color is connected to every node of the other color. Figure 7.1b provides a visual representation of $K_{4,5}$. Let i, j, and k be three nodes that belong to *the same* color. Then we have that

$$\mathcal{N}(i) \setminus \{j, k\} = \mathcal{N}(j) \setminus \{i, k\} = \mathcal{N}(k) \setminus \{i, j\} = \text{Nodes of the other color}, \quad (7.38)$$

which shows that an unweighted complete bi-partite graph of an arbitrary size (given that a color has at least 3 nodes) has a 3-sparse eigenvector, which in particular implies that it has a 2-sparse eigenvector as well.

Star Graph, S_N

A star graph of size N is a complete bi-partite graph $K_{1,N\text{-}1}$. In particular, it has a center node that is connected to any other node, and all the nodes are connected only to the center node. Figure 7.1c provides a visual representation of S_9. Assume that the center node is labeled as 1. Let i, j and k be three nodes other than the center node. Then we have $\mathcal{N}(i) = \mathcal{N}(j) = \mathcal{N}(k) = \{1\}$. Therefore,

$$\mathcal{N}(i) \setminus \{j, k\} = \mathcal{N}(j) \setminus \{i, k\} = \mathcal{N}(k) \setminus \{i, j\} = \{1\}, \quad (7.39)$$

which shows that an unweighted star graph of an arbitrary size ($N \geq 3$) has a 3-sparse eigenvector, which in particular implies that it has a 2-sparse eigenvector as well (Theorem 7.10).

Cycle Graph, C_N

A cycle graph of size N contains a single cycle through all nodes. Figure 7.1d provides a visual representation of C_8. Notice that $C_2 = K_2$, $C_3 = K_3$, $C_4 = K_{2,2}$, hence they have 2-sparse eigenvectors as shown above. For $N \geq 5$, C_N does *not* have a pair of nodes that satisfy (7.34). Therefore, a cycle graph for $N \geq 5$ does *not* have a 2-sparse eigenvector, which, in particular, implies that it does *not* have a 3-sparse eigenvector as well (Theorem 7.10). In fact, it can be formally shown that an eigenvector of a cycle graph of size N has at least $N/2$ non-zero values (see Section 7.7.1).

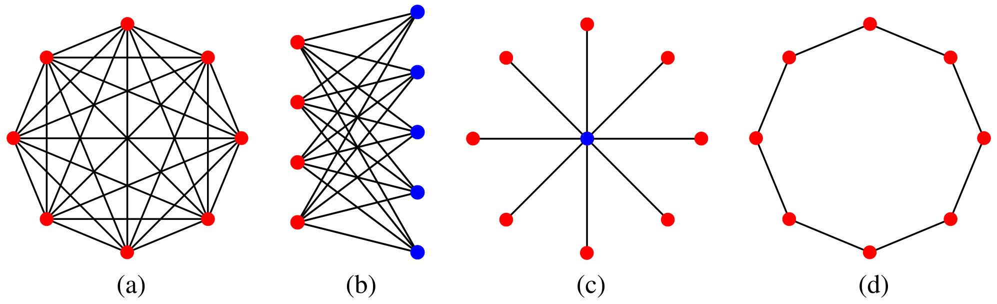


Figure 7.1: a) K_8, complete graph of size 8, b) S_9, star graph of size 9, c) C_8, cycle graph of size of 8.

Above examples are carefully selected to point out an important observation: *sparsity of the graph and existence of sparse eigenvectors do not imply each other*. This follows from the following three facts:

1) A complete graph is dense, yet it has a sparse eigenvector.

2) A cycle graph is sparse, yet it does not have a sparse eigenvector.

3) A star graph is sparse, and it has a sparse eigenvector.

One can also use Theorem 7.8 to find a sparse GFB of a given graph. Existence of a pair of nodes that satisfy (7.34) guarantees the existence of a 2-sparse eigenvector. When there is more than one pair, it is possible to find various 2-sparse eigenvectors. Even though those eigenvectors may not be orthonormal to each other they provide a sparse GFB. In fact, N-1 eigenvectors of the graph Laplacian of a complete graph of size N can be selected to be 2-sparse. These eigenvectors will be linearly independent, but not orthonormal. In this case, GFB has only 3N-2 nonzero entries. Details of these will not be elaborated here, and deserve an independent study.

It is important to notice that the condition in (7.33) is purely algebraic, and does not require any numerical computation. Therefore, Theorems 7.8 and 7.11 are *not* subject to the problems discussed in Section 7.4.2. In order to find a pair of nodes with the property in (7.33), one can check every pair in a brute-force manner, which results in $\binom{N}{2}$ tests in total. Therefore, the complexity of verifying that a graph has a pair of nodes with the property (7.33) is at most $O(N^2)$. However, there may exist more efficient search algorithms for this purpose.

7.7 Examples of Uncertainty Bounds

7.7.1 Standard Examples from Graph Theory

Circulant Graphs

A graph is said to be circulant when its adjacency matrix is a circulant matrix under suitable permutation of the node numbering [56]. This is a broad family including cyclic graphs (directed or undirected), complete graphs, complete bi-partite graphs, and more. The directed cyclic graph of size N, whose adjacency matrix is given as

$$\mathbf{C}_N = \begin{bmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix} \in \mathbb{R}^N, \tag{7.40}$$

is particularly important since it relates the graph signal processing to classical signal processing [152, 151].

The adjacency matrix of a circulant graph is a circulant matrix (with suitable permutation of vertices), and can be diagonalized by the DFT matrix:

$$\mathbf{A} = \mathbf{W}_N^{\mathrm{H}} \, \mathbf{\Lambda} \, \mathbf{W}_N, \tag{7.41}$$

for some diagonal $\mathbf{\Lambda}$, where $\mathbf{W}_N$ is the normalized DFT matrix of size N. Hence, the graph Fourier transform based on the adjacency matrix is $\mathbf{F} = \mathbf{W}_N$, and we have $||\mathbf{V}||_{\max}^{-1} = \sqrt{N}$. As a result, the strong uncertainty principle for circulant graphs of size N is $s_0(\mathbf{x}) \geq \sqrt{N}$.

As shown in [60], this is a tight bound when N is a perfect square. Consider the "picket fence" signal which has support $S = \{1, 1{+}\sqrt{N}, 1{+}2\sqrt{N}, \cdots, 1{+}N{-}\sqrt{N}\}$ with

$$\mathbf{x}_S = 1, \qquad \mathbf{x}_{\bar{S}} = 0. \tag{7.42}$$

Then we have $\widehat{\mathbf{x}} = \mathbf{F}\mathbf{x} = \mathbf{x}$. Notice that $|S| = \sqrt{N}$. As a result we have $s_0(\mathbf{x}) = \sqrt{N} = ||\mathbf{V}||_{\max}^{-1}$, that is, strong ℓ_0 uncertainty is achieved (Corollary 7.3).

For the weak uncertainty we have $p_0(\mathbf{x}) \geq \sqrt{N}$, that is, $||\mathbf{x}||_0 \, ||\widehat{\mathbf{x}}||_0 \geq N$, where $\widehat{\mathbf{x}}$ corresponds to DFT of $\mathbf{x}$. This is a well-known uncertainty result given in [54]. Unlike the strong uncertainty, weak uncertainty bound can be achieved for any N. Let $\mathbf{x}$ be an impulse, then $\widehat{\mathbf{x}}$ will have no zero elements, resulting in $||\mathbf{x}||_0 \, ||\widehat{\mathbf{x}}||_0 = N$.

If the graph is unweighted, then circulant graphs are regular (each node has the same degree). In this case the graph Laplacian can be written as $\mathbf{L} = d\mathbf{I} - \mathbf{A}$, where d is the degree of each node. Therefore, $\mathbf{L}$ is also a circulant matrix, and diagonalizable by $\mathbf{W}_N$. As a result, strong uncertainty bound based on the graph Laplacian is also tight.

Cycle Graph

In this part we will focus on the *undirected* cyclic graph as visually shown in Figure 7.1d. Eigenvalues of the Laplacian of a cyclic graph are given as $\lambda_k = 2 - 2\cos(2\pi k/N)$ for $0 \leq k \leq N\text{-}1$ [29]. Notice that $\lambda_0 = 0$ is not a repeated eigenvalue. However, other eigenvalues have the property $\lambda_k = \lambda_{N\text{-}k}$ for $k \geq 1$. Therefore, the Laplacian of a cycle graph has 2-dimensional eigenspaces. Let $\mathcal{S}_k$ denote the 2-dimensional eigenspace of the Laplacian corresponding to eigenvalue λ_k. Let $\mathbf{w}_k$ denote the k^{th} column of $\mathbf{W}_N^{\mathrm{H}}$. Then $\mathbf{U}_k = [\mathbf{w}_k \;\; \mathbf{w}_{N\text{-}k}]$ spans the eigenspace $\mathcal{S}_k$ since the Laplacian is diagonalized by $\mathbf{W}_N$. Notice that $|(\mathbf{W}_N)_{i,j}| = 1/\sqrt{N}$ for all pairs of (i, j). As a result, each row of $\mathbf{U}_k$ has ℓ_2 norm of $\sqrt{2/N}$. Then, Theorem 7.5 gives that $m(\mathcal{S}_k) = \sqrt{2/N}$. Using (7.31) we conclude that the total number of nonzero elements of an eigenvector in $\mathcal{S}_k$ can be at least $(m(\mathcal{S}_k))^{-2} = N/2$. Since this is true for any eigenspace, *any eigenvector of the Laplacian of a cycle graph (of size N) has at least $N/2$ nonzero values.*

As discussed in the previous sub-section, we have $s_0(\mathbf{x}) \geq \sqrt{N}$ for all nonzero signals on a cycle graph when GFB selected as $\mathbf{W}_N^{\mathrm{H}}$. When we consider the additive uncertainty of elements of GFB, as in (7.26), we get $s_0(\mathbf{v}_i) \geq (N\text{+}2)/4$ since each eigenvector has at least $N/2$ nonzeros. As a result, elements of GFB are not useful candidates to achieve the bound $s_0(\mathbf{x}) \geq \sqrt{N}$ with equality. This is because eigenvectors of the cycle graph are not sparse.

Complete Graph

Being a circulant graph, GFB of a complete graph *can be* selected as $\mathbf{W}_N^{\mathrm{H}}$. With this selection, the additive uncertainty bound is given as $s_0(\mathbf{x}) \geq \sqrt{N}$. It should be noted that the Laplacian of a complete graph (of size N) has only two distinct eigenvalues: 0 with multiplicity 1, N with multiplicity $N - 1$. One can select different set of vectors to span the N-1 dimensional eigenspace. In fact, as discussed in Section 7.6, one of these vectors can be selected to be 2-sparse. With such a selection, by virtue of Theorem 7.11, the uncertainty bound is given as $s_0(\mathbf{x}) \geq 3/2$, which is significantly different from the one when $\mathbf{W}_N^{\mathrm{H}}$ is used as GFB. At this point we are not favoring one selection of GFB over another. The sole purpose of this example is to show that selection of the GFB is an important issue in the presence of repeated eigenvalues.

7.7.2 M-Block Cyclic Graphs

In Chapter 6, it is shown that M-Block cyclic graphs play an important role in the development of multirate processing of graph signals. When we assume that all the edges have unit weights, the adjacency matrix of an M-Block cyclic graph of size N can be written as: $\mathbf{A} = \mathbf{C}_M \otimes \left(\mathbf{1}_{N/M}\ \mathbf{1}_{N/M}^{\mathrm{T}}\right)$, where $\mathbf{C}_M$ is given in (7.40) and $\mathbf{1}_N$ is a vector of size N with all 1 entries. Notice that both $\mathbf{C}_M$ and $\mathbf{1}_{N/M}\ \mathbf{1}_{N/M}^{\mathrm{T}}$ are circulant matrices, hence they are diagonalizable by normalized DFT matrices of respective sizes. As a result, GFT based on the adjacency matrix (and the graph Laplacian since all the nodes have the same degree) can be selected as $\mathbf{F} = \mathbf{W}_M \otimes \mathbf{W}_{N/M}$. Notice that $\mathbf{V} = \mathbf{F}^{\mathrm{H}}$ is unitary and $\|\mathbf{V}\|_{\max}^{-1} = \sqrt{N}$.

As an example consider the case $N = 9$ and $M = 3$, and consider the following signal $\mathbf{x} = [1\ 1\ 1]^{\mathrm{T}} \otimes [1\ 0\ 0]^{\mathrm{T}}$. The GFT of this signal, $\widehat{\mathbf{x}}$, is given as

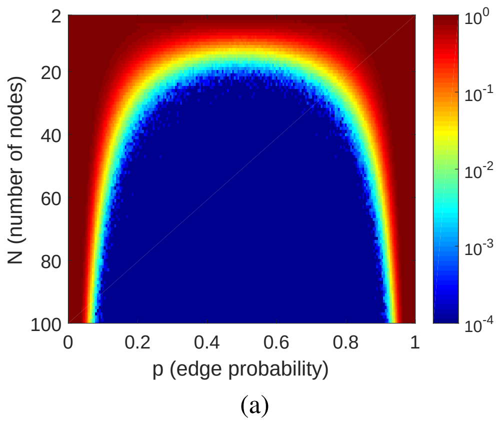
$$\widehat{\mathbf{x}} = \mathbf{W}_3[1\ 1\ 1]^{\mathrm{T}} \otimes \mathbf{W}_3[1\ 0\ 0]^{\mathrm{T}} = [1\ 0\ 0]^{\mathrm{T}} \otimes [1\ 1\ 1]^{\mathrm{T}}. \tag{7.43}$$

Hence, we have $\|\mathbf{x}\|_0 = \|\widehat{\mathbf{x}}\|_0 = 3$, and $s_0(\mathbf{x}) = 3 = \|\mathbf{V}\|_{\max}^{-1}$. Therefore, strong uncertainty bound is achieved. In general, let N be a perfect square and $M = \sqrt{N}$. *Then for any M-Block cyclic graph of size N with unit weights, we can find a signal that achieves the strong uncertainty bound.*

7.7.3 Erdős-Rényi Graphs

An Erdős-Rényi graph $G(N, p)$ is a simple graph of N nodes where an edge between a pair of nodes appears randomly and independently with probability p [134, 83]. In Figure 7.2a we empirically compute the probability of a $G(N, p)$ having a pair of nodes satisfying the condition in (7.34). Notice that if a graph has such a pair of nodes then the same pair satisfies (7.34) on the complement of the graph as well. This is due to the equality condition in (7.34) that remains satisfied when all the edges are complemented. Further notice that complement of a $G(N, p)$ is a $G(N, 1\text{-}p)$ graph. As a result $G(N, p)$ and $G(N, 1\text{-}p)$ have the same probability of having a pair of nodes satisfying (7.34). This explains the symmetry of Figure 7.2a around $p = 1/2$.

It is important to note that Theorem 7.8 specifically considers the case of connected graphs since it is trivial to find sparse eigenvectors in disconnected graphs (see (7.32)). However, a $G(N,p)$ tends to be disconnected when p is small. In fact $p < log(N)/N$ results in (almost surely) isolated vertices, and $p > log(N)/N$ guarantees (almost surely) $G(N,p)$ to be connected [83]. In order to get rid of the trivial cases we need to consider the probability of $G(N,p)$ having a pair of nodes with (7.34) *and* being connected. Experimental computation of this probability is given in Figure 7.2b. Notice that connectivity is not preserved under complementation, hence Figure 7.2b is not symmetric around $p = 1/2$.



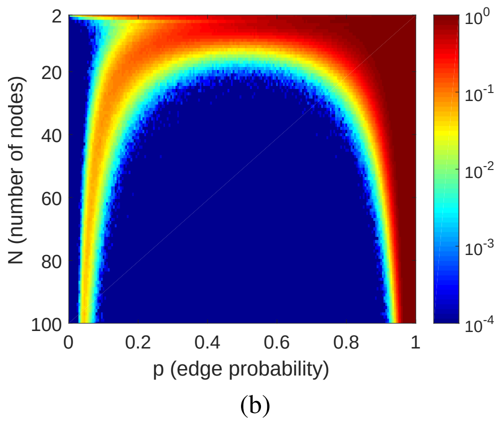


Figure 7.2: Probability of $G(N,p)$ a) having a pair of nodes satisfying (7.34), b) being connected *and* having a pair of nodes satisfying (7.34). Probabilities are obtained via averaging over 10^4 experiments, hence the lowest observed probability is 10^{-4}.

Figure 7.2b shows the existence of connected Erdős-Rényi graphs with 2-sparse eigenvectors. Figure 7.2b also suggests that as N gets larger it is less likely to find such graphs, which can be explained as follows. The study in [191] states that the ℓ_∞-norm of any unit eigenvector of $G(N,p)$ is almost surely $o(1)$ for $p = \omega(log(N)/N)$, where $o(\cdot)$ and $\omega(\cdot)$ denotes little-o and little-omega notations, respectively. That is, as $N \to \infty$ we have $\|\mathbf{v}\|_\infty \to 0$ for all $\mathbf{v}$. Therefore, $\|\mathbf{V}\|_{\max} \to 0$, hence, $\|\mathbf{V}\|_{\max}^{-1} \to \infty$. Since the uncertainty bound in (7.10) goes to infinity we do not expect to find 2-sparse eigenvectors in connected Erdős-Rényi graphs for large values of N.

7.7.4 Real World Examples

In the following we will use the term $\lambda^{(k)}$ to denote that the eigenvalue λ has multiplicity k.

Minnesota Road Graph

In this part, we will consider the Minnesota road graph visualized in Figure 6.22. We use the data publicly available in [128]. This graph has 2642 nodes in total where 2 nodes are disconnected to the rest of the graph. Since a road graph is expected to be connected, we disregard those two nodes. Here each node is an intersection, and $a_{i,j} = 1$ if there is a road connecting the intersections, otherwise $a_{i,j} = 0$. There are total of 3302 undirected unweighted edges. The graph is simple and connected, $\mathbf{A}$ and $\mathbf{L}$ are symmetric matrices (in particular, diagonalizable), hence $\mathbf{V}_A$ and $\mathbf{V}_L$ can be selected to be unitary.

For the Minnesota road graph, both $\mathbf{A}$ and $\mathbf{L}$ have repeated eigenvalues. As a result, $\mathbf{V}_A$ and $\mathbf{V}_L$ are *not* unique. In fact, the adjacency matrix has repeated eigenvalues of $-1^{(15)}$, $0^{(44)}$, and $1^{(13)}$. The graph Laplacian has repeated eigenvalues of $0.3820^{(2)}$, $1^{(10)}$, $2^{(7)}$, $2.6180^{(2)}$, and $3^{(6)}$. In order to minimize the uncertainty bound given by Corollary 7.3, we can select $\mathbf{V}_A$ and $\mathbf{V}_L$ such that $\|\mathbf{V}_A\|_{\max}$ and $\|\mathbf{V}_L\|_{\max}$ are maximized. This idea is discussed in Section 7.4, and the closed-form solution for such a selection is provided by Theorem 7.6. Via numerical evaluation of Theorem 7.6 on the Minnesota road graph, we obtain the following:

$$0.7071 = \max_{\mathbf{V}} \ \|\mathbf{V}\|_{\max} \quad \text{s.t.} \quad \mathbf{A} = \mathbf{V}\mathbf{\Lambda}_A\mathbf{V}^{\mathrm{H}}, \tag{7.44}$$

$$0.8343 = \max_{\mathbf{V}} \ \|\mathbf{V}\|_{\max} \quad \text{s.t.} \quad \mathbf{L} = \mathbf{V}\mathbf{\Lambda}_L\mathbf{V}^{\mathrm{H}}. \tag{7.45}$$

Due to Corollaries 7.2 and 7.3, when $\mathbf{V}_A$ is selected to be the GFB, (7.44) gives the following uncertainty bounds $s_0(\mathbf{x}) \geq p_0(\mathbf{x}) \geq \|\mathbf{V}_A\|^{-1}_{\max} = 1.4142$. When $\mathbf{V}_L$ is selected to be the GFB, (7.45) gives the following uncertainty bounds $s_0(\mathbf{x}) \geq p_0(\mathbf{x}) \geq \|\mathbf{V}_L\|^{-1}_{\max} = 1.1987$.

It is important to remember that $s_0(\mathbf{x})$ can have values only on a discrete set, namely, $s_0(\mathbf{x}) = k/2$ for some integer $k \geq 2$. As a result, for both selection of GFB, signals on the Minnesota road graph *cannot* attain the uncertainty bound in Corollary 7.3 in a strict sense. However, by rounding-off the value of both $\|\mathbf{V}_A\|^{-1}_{\max}$ and $\|\mathbf{V}_L\|^{-1}_{\max}$ to the next attainable value of $s_0(\mathbf{x})$, we get

$$s_0(\mathbf{x}) \geq 3/2, \tag{7.46}$$

for the strong uncertainty bound for both selection of GFB.

Even though (7.46) is a valid bound, Corollary 7.3 and Theorem 7.6 gives no further information about existence and characterization of a signal that achieves the bound.

At this point it is quite interesting to observe that the bound in (7.46) is the same as the bound provided by Theorem 7.11 (for $\mathbf{V}_L$ as GFB), which requires existence of a pair of nodes with the property in (7.34). In fact, the Minnesota road graph *does* have 6 different pairs of nodes with the property in (7.34). These pairs are visualized in Figure 7.3. As a result, the bound in (7.46) is *tight*, and the signals that achieve the bound are defined by the pairs of nodes in Figure 7.3 (see Theorem 7.11). It should be noted that tightness of (7.46) is valid when $\mathbf{V}_L$ is *selected* to include at least one 2-sparse eigenvector generated by the pairs in Figure 7.3.

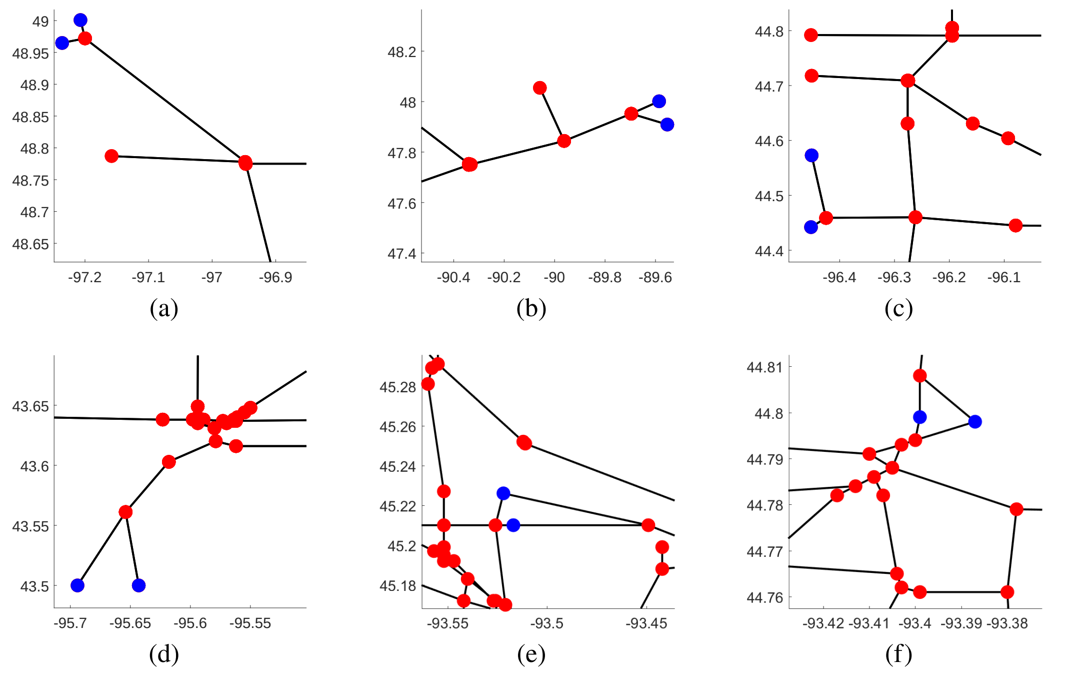


Figure 7.3: Pair of nodes in Minnesota road graph that result in 2-sparse eigenvectors. Axes represent the geographical coordinates. The pairs that satisfy the condition in (7.33) are colored in blue. Notice that the pairs in (a)-(d) generate eigenvectors with eigenvalue 1, and the pairs in (e)-(f) generate eigenvectors with eigenvalue 2. Axes represent the geographical location of the intersections. (See Theorem 7.8.)

By using a brute-force search over the graph, we have found that the graph does *not* have a triplet of nodes with the property in (7.35), hence the graph does not have a 3-sparse eigenvector. However, it *does* have 6 different pairs of nodes with the property in (7.34). Hence, the graph has 2-sparse eigenvectors. Notice that the eigenvectors generated by the nodes in Figure 7.3a-7.3d are orthogonal to each other and have eigenvalue 1. Using linear combinations of 2-sparse eigenvectors, we can verify that the graph has 4, 6, and 8-sparse eigenvectors as well.

Co-appearance Network

In this example, we will consider the co-appearance network of characters in the famous novel Les Misérables by Victor Hugo [103]. Data is publicly available in [133]. This is an undirected but *weighted* graph, where two characters are connected if they appear in the same scene, and the weight of an edge is the total number of co-appearances through the novel. See Figure 7.4 for a visual illustration.

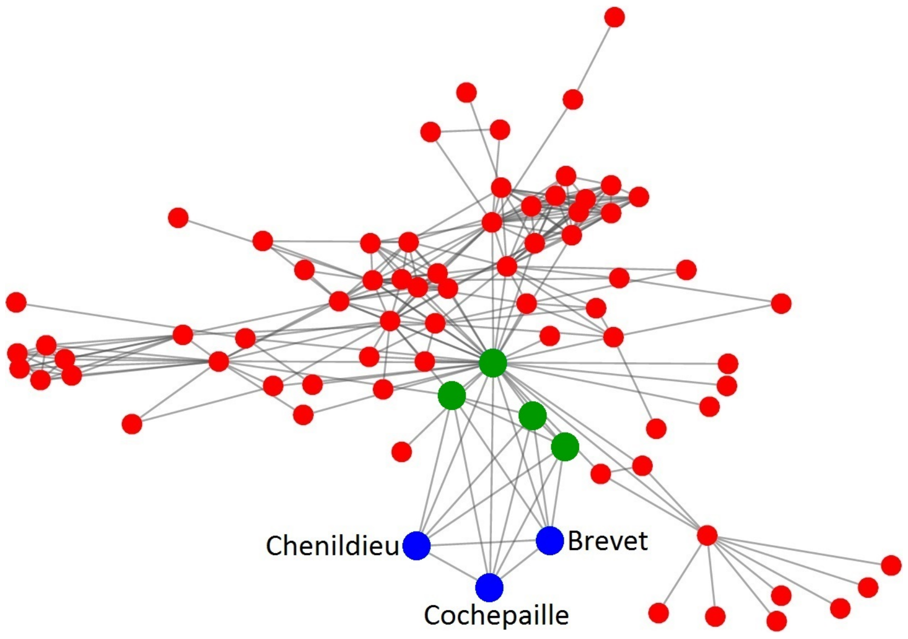


Figure 7.4: Co-appearance network of characters in the famous novel Les Misérables by Victor Hugo. In this example 2-sparse eigenvector implies the existence of characters (of the novel) appearing together throughout the scenes.

The graph has 77 nodes and 254 (weighted) edges in total. The Laplacian of the graph has repeated eigenvalues of $1^{(9)}$, $13^{(2)}$, and $28^{(2)}$. As a result, GFB with respect to the graph Laplacian, $\mathbf{V}_L$, is not unique. In order to minimize the bound given in Corollary 7.3, we use Theorem 7.6 and obtain the following result:

$$0.9398 = \max_{\mathbf{V}} \ \|\mathbf{V}\|_{\max} \quad \text{s.t.} \quad \mathbf{L} = \mathbf{V}\,\boldsymbol{\Lambda}_L\,\mathbf{V}^{\mathrm{H}}. \tag{7.47}$$

When $\mathbf{V}_L$ is selected to be the GFB, (7.47) gives the following uncertainty bounds (Corollaries 7.2 and 7.3) $s_0(\mathbf{x}) \geq p_0(\mathbf{x}) \geq \|\mathbf{V}_L\|_{\max}^{-1} = 1.0641$. Due to discrete nature of $s_0(\mathbf{x})$ it is clear that this bound cannot be satisfied with equality. When $\|\mathbf{V}_L\|_{\max}^{-1}$ is rounded-off to the next attainable value of $s_0(\mathbf{x})$, we get the same bounds as in (7.46). At this point we can use Theorem 7.11 to find signals (if there is any) that achieve the bound in (7.46).

In a co-appearance graph, pair of nodes with the condition in (7.33) has a meaningful interpretation. If two characters always appear simultaneously, they will have the same number of co-appearances with other characters, which implies the condition in (7.33) mathematically. As an example, consider characters “Brevet”, “Chenildieu”, and “Cochepaille” of the novel Les Misérables. They are three witnesses in Champmathieu’s trial, and appear simultaneously through the court scenes. Nodes (of the graph) that correspond to any two of these three characters satisfy the condition in (7.33), which implies that the graph Laplacian has a 2-sparse eigenvector, and $s_0(\mathbf{x}) \geq 3/2$ is a tight uncertainty bound when $\mathbf{V}_L$ is selected as GFB.

7.8 Concluding Remarks

In this chapter, we studied the concept of uncertainty principle for signals defined over graphs. Unlike existing studies we took a non-local and discrete approach, where the vertex and the spectral domain spreads of a signal are defined as the number of nonzero elements of the signal and its GFT, respectively. We derived a lower bound for the total number of nonzero elements in both domains (on the graph and in the GFB) and showed that a signal and its corresponding GFT cannot be arbitrarily sparse simultaneously. Based on this, we obtained a new form of uncertainty principle for graph signals. When the graph has repeated eigenvalues we explained that GFB is not unique, and the derived lower bound can have different values depending on the selected GFB. We provided a constructive method to find a GFB that yields the smallest uncertainty bound. In order to find the signals that achieve the derived lower bound we considered sparse eigenvectors of the graph. We showed that the graph Laplacian has a 2-sparse eigenvector (which is also known as a Faria vector) if and only if there exists a pair of nodes with the same neighbors. When this happens, the uncertainty bound is very low and the 2-sparse eigenvectors achieve this bound. In addition, we provided the necessary and sufficient condition for the existence of 3-sparse eigenvectors. We further showed that, for unweighted graphs, the existence of a 3-sparse eigenvector implies the existence of a 2-sparse eigenvector. We also provided counter-examples to show that this result does not extend to arbitrary sparsity. We presented examples of both classical and real-world graphs with 2 and 3-sparse eigenvectors. We also discussed that, in some examples, the neighborhood structure has a meaningful interpretation.

7.9 Appendices

7.9.1 Proof of Theorem 7.1

First notice that $\mathbf{x} = \sum_{i=1}^{N} x_i \, \mathbf{e}_i$ where $\mathbf{e}_i$ is the i^{th} vector of the canonical basis, and x_i is the i^{th} element of $\mathbf{x}$. Let $\mathbf{f}_j^{\mathrm{H}}$ denote the j^{th} row of $\mathbf{F}$. For the j^{th} element of the GFT of $\mathbf{x}$ we have

$$|\widehat{x}_j| = |\mathbf{f}_j^{\mathrm{H}} \, \mathbf{x}| = \left| \sum_{i \in \mathcal{S}} x_i \, \mathbf{f}_j^{\mathrm{H}} \, \mathbf{e}_i \right|, \tag{7.48}$$

where $\mathcal{S}$ denotes the support (set of nonzero indices) of the signal $\mathbf{x}$. Notice that $|\mathcal{S}| = ||\mathbf{x}||_0$. We can upper bound $|\widehat{x}_j|$ as

$$|\widehat{x}_j| = \left| \sum_{i \in \mathcal{S}} x_i \, \mathbf{f}_j^{\mathrm{H}} \, \mathbf{e}_i \right| \leq \sum_{i \in \mathcal{S}} \left| x_i \, \mathbf{f}_j^{\mathrm{H}} \, \mathbf{e}_i \right|, \tag{7.49}$$

using the triangular inequality. Using Cauchy-Schwarz inequality, this can be further bounded as

$$|\widehat{x}_j| \leq \Big(\sum_{i \in \mathcal{S}} |x_i|^2 \Big)^{1/2} \Big(\sum_{i \in \mathcal{S}} |\mathbf{f}_j^{\mathrm{H}} \mathbf{e}_i|^2 \Big)^{1/2}, \tag{7.50}$$

$$\leq ||\mathbf{x}||_2 \left(|\mathcal{S}| \; ||\mathbf{F}||_{\max}^2 \right)^{1/2} = ||\mathbf{x}||_2 \; ||\mathbf{x}||_0^{1/2} \; ||\mathbf{F}||_{\max}, \tag{7.51}$$

where we use the fact that $\mathbf{f}_j^{\mathrm{H}} \, \mathbf{e}_i$ is the $(j,i)^{th}$ element of $\mathbf{F}$ whose magnitude is upper bounded by $||\mathbf{F}||_{\max}$. Now consider the ℓ_2 norm of $\widehat{\mathbf{x}} = \mathbf{F}\mathbf{x}$:

$$||\widehat{\mathbf{x}}||_2^2 = \sum_{j \in K} |\widehat{x}_j|^2 \leq ||\widehat{\mathbf{x}}||_0 \; ||\mathbf{x}||_2^2 \; ||\mathbf{x}||_0 \; ||\mathbf{F}||_{\max}^2, \tag{7.52}$$

where K denotes the support of $\widehat{\mathbf{x}}$, $||\widehat{\mathbf{x}}||_0 = |K|$. Hence we have,

$$\frac{1}{||\widehat{\mathbf{x}}||_0 \; ||\mathbf{x}||_0} \leq \left(||\mathbf{F}||_{\max} \frac{||\mathbf{x}||_2}{||\widehat{\mathbf{x}}||_2} \right)^2. \tag{7.53}$$

Notice that $\max_{\mathbf{x}} ||\mathbf{x}||_2 / ||\widehat{\mathbf{x}}||_2 = \max_{\mathbf{y}} ||\mathbf{F}^{-1}\mathbf{y}||_2 / ||\mathbf{y}||_2 = ||\mathbf{F}^{-1}||_2$. Therefore,

$$\frac{1}{||\widehat{\mathbf{x}}||_0 \; ||\mathbf{x}||_0} \leq \left(||\mathbf{F}||_{\max} \frac{||\mathbf{x}||_2}{||\widehat{\mathbf{x}}||_2} \right)^2 \leq \left(||\mathbf{F}||_{\max} \; ||\mathbf{F}^{-1}||_2 \right)^2 \tag{7.54}$$

which implies $\sqrt{||\mathbf{x}||_0 \; ||\mathbf{F}\mathbf{x}||_0} \geq \left(||\mathbf{F}^{-1}||_2 \; ||\mathbf{F}||_{\max} \right)^{-1}$.

7.9.2 Proof of Theorem 7.3

Notice that $\mathbf{F}$ is an invertible matrix, therefore $||\widehat{\mathbf{x}}||_2 / ||\mathbf{x}||_2$ is lower and upper bounded as follows:

$$||\mathbf{F}^{-1}||_2^{-2} \leq \frac{||\widehat{\mathbf{x}}||_2^2}{||\mathbf{x}||_2^2} \leq ||\mathbf{F}||_2^2. \tag{7.55}$$

Therefore we have

$$\|\mathbf{F}^{-1}\|_2^{-2}\,\|\mathbf{x}\|_2^2 \leq \|\widehat{\mathbf{x}}\|_2^2 \;=\; \widehat{\mathbf{x}}^{\mathrm{H}}\,\mathbf{F}\,\mathbf{x} = \sum_{i,j} \widehat{x}_i^*\, F_{i,j}\, x_j, \tag{7.56}$$

$$\leq \sum_{i,j} \big|\widehat{x}_i^*\, F_{i,j}\, x_j\,\big| \leq \sum_{i,j} |\widehat{x}_i|\; \|\mathbf{F}\|_{\max}\, |x_j|, \tag{7.57}$$

$$= \|\widehat{\mathbf{x}}\|_1\; \|\mathbf{x}\|_1\; \|\mathbf{F}\|_{\max}, \tag{7.58}$$

where we use the fact that $|F_{i,j}| \leq \|\mathbf{F}\|_{\max}$ for all (i, j) in (7.57). Notice that taking square-root of both sides and re-arranging the terms in (7.58) give the result in (7.17).

7.9.3 Proof of Theorem 7.4

We start with the following change of variables,

$$\Big(p_1(\mathbf{x})\Big)^2 = \frac{\|\mathbf{x}\|_1\, \|\mathbf{F}\,\mathbf{x}\|_1}{\|\mathbf{x}\|_2^2} = \|\bar{\mathbf{x}}\|_1\, \|\mathbf{F}\,\bar{\mathbf{x}}\|_1, \tag{7.59}$$

where we define $\bar{\mathbf{x}} = \mathbf{x}/\|\mathbf{x}\|_2$.

Given any two vectors $\mathbf{x}$ and $\mathbf{y}$ with $\|\mathbf{x}\|_2 = \|\mathbf{y}\|_2 = 1$, we have the following relation (page 20 of [60]):

$$\|\mathbf{x}\|_1\, \|\mathbf{y}\|_1 \leq \sqrt{\|\mathbf{x}\|_0\, \|\mathbf{y}\|_0}. \tag{7.60}$$

Notice that we have $\|\bar{\mathbf{x}}\|_2 = 1$. Since we have assumed that the GFB is unitary, we further have $\|\mathbf{F}\,\bar{\mathbf{x}}\|_2 = 1$. Then (7.60) gives the following:

$$\|\bar{\mathbf{x}}\|_1\, \|\mathbf{F}\,\bar{\mathbf{x}}\|_1 \leq \sqrt{\|\bar{\mathbf{x}}\|_0\, \|\mathbf{F}\,\bar{\mathbf{x}}\|_0}. \tag{7.61}$$

Notice that left-hand-side of (7.61) is equal to $(p_1(\mathbf{x}))^2$ due to (7.59). Remember that $p_0(\mathbf{x})$ is scale-invariant. As a result we have $\sqrt{\|\bar{\mathbf{x}}\|_0\, \|\mathbf{F}\,\bar{\mathbf{x}}\|_0} = \sqrt{\|\mathbf{x}\|_0\, \|\mathbf{F}\,\mathbf{x}\|_0}$, which shows that right-hand-side of (7.61) is equal $p_0(\mathbf{x})$. Hence, we conclude that $(p_1(\mathbf{x}))^2 \leq p_0(\mathbf{x})$.

7.9.4 Proof of Theorem 7.5

Let $\mathbf{U}$ and $\mathbf{Q}$ be two matrices with orthonormal columns such that $span(\mathbf{U}) = span(\mathbf{Q}) = \mathcal{S}$. Since both span the same sub-space, we can write $\mathbf{Q} = \mathbf{U}\,\mathbf{X}$ for some unitary matrix $\mathbf{X}$ of size M. Then we can write $m(\mathcal{S})$ as

$$m(\mathcal{S}) = \max_{\mathbf{X}\in\mathbb{C}^{M\times M}} \;\|\mathbf{U}\mathbf{X}\|_{\max} \quad \text{s.t.} \quad \mathbf{X}^{\mathrm{H}}\,\mathbf{X} = \mathbf{I}, \tag{7.62}$$

where $\mathbf{U}$ is an arbitrary matrix with orthonormal columns that span $\mathcal{S}$. Let $\mathbf{x}_i$ be the i^{th} column of $\mathbf{X}$, and $\mathbf{u}_j^{\mathrm{H}}$ be the j^{th} row of $\mathbf{U}$. Then we can write (7.62) as

$$\begin{aligned} m(\mathcal{S}) &= \max_{\substack{1\leq j\leq N\\ 1\leq i\leq M\\ \mathbf{x}_i}} \left|\mathbf{u}_j^{\mathrm{H}}\mathbf{x}_i\right| \quad \text{s.t.} \quad \mathbf{x}_i^{\mathrm{H}}\mathbf{x}_j = \delta_{i,j} && (7.63)\\ &= \max_{\substack{1\leq j\leq N\\ \mathbf{x}_1}} \left|\mathbf{u}_j^{\mathrm{H}}\mathbf{x}_1\right| \quad \text{s.t.} \quad \|\mathbf{x}_1\|_2 = 1 && (7.64)\\ &= \max_{1\leq j\leq N} \|\mathbf{u}_j^{\mathrm{H}}\|_2 = \max_{1\leq j\leq N} \sqrt{\left(\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\right)_{j,j}}, && (7.65) \end{aligned}$$

where we assume (w.l.o.g.) in (7.63) that $\mathbf{x}_1$ is the vector that achieves the maximum. Furthermore, other $\mathbf{x}_i$'s will have the additional constraint of being orthonormal to $\mathbf{x}_1$. As a result, $\mathbf{x}_i$'s for $2 \leq i \leq M$ *cannot* produce a larger inner product. Further, once the optimal $\mathbf{x}_1$ is selected, the remaining $\mathbf{x}_i$'s can be selected arbitrarily as long as they are orthonormal to each other. This can be done via Gram-Schmidt process. In (7.64) we use the fact that inner product is maximized when the vectors are aligned with each other. Equality in (7.65) follows from the fact that ℓ_2-norm-square of a row is the corresponding diagonal entry of the outer product.

7.9.5 Proof of Theorem 7.7

We prove (1) implies (2): Assume that the graph has an isolated node. According to (7.32), there exists a 1-sparse eigenvector of the graph Laplacian.

We now prove that (2) implies (1): Let $\mathbf{v}$ be the 1-sparse eigenvector. Without loss of generality assume that the first index is nonzero $v_1 = 1$ and the rest is zero. Therefore $\mathbf{Lv}$ is equivalent to the first column of $\mathbf{L}$. That is, $\mathbf{Lv} = [d_1 \;\; \text{-}\mathbf{a}_{r,1}^{\mathrm{T}}]^{\mathrm{T}} = \lambda[1 \;\; \mathbf{0}^{\mathrm{T}}]^{\mathrm{T}}$, where $\mathbf{a}_{r,1} \in \mathbb{R}^{N\text{-}1}$ is the vector that denotes the adjacency of node 1, and $d_1 = \|\mathbf{a}_{r,1}\|_1$ is the degree of node 1. Therefore we have $\mathbf{a}_{r,1} = \mathbf{0}$, hence $d_1 = 0$. Since edge weights are non-negative, node 1 is an isolated node.

Next we will prove that (2) implies (3): Let $\mathbf{v}$ be a 1-sparse eigenvector of the graph Laplacian. Then, $\mathbf{v}$ *can* be selected to be an element of the GFT. In this case, $\|\mathbf{v}\|_0 = \|\mathbf{F}\,\mathbf{v}\|_0 = 1$, hence $s_0(\mathbf{v}) = 1$.

Finally, we prove (3) implies (2): Let $s_0(\mathbf{x}) = 1$ for some $\mathbf{x} \neq 0$. Since $\|\mathbf{x}\|_0 \geq 1$ for any nonzero signal, we must have $\|\mathbf{x}\|_0 = \|\mathbf{F}\,\mathbf{x}\|_0 = 1$. $\|\mathbf{F}\,\mathbf{x}\|_0 = 1$ implies that $\mathbf{x}$ is an eigenvector of the graph Laplacian and $\|\mathbf{x}\|_0 = 1$ implies that $\mathbf{x}$ is 1-sparse. Hence the graph Laplacian has a 1-sparse eigenvector.

7.9.6 Proof of Theorem 7.8

Assume that the graph Laplacian of a connected graph has a two-sparse eigenvector **v** with nonzero eigenvalue. Due to permutation invariance of the node labels, without loss of any generality assume that the first two indices are nonzero, that is, $v_1 \neq 0$ and $v_2 \neq 0$, but $v_i = 0$ for $i \geq 3$.

For a connected graph, notice that the all-1 vector is the only eigenvector of the graph Laplacian with the zero eigenvalue. Since the graph Laplacian is a symmetric matrix the eigenspaces are orthogonal to each other. Therefore the 2-sparse eigenvector (with nonzero eigenvalue) **v** is orthogonal to the all-1 vector, which implies that $v_1 + v_2 = 0$. Then, we can select $v_1 = -v_2 = 1$ without loss of any generality.

Let **A** denote the adjacency matrix of the graph. We have

$$\mathbf{A} = \begin{bmatrix} 0 & a_{1,2} & \mathbf{a}_{r,1}^{\mathrm{T}} \\ a_{2,1} & 0 & \mathbf{a}_{r,2}^{\mathrm{T}} \\ \mathbf{a}_{r,1} & \mathbf{a}_{r,2} & \mathbf{A}_r \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} d_1 & -a_{1,2} & -\mathbf{a}_{r,1}^{\mathrm{T}} \\ -a_{2,1} & d_2 & -\mathbf{a}_{r,2}^{\mathrm{T}} \\ -\mathbf{a}_{r,1} & -\mathbf{a}_{r,2} & \mathbf{L}_r \end{bmatrix}, \tag{7.66}$$

where $\mathbf{A}_r \in \mathbb{C}^{(N-2)\times(N-2)}$ and $\mathbf{L}_r \in \mathbb{C}^{(N-2)\times(N-2)}$ are the partitions of the adjacency matrix and the graph Laplacian, respectively. $\mathbf{a}_{r,1} \in \mathbb{R}^{N-2}$ is the vector that denotes the adjacency of node 1 except node 2. $\mathbf{a}_{r,2}$ is the same for node 2. Notice that $d_1 = a_{2,1} + \|\mathbf{a}_{r,1}\|_1$ and $d_2 = a_{1,2} + \|\mathbf{a}_{r,2}\|_1$. Then, consider the following:

$$\mathbf{Lv} = \begin{bmatrix} d_1 + a_{1,2} \\ -(a_{2,1} + d_2) \\ -\mathbf{a}_{r,1} + \mathbf{a}_{r,2} \end{bmatrix} = \lambda\, \mathbf{v} = \lambda \begin{bmatrix} 1 \\ -1 \\ \mathbf{0} \end{bmatrix}. \tag{7.67}$$

Therefore we have $\mathbf{a}_{r,1} = \mathbf{a}_{r,2}$, which in particular implies that $d_1 = d_2$ since $a_{1,2} = a_{2,1}$ (graph is undirected). Furthermore the corresponding eigenvalue is $\lambda = d_1 + a_{1,2}$. Since the graph is connected $d_1 > 0$, λ is nonzero. Notice that the condition $\mathbf{a}_{r,1} = \mathbf{a}_{r,2}$ is the same as the condition in (7.33).

Conversely, assume that there exist two nodes with the property in (7.33). Without loss of generality, assume $i = 1$ and $j = 2$, and let **v** be a 2-sparse vector with $v_1 = -v_2 = 1$. Then partition the graph Laplacian as in (7.66). Due to (7.33), we have $\mathbf{a}_{r,1} = \mathbf{a}_{r,2}$, and $d_1 = d_2$. Then we have $\mathbf{Lv} = \lambda\mathbf{v}$ with $\lambda = d_1 + a_{r,1}$. Therefore, **v** is a 2-sparse eigenvector of **L**. Further, the graph is connected $d_1 > 0$, hence $\lambda > 0$.

7.9.7 Proof of Theorem 7.9

Assume that the graph Laplacian of a connected graph has a three-sparse eigenvector **v** with nonzero eigenvalue. Due to permutation invariance of the node labels, without loss of generality assume that $v_1 \neq 0$, $v_2 \neq 0$, $v_3 \neq 0$ but $v_i = 0$ for $i \geq 4$.

For a connected graph, the all-1 vector is the only eigenvector of the graph Laplacian with the zero eigenvalue. Since the 3-sparse eigenvector (with nonzero eigenvalue) **v** is orthogonal to the all-1 vector we have $v_1 + v_2 + v_3 = 0$. By scaling the eigenvector, without loss of any generality, we can select $v_1 = 1$, $v_2 = \gamma$, and $v_3 = \text{-}1\text{-}\gamma$ for some γ, where $\gamma \neq 0$ and $\gamma \neq \text{-}1$ since the eigenvector **v** is exactly 3-sparse.

Similar to (7.66), the graph Laplacian can be partitioned as follows:

$$\mathbf{L} = \begin{bmatrix} d_1 & \text{-}a_{1,2} & \text{-}a_{1,3} & \text{-}\mathbf{a}_{R,1}^{\mathrm{T}} \\ \text{-}a_{2,1} & d_2 & \text{-}a_{2,3} & \text{-}\mathbf{a}_{R,2}^{\mathrm{T}} \\ \text{-}a_{3,1} & \text{-}a_{3,2} & d_3 & \text{-}\mathbf{a}_{R,3}^{\mathrm{T}} \\ \text{-}\mathbf{a}_{R,1} & \text{-}\mathbf{a}_{R,2} & \text{-}\mathbf{a}_{R,3} & \mathbf{L}_R \end{bmatrix}, \tag{7.68}$$

where $\mathbf{L}_R \in \mathbb{R}^{(N\text{-}3)\times(N\text{-}3)}$ is the partition of the Laplacian, and for $1 \leq i \leq 3$, $\mathbf{a}_{R,i} \in \mathbb{R}^{N\text{-}3}$ is the vector that denotes the adjacency of node i with the nodes $\{4, \cdots, N\}$. Since **v** is an eigenvector, it should satisfy the following eigenvalue equation:

$$\mathbf{L}\,\mathbf{v} = \begin{bmatrix} d_1 - \gamma\, a_{1,2} + (1{+}\gamma)\, a_{1,3} \\ -a_{2,1} + \gamma\, d_2 + (1{+}\gamma)\, a_{2,3} \\ -a_{3,1} - \gamma\, a_{3,2} - (1{+}\gamma)\, d_3 \\ -\mathbf{a}_{R,1} - \gamma\, \mathbf{a}_{R,2} + (1{+}\gamma)\, \mathbf{a}_{R,3} \end{bmatrix} = \lambda \mathbf{v} = \lambda \begin{bmatrix} 1 \\ \gamma \\ \text{-}(1{+}\gamma) \\ \mathbf{0} \end{bmatrix}. \tag{7.69}$$

This implies that $\mathbf{a}_{R,1} + \gamma\, \mathbf{a}_{R,2} - (1{+}\gamma)\, \mathbf{a}_{R,3} = \mathbf{0}$. This vector equation holds true if and only if it is satisfied element-wise. That is

$$a_{r,1} - a_{r,3} = \gamma\ (a_{r,3} - a_{r,2}) \qquad \forall\, r \in \{4, \cdots, N\}. \tag{7.70}$$

Remember that the graph is assumed to be unweighted, therefore, $a_{r,1}$, $a_{r,2}$, and $a_{r,3}$ are either 1 or 0. Hence, (7.70) can appear in $2^3 = 8$ different variations, each of which results in a different value for γ. The following table considers each case separately and provides the solution(s) for γ.

$a_{r,1}$	$a_{r,2}$	$a_{r,3}$	γ	Validness
0	0	0	$\mathbb{R}$	✓
0	0	1	-1	✗
0	1	0	0	✗
0	1	1	$-\infty$	✗
1	0	0	∞	✗
1	0	1	0	✗
1	1	0	-1	✗
1	1	1	$\mathbb{R}$	✓

In the table, $\mathbb{R}$ means that any real number is a solution. Remember that $\gamma \neq 0$ and $\gamma \neq$ -1 since the eigenvector $\mathbf{v}$ is assumed to be exactly 3-sparse. As a result, a solution to (7.70) exists only if $a_{r,1} = a_{r,2} = a_{r,3}$. Since this is necessary for all $r \in \{4, \cdots, N\}$, we get $\mathbf{a}_{R,1} = \mathbf{a}_{R,2} = \mathbf{a}_{R,3}$. This condition is the same as (7.35).

Conversely, assume that the condition (7.35) holds. Without loss of generality assume that $i = 1$, $j = 2$, and $k = 3$. Then we have $\mathbf{a}_{R,1} = \mathbf{a}_{R,2} = \mathbf{a}_{R,3}$, where $\mathbf{a}_{R,i}$ is the same as in (7.68). Define $s = \|\mathbf{a}_{R,1}\|_1 = |\mathcal{N}(i) \setminus \{j, k\}|$. Notice that $s > 0$, since $s = 0$ implies that the first three nodes are disconnected from the rest of the graph. Now consider the following eigenvalue equation:

$$\begin{bmatrix} a_{1,2}+a_{1,3} & -a_{1,2} & -a_{1,3} \\ -a_{1,2} & a_{1,2}+a_{2,3} & -a_{2,3} \\ -a_{1,3} & -a_{2,3} & a_{1,3}+a_{2,3} \end{bmatrix} \begin{bmatrix} 1 \\ \gamma \\ -(1+\gamma) \end{bmatrix} = (\lambda\text{-}s) \begin{bmatrix} 1 \\ \gamma \\ -(1+\gamma) \end{bmatrix}. \tag{7.71}$$

Notice that the matrix on the left-hand side is the Laplacian of the subgraph on the first three nodes. Since the graph is unweighted, this matrix can have $2^3 = 8$ different forms. By exhaustively considering each case, one can show that (7.71) can always be solved for λ and γ with $\gamma \neq 0$ and $\gamma \neq$ -1. However, values of both λ and γ depend on the matrix. Eigenvalues of a graph Laplacian are always non-negative, therefore $\lambda\text{-}s \geq 0$. As a result $\lambda \geq s > 0$.

Notice that $d_1 = s + a_{1,2} + a_{1,3}$, $d_2 = s + a_{1,2} + a_{2,3}$, and $d_3 = s + a_{1,3} + a_{2,3}$. Therefore, a pair of (λ, γ) that satisfies (7.71) also satisfies (7.69). Hence, using γ solved from (7.71), a 3-sparse vector $\mathbf{v}$ constructed as $v_1 = 1$, $v_2 = \gamma$, $v_3 = -(1+\gamma)$, and $v_i = 0$ for $i \geq 4$ is an eigenvector of the graph Laplacian $\mathbf{L}$. Furthermore, the corresponding eigenvalue λ (computed via (7.71)) is nonzero.

7.9.8 Proof of Theorem 7.10

Assume that the Laplacian of an undirected, unweighted and connected graph has a 3-sparse eigenvector. Then, due to Theorem 7.9, there exist nodes i, j, k with the condition in (7.35). Let $\mathcal{S} = \mathcal{N}(i) \setminus \{j, k\} = \mathcal{N}(j) \setminus \{i, k\} = \mathcal{N}(k) \setminus \{i, j\}$. The relations in-between the nodes i, j, k can have 4 different forms. This follows from the fact that there are 4 non-isomorphic simple graphs on 3 nodes (page 4 of [77]). These cases are illustrated Figure 7.5. In the following table, we consider all 4 cases

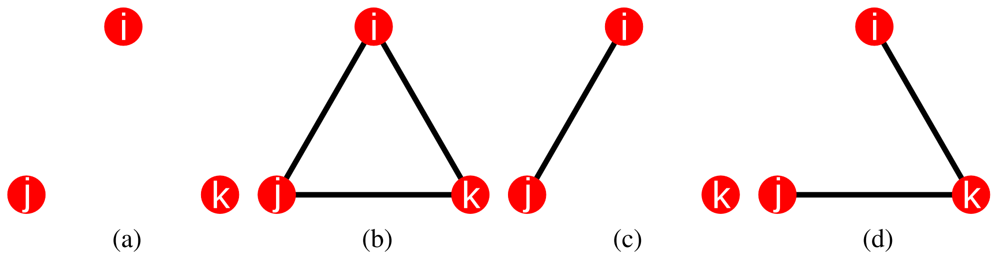


Figure 7.5: All four non-isomorphic graphs on 3 nodes.

separately and show that there exists a pair of nodes i, j with $\mathcal{N}(i)\setminus\{j\} = \mathcal{N}(j)\setminus\{i\}$.

Case	$\mathcal{N}(i)$	$\mathcal{N}(j)$	$\mathcal{N}(i) \setminus \{j\}$	$\mathcal{N}(j) \setminus \{i\}$
Figure 7.5a	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$
Figure 7.5b	$\mathcal{S} \cup \{j, k\}$	$\mathcal{S} \cup \{i, k\}$	$\mathcal{S} \cup \{k\}$	$\mathcal{S} \cup \{k\}$
Figure 7.5c	$\mathcal{S} \cup \{j\}$	$\mathcal{S} \cup \{i\}$	$\mathcal{S}$	$\mathcal{S}$
Figure 7.5d	$\mathcal{S} \cup \{k\}$	$\mathcal{S} \cup \{k\}$	$\mathcal{S} \cup \{k\}$	$\mathcal{S} \cup \{k\}$

As a result, due to Theorem 7.8, the graph Laplacian has a 2-sparse eigenvector *independent* of the relation between the nodes i, j, k.

Chapter 8

ENERGY COMPACTION FILTERS ON GRAPHS

8.1 Introduction

In the study of graph signals, polynomial filters play an important role. Their significance follows from their localization property: when implemented on a graph, a polynomial filter of order L requires a node to communicate only with its L-hop neighbors. Moreover, polynomial graph filters are analogous to finite impulse response (FIR) filters of classical signal processing. Elements of the graph Fourier basis can be amplified or suppressed according to the behavior of the filter. Thus, the design of such polynomials in the context of graphs is an important problem.

The spectral concentration problem in classical signal processing searches for the optimal FIR filter (of fixed order) that confines the largest amount of energy into a specific bandwidth. The problem was first formulated and analyzed by Slepian in his seminal works [162, 161]. The solution to the problem is known as the prolate sequence, and it provides the optimal (in the least squares sense) window for the filter design problem [200].

In this chapter, we consider the spectral concentration problem for polynomial graph filters. Given a filter order L and a bandwidth σ, we consider the optimal selection of the coefficients such that the energy confined in the band (of the graph) is maximized. This problem is analogous to the classical spectral concentration problem [162, 161, 200]. The difference lies in the definition of the spectrum: in the classical case the spectrum is defined with respect to the unit circle, whereas in the case of graphs the spectrum is an interval on the real line. In spite of their conceptual similarity, the analysis in the graph case differs from the classical one, and requires additional attention.

For the energy compaction problem on graphs, we take two approaches. In the first one, we assume that the spectrum of the graph is continuous. In this case the problem reduces to the polynomial concentration problem studied more generally by Slepian in [69]. We re-visit the problem, compare it with the classical case and present its asymptotic behavior in the case of narrow bandwidth. Although the continuous approach provides a theoretical and graph-free reference point, it is not applicable to graphs directly as graphs have discrete spectrum of finite size. In the second approach, we define the problem with respect to the spectrum of the graph. Thus, the optimal filter becomes specific to the underlying graph. We consider different examples of graphs, and compare the behavior of the maximum energy compaction as well as the optimal filter.

We would like to note that the studies in [195, 205] and Chapter 7 focus on the concentration and localization properties of graph signals. In particular, [195, 205] extend the classical time-frequency concentration problem to the case of graphs. Different from [195, 205] and Chapter 7, this chapter focuses on the energy concentration properties of polynomial graph filters. Thus, results here do not involve vertex domain properties.

In Section 8.2 we provide a quick overview of the classical spectral concentration problem. In Section 8.3 we define the energy compaction problem for the continuous case and study the behavior of the solution. In Section 8.4 we consider the discrete graph-dependent counter-part of the problem and investigate the effect of the spectrum of the graph on the optimal filter.

The content of this chapter is drawn from [170].

8.2 The Energy Compaction Problem

Let $H(e^{j\omega})$ denote the frequency response of a causal FIR filter of order L that is defined as follows:

$$H(e^{j\omega}) = \sum_{k=0}^{L} h_k \, e^{-j\omega k}, \tag{8.1}$$

where $h_k \in \mathcal{R}$ denote the coefficients of the filter. The problem of energy compaction (or, spectral concentration) searches for the filter whose energy is maximized in the specified passband. This can be described precisely with the following optimization problem:

$$\phi(\sigma) = \max_{h_k} \int_0^{\pi\sigma} \left|H(e^{j\omega})\right|^2 \frac{\mathrm{d}\omega}{\pi} \quad \text{s.t.} \quad \int_0^{\pi} \left|H(e^{j\omega})\right|^2 \frac{\mathrm{d}\omega}{\pi} = 1, \tag{8.2}$$

where $0 < \sigma < 1$ denotes the (normalized) bandwidth of the passband, and $\phi(\sigma)$ denotes the maximum amount of energy that can be confined in the band $[0 \;\; \sigma]$. As described clearly in Chapter 3.2.2 of [200], when the coefficients of an FIR filter are represented as a vector $\mathbf{h} = [h_0 \;\cdots\; h_L]^{\mathrm{T}}$, the problem in (8.2) can be reformulated as the following Rayleigh quotient:

$$\widehat{\mathbf{h}} = \arg\max_{\mathbf{h}} \quad \mathbf{h}^{\mathrm{H}} \, \mathbf{P} \, \mathbf{h} \quad \text{s.t.} \quad \|\mathbf{h}\|_2^2 = 1, \tag{8.3}$$

where the kernel matrix $\mathbf{P} \in \mathbb{R}^{(L+1)\times(L+1)}$ is given as follows:

$$(\mathbf{P})_{m,n} = \sigma \ \operatorname{sinc}\big(\sigma\,(m-n)\big), \qquad 1 \leq m,n \leq L{+}1, \tag{8.4}$$

where $\operatorname{sinc}(x) = \sin(\pi x)/(\pi x)$. Then, the optimal filter, $\widehat{\mathbf{h}}$, and its energy compaction, $\phi(\sigma)$, can be found as the dominant eigenvector-eigenvalue pair of the positive-definite and Toeplitz matrix $\mathbf{P}$. The solution, $\widehat{\mathbf{h}}$, is also known as the prolate sequence. Many other properties of the eigenvectors and the eigenvalues of the matrix $\mathbf{P}$ were studied by Slepian in [161].

8.3 Graph Independent Continuous Spectrum

Given a graph operator $\mathbf{A}$, a polynomial graph filter of order L (or FIR graph filter) is defined as follows:

$$H(\mathbf{A}) = \sum_{k=0}^{L} h_k\, \mathbf{A}^k. \tag{8.5}$$

Since $\mathbf{A}$ and $H(\mathbf{A})$ are simultaneously diagonalizable, the filter scales a graph Fourier component corresponding to an eigenvalue λ with $H(\lambda)$. Thus, the frequency response of a polynomial filter can be written as follows:

$$H(\lambda) = \sum_{k=0}^{L} h_k\, \lambda^k. \tag{8.6}$$

At the core of most practical applications lie low-pass filters, which can be described conceptually as follows:

$$\big|H(\lambda)\big| \approx \begin{cases} 1, & \text{if } \lambda \leq \sigma, \\ 0, & \text{if } \lambda > \sigma, \end{cases} \tag{8.7}$$

where $0 < \sigma < 1$, and σ denotes the cut-off frequency of the filter. Depending on the design criteria one can construct different filters to achieve the behavior in (8.7). Motivated by the results in [200, 162, 161], we consider here the energy compaction filter similar to the one in (8.2). More specifically we consider the following problem, which was first addressed in [69]:

$$\gamma(\sigma) = \max_{h_k} \int_0^{\sigma} |H(\lambda)|^2 \,\mathrm{d}\lambda \quad \text{s.t.} \quad \int_0^{1} |H(\lambda)|^2 \,\mathrm{d}\lambda = 1. \tag{8.8}$$

It should be noted that we treat $\lambda \in \mathbb{R}$ (spectrum of the graph) as a continuous parameter in (8.8), which is contrary to the fact that a spectrum of a graph is discrete and has at most N eigenvalues (N being the size of graph). More importantly, eigenvalues of a graph are not spaced uniformly, they may be concentrated (or, clustered) around some specific intervals. (See Figure 8.4 later.) Nevertheless, the problem in (8.8) has two theoretical advantages: 1) The formulation is graph-free. Therefore, it considers a unified approach to the filter design problem. 2) It provides a theoretical reference point and allows us to answer the following question: can we ignore the underlying graph structure and design filters universally? As we shall discuss in Secion 8.4, *the answer is no: the graph spectrum matters.*

In order to convert the problem (8.8) into matrix-vector equations we first define the following vector variables:

$$\boldsymbol{\lambda} = \begin{bmatrix} 1 \\ \lambda \\ \vdots \\ \lambda^L \end{bmatrix} \in \mathbb{R}^{L+1}, \qquad \mathbf{h} = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_L \end{bmatrix} \in \mathbb{C}^{L+1}. \tag{8.9}$$

Since $\boldsymbol{\lambda} \in \mathbb{R}^{L+1}$ we have

$$H(\lambda) = \boldsymbol{\lambda}^{\mathrm{H}}\, \mathbf{h}, \qquad |H(\lambda)|^2 = \mathbf{h}^{\mathrm{H}}\, \boldsymbol{\lambda}\, \boldsymbol{\lambda}^{\mathrm{H}}\, \mathbf{h}. \tag{8.10}$$

Then, the objective function (as well as the constraint) in (8.8) can be written as follows:

$$\int_0^\sigma |H(\lambda)|^2 \,\mathrm{d}\lambda = \int_0^\sigma \mathbf{h}^{\mathrm{H}}\, \boldsymbol{\lambda}\, \boldsymbol{\lambda}^{\mathrm{H}}\, \mathbf{h} \,\mathrm{d}\lambda = \mathbf{h}^{\mathrm{H}}\, \mathbf{Q}(\sigma)\, \mathbf{h}, \tag{8.11}$$

where the matrix $\mathbf{Q}(\sigma)$ consists of the following terms:

$$\Big(\mathbf{Q}(\sigma)\Big)_{m,n} = \int_0^\sigma \lambda^{m+n-2} \,\mathrm{d}\lambda \;=\; \frac{\sigma^{m+n-1}}{m+n-1}. \tag{8.12}$$

So, the matrix $\mathbf{Q}(\sigma)$ has the following form:

$$\mathbf{Q}(\sigma) = \begin{bmatrix} \sigma & \frac{\sigma^2}{2} & \cdots & \frac{\sigma^{L+1}}{L+1} \\ \frac{\sigma^2}{2} & \frac{\sigma^3}{3} & \cdots & \frac{\sigma^{L+2}}{L+2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\sigma^{L+1}}{L+1} & \frac{\sigma^{L+2}}{L+2} & \cdots & \frac{\sigma^{2L+1}}{2L+1} \end{bmatrix}. \tag{8.13}$$

Using (8.11), the problem in (8.8) can be written as follows:

$$\gamma(\sigma) = \max_{\mathbf{h}} \quad \mathbf{h}^{\mathrm{H}}\, \mathbf{Q}(\sigma)\, \mathbf{h} \qquad \text{s.t.} \qquad \mathbf{h}^{\mathrm{H}}\, \mathbf{Q}(1)\, \mathbf{h} = 1. \tag{8.14}$$

Thus, the optimal energy compaction problem on graphs can be formulated as a *generalized Rayleigh quotient* problem. Before elaborating on the solution of (8.14), we first present the following lemma:

Lemma 8.1. $\mathbf{Q}(\sigma)$ *is a symmetric matrix with the Hankel structure. Moreover, it satisfies the following ordering for* $0 < \sigma_1 < \sigma_2 \leq 1$*:*

$$\mathbf{0} \prec \mathbf{Q}(\sigma_1) \prec \mathbf{Q}(\sigma_2) \preceq \sigma_2\, \pi\, \mathbf{I}. \tag{8.15}$$

Proof. For a given arbitrary complex vector $\mathbf{h} \neq \mathbf{0}$, (8.11) gives the following:

$$\mathbf{h}^{\mathrm{H}} \Big(\mathbf{Q}(\sigma_2) - \mathbf{Q}(\sigma_1)\Big)\, \mathbf{h} = \int_0^{\sigma_2} |H(\lambda)|^2 \,\mathrm{d}\lambda - \int_0^{\sigma_1} |H(\lambda)|^2 \,\mathrm{d}\lambda = \int_{\sigma_1}^{\sigma_2} |H(\lambda)|^2 \,\mathrm{d}\lambda > 0,$$

where the strict positivity follows from the fact that the right-hand-side is an integral of a non-negative polynomial over an interval and the polynomial only has finite number of zeros. Thus, $\mathbf{Q}(\sigma_2) \succ \mathbf{Q}(\sigma_1)$ for any $\sigma_2 > \sigma_1$. In particular note that $\mathbf{Q}(0) = \mathbf{0}$, so $\mathbf{Q}(\sigma) \succ \mathbf{0}$ for any $\sigma > 0$.

For the last inequality in (8.15), we will use the Hilbert's inequality given in the following form [112]:

$$\sum_{i=0} \sum_{j=0} \frac{x_i\, x_j}{i + j + 1} \leq \pi \sum_{i=0} x_i^2. \tag{8.16}$$

Let $\mathbf{x}$ denote the eigenvector of $\mathbf{Q}(\sigma)$ corresponding to the largest eigenvalue. Since $\mathbf{Q}(\sigma)$ has positive values, we know that $\mathbf{x}$ has positive values as well (This is due to Perron-Frobenius theorem). Then,

$$\mathbf{x}^{\mathrm{H}}\, \mathbf{Q}(\sigma)\, \mathbf{x} = \sum_{i=1} \sum_{j=1} \frac{x_i\, x_j\, \sigma^{i+j-1}}{i + j - 1} = \sigma \sum_{i=0} \sum_{j=0} \frac{x_{i+1}\, \sigma^i \;\; x_{j+1}\, \sigma^j}{i + j + 1} \tag{8.17}$$

$$\leq \sigma\, \pi \sum_{i=0} \Big(x_{i+1}\, \sigma^i\Big)^2 \leq \sigma\, \pi \sum_{i=0} x_{i+1}^2 = \sigma\, \pi\, \|\mathbf{x}\|_2^2, \tag{8.18}$$

where we first use the Hilbert's inequality, then we use the assumption that $\sigma \leq 1$ in (8.18). This proves the last inequality in (8.15). □

Since $\mathbf{Q}(\sigma)$ does not have a null-space and is bounded, the problem in (8.14) is well-defined. Moreover, it can be converted into a standard Rayleigh quotient problem. For this purpose, consider the Cholesky decomposition of $\mathbf{Q}(1)$:

$$\mathbf{Q}(1) = \mathbf{C}\,\mathbf{C}^{\mathrm{H}}, \tag{8.19}$$

where we assume that $\mathbf{C}$ is a lower triangular matrix with strictly positive diagonal entries, hence $\mathbf{C}$ is unique and invertible. Then, the problem in (8.14) can be equivalently written as follows:

$$\begin{aligned}\gamma(\sigma) &= \max_{\mathbf{v}} \quad \mathbf{v}^{\mathrm{H}}\,\mathbf{C}^{-1}\,\mathbf{Q}(\sigma)\,\mathbf{C}^{-\mathrm{H}}\,\mathbf{v} \qquad \text{s.t.} \qquad \mathbf{v}^{\mathrm{H}}\,\mathbf{v} = 1 && (8.20)\\ &= \|\mathbf{C}^{-1}\;\mathbf{Q}(\sigma)\;\mathbf{C}^{-\mathrm{H}}\|_2. && (8.21)\end{aligned}$$

Furthermore, the optimal filter that achieves the maximum energy compaction can be found as follows:

$$\widehat{\mathbf{h}} = \mathbf{C}^{-\mathrm{H}}\;\mathbf{v}, \tag{8.22}$$

where $\mathbf{v}$ is the dominant eigenvector of the symmetric matrix in (8.21).

It should be noted that the matrix $\mathbf{Q}(1)$ corresponds to *a Hilbert matrix* of size L+1 [82], which has been used extensively in the study of polynomial approximations. A Hilbert matrix has many interesting properties and challenges, among which lies the condition number. A Hilbert matrix is positive definite for any size as shown by Lemma 8.1. However, the condition number grows like $(1+\sqrt{2})^{4n}/\sqrt{n}$ for the size n Hilbert matrix [15] making the matrix so ill-conditioned that MATLAB *fails* to compute the Cholesky decomposition in (8.19) for $L \geq 13$. Nevertheless, researchers have obtained closed form expressions for the matrices that are related to a Hilbert matrix. For example, the study in [37] shows that the inverse of the Cholesky factor in (8.19) has the following entries:

$$\left(\mathbf{C}^{-1}\right)_{m,n} = (\text{-}1)^{m+n}\,\sqrt{2m-1}\,\binom{m+n-2}{n-1}\binom{m-1}{n-1}, \quad m \geq n, \tag{8.23}$$

which allows the direct computation of the matrix in (8.21). It is important to note that entries in $\mathbf{C}^{-1}$ grow exponentially with its size L. Thus, direct computation of the matrix in (8.21) is still prone to the numerical problems for large values of L.

It is well-known that (8.14) can be converted into the following generalized eigenvalue problem with the use of Lagrange multiplier:

$$\mathbf{Q}(\sigma)\;\mathbf{h} = \gamma\;\mathbf{Q}(1)\;\mathbf{h}, \tag{8.24}$$

whose dominant eigenvalue-eigenvector pair provides the maximum amount of energy compaction and the corresponding filter that achieves it. Although the formulation in (8.24) is easier to implement in numerical environments, it still suffers from numerical precision even for moderate values of L.

8.3.1 The Optimal Filter and the Maximum Energy Compaction

Although closed form solution for the dominant eigenpair of (8.24) is not available, a numerical solution is possible to obtain for small values of L. In Figure 8.1 we present the maximum energy compaction, $\gamma(\sigma)$, as a function of the bandwidth σ for different values of L. For a fixed order L, notice that $\gamma(\sigma)$ is an increasing function of σ, that is, larger amount of energy can be confined in a larger bandwidth. Moreover, for a fixed bandwidth σ, the amount of energy compaction increases as the filter order L gets larger. This shows a trade-off between the locality of the graph filter and better (close to the ideal) low-pass characteristics.

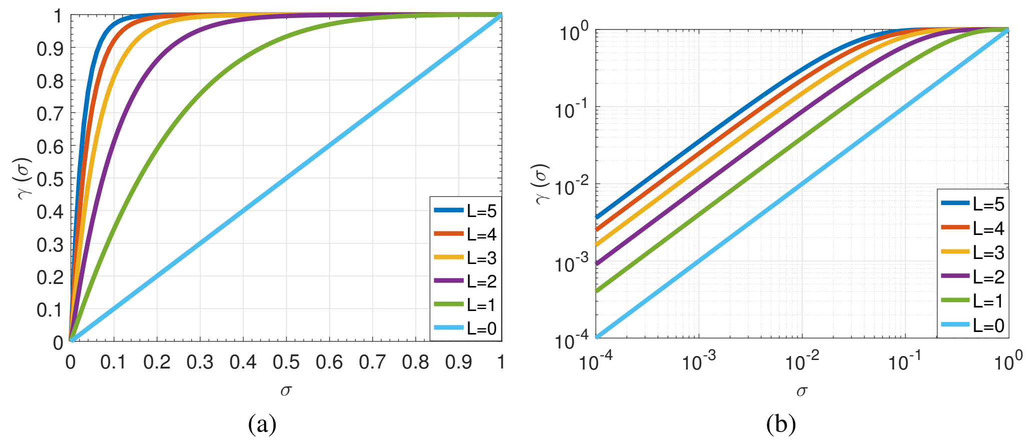


Figure 8.1: Dominant eigenvalue of (8.24) in (a) linear-scale (b) log-scale.

The optimal filters that achieve the maximum energy compaction are presented in Figure 8.2 in which the bandwidth is selected as $\sigma = 0.2$, and filters with different orders are considered. As seen in Figure 8.2b the filters have zeros in the interval $[0.2 \quad 1]$. In fact, numerical observations suggests that the optimal filter of order L for the bandwidth σ has exactly L zeros in the interval $(\sigma \quad 1]$. This is an expected result since the problem in (8.8) minimizes the energy confined in $(\sigma \quad 1]$. Thus, all the zeros are located in this interval.

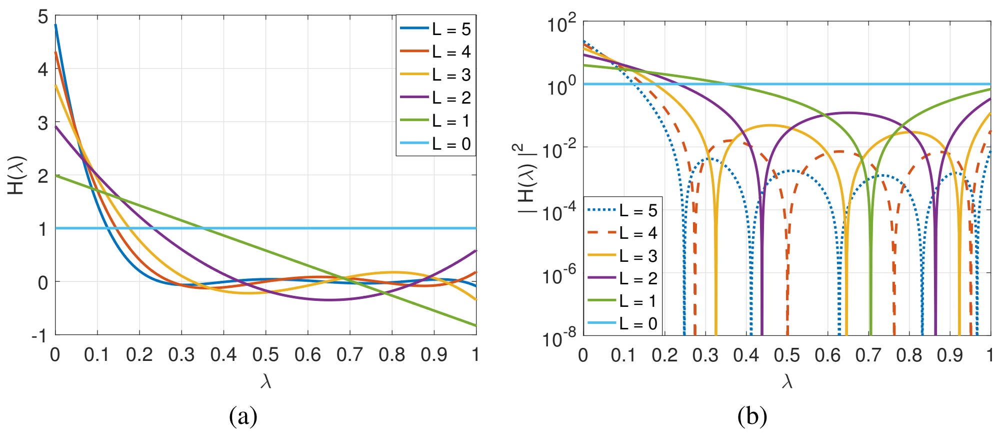


Figure 8.2: The optimal filter responses for $\sigma = 0.2$ and different values of L. Filter responses in (a) linear-scale, (b) log-scale.

8.3.2 Narrow Bandwidth Behavior

Although a closed form solution for the dominant eigenpair of (8.24) does not exist, for small values of σ, Figure 8.1b suggests that the amount of energy compaction depends *linearly* on the bandwidth. The following theorem shows that this is in fact the case:

Theorem 8.1. *For small σ, the maximum amount of energy concentration of an order L filter is approximated as follows:*

$$\gamma(\sigma) \approx \sigma \ (L+1)^2. \tag{8.25}$$

Moreover, the coefficients of the optimal filter can be approximated as

$$h_k = (-1)^k \ \frac{(L+k+1)!}{(L-k)! \ \ k! \ \ (k+1)!}, \qquad 0 \le k \le L. \tag{8.26}$$

Proof. Assume that σ is small. Then, the entires of the matrix $\mathbf{Q}(\sigma)$ vanish (except the top-left one) since they have powers of σ. Then, we have

$$\mathbf{Q}(\sigma) \approx \begin{bmatrix} \sigma & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} = \sigma \ \mathbf{e}_1 \ \mathbf{e}_1^{\mathrm{H}}, \tag{8.27}$$

where $\mathbf{e}_1 \in \mathbb{R}^{L+1}$ and $\mathbf{e}_1 = [1 \ 0 \ \cdots \ 0]^{\mathrm{H}}$. Using (8.27), we approximate (8.21) as follows:

$$\gamma(\sigma) \approx \sigma \ \left\| \mathbf{C}^{-1} \ \mathbf{e}_1 \ \mathbf{e}_1^{\mathrm{H}} \ \mathbf{C}^{-\mathrm{H}} \right\|_2 \tag{8.28}$$

Since $\mathbf{C}^{-1}\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}$ is a rank-1 and positive semi-definite matrix, we have

$$\left\|\mathbf{C}^{-1}\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}\right\|_2 = \mathrm{tr}\left(\mathbf{C}^{-1}\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}\right) \tag{8.29}$$

$$= \mathrm{tr}\left(\mathbf{e}_1^{\mathrm{H}}\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}\,\mathbf{C}^{-1}\right) = \mathrm{tr}\left(\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{Q}(1)^{-1}\right) \tag{8.30}$$

$$= \mathrm{tr}\left(\mathbf{e}_1^{\mathrm{H}}\,\mathbf{Q}(1)^{-1}\,\mathbf{e}_1\right) = \mathbf{e}_1^{\mathrm{H}}\,\mathbf{Q}(1)^{-1}\,\mathbf{e}_1. \tag{8.31}$$

Remember that $\mathbf{Q}(1)$ is the Hilbert matrix of size $L{+}1$, and the inverse of a size N Hilbert matrix can be written explicitly as follows [37]:

$$\left(\mathbf{Q}(1)^{-1}\right)_{i,j} = (-1)^{i+j}\,(i+j-1)\binom{N+i-1}{N-j}\binom{N+j-1}{N-i}\binom{i+j-2}{i-1}^2. \tag{8.32}$$

Then, by selecting $i = j = 1$ and $N = L+1$ in (8.32), we get the following:

$$\mathbf{e}_1^{\mathrm{H}}\,\mathbf{Q}(1)^{-1}\,\mathbf{e}_1 = \left(\mathbf{Q}(1)^{-1}\right)_{1,1} = (L+1)^2 \tag{8.33}$$

which proves (8.25) due to (8.28).

In (8.22) the optimal filter is give as $\widehat{\mathbf{h}} = \mathbf{C}^{-\mathrm{H}}\,\mathbf{v}$ where $\mathbf{v}$ is the dominant eigenvector of $\mathbf{C}^{-1}\,\mathbf{Q}(\sigma)\,\mathbf{C}^{-\mathrm{H}}$. According to the approximation in (8.27), we have that

$$\mathbf{C}^{-1}\,\mathbf{Q}(\sigma)\,\mathbf{C}^{-\mathrm{H}} \approx \sigma\,\mathbf{C}^{-1}\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}. \tag{8.34}$$

Since the right-hand-side of the above equation is in the form of a single outer product, the vector that forms the outer product is the dominant (and the only) eigenvector. Therefore,

$$\mathbf{v} \approx \frac{1}{L{+}1}\,\mathbf{C}^{-1}\,\mathbf{e}_1, \tag{8.35}$$

where the scale factor $1/(L{+}1)$ is selected such that $\|\mathbf{v}\|_2 = 1$:

$$\|\mathbf{v}\|_2^2 = \frac{1}{(L{+}1)^2}\;\mathrm{tr}\left(\mathbf{C}^{-1}\,\mathbf{e}_1\,\mathbf{e}_1^{\mathrm{H}}\,\mathbf{C}^{-\mathrm{H}}\right) = 1, \tag{8.36}$$

which follows from (8.29) and (8.33). Then we have:

$$\widehat{\mathbf{h}} = \mathbf{C}^{-\mathrm{H}}\,\mathbf{v} \approx \frac{1}{L{+}1}\,\mathbf{C}^{-\mathrm{H}}\,\mathbf{C}^{-1}\,\mathbf{e}_1 = \frac{1}{L{+}1}\,\mathbf{Q}(1)^{-1}\,\mathbf{e}_1. \tag{8.37}$$

Notice that $\mathbf{Q}(1)^{-1}\,\mathbf{e}_1$ is the first column of the inverse of the Hilbert matrix of size $L+1$ whose elements can be found explicitly by using (8.32) with $i = k{+}1$, $j = 1$, and $N = L+1$:

$$h_k = \frac{1}{L{+}1}\,(-1)^{k+2}\,(k+1)\binom{L+k+1}{L}\binom{L+1}{L-k}\binom{k}{k}^2 \tag{8.38}$$

$$= (-1)^k\,\frac{k+1}{L+1}\,\frac{(L+k+1)!}{L!\,(k+1)!}\,\frac{(L+1)!}{(L-k)!\,(k+1)!}, \tag{8.39}$$

which is equivalent to (8.26). □

The asymptotic behavior of the energy compaction of polynomial filters resembles that of the classical FIR filters. Eq. (64) of [161] approximated the solution of the energy compaction problem in (8.2) as $\phi(\sigma) \approx \sigma\,(L+1)$ for small values of σ. Although both $\phi(\sigma)$ and $\gamma(\sigma)$ depend linearly on the bandwidth, $\phi(\sigma)$ depends on the order linearly, whereas $\gamma(\sigma)$ has a quadratic dependence resulting in $\gamma(\sigma) \geq \phi(\sigma)$ in the case of narrow bandwidth. In fact, as observed in Figure 8.3a, $\gamma(\sigma) \geq \phi(\sigma)$ for all values of σ. Thus, polynomial filters (graph filters) can confine more energy.

It is also interesting to see that the approximation of the optimum filter given in (8.26) has *integer valued coefficients* with alternating signs. In the case of $L = 3$, which is illustrated in Figure 8.3b, these coefficients can be found as follows:

$$h_0 = 4, \qquad h_1 = -30, \qquad h_2 = 60, \qquad h_3 = -35. \tag{8.40}$$

As Figure 8.3b shows, (8.26) approximates the optimal filter very well, and the approximation gets better as σ decreases.

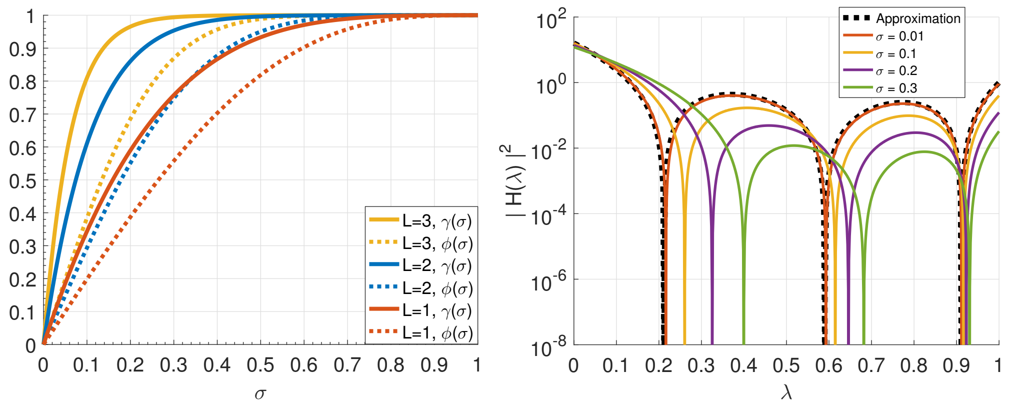


Figure 8.3: (a) Comparison of the maximum energy compaction achieved in classical and graph filters. (b) Magnitude response of the optimal filter for $L = 3$ for different values of σ. Response of the approximation (8.40) is also shown.

8.4 Graph Dependent Discrete Spectrum

In the previous section analysis of the energy compaction is based on the continuous spectrum. Although such an analysis is theoretically important, its practical importance is limited since graphs have finite number of eigenvalues. In this section, we take the eigenvalues of the graph into account and formulate the discrete counterpart of the problem in (8.8) as follows:

$$\rho(K) = \max_{h_k} \frac{1}{N}\sum_{i=1}^{K} |H(\lambda_i)|^2 \qquad \text{s.t.} \qquad \frac{1}{N}\sum_{i=1}^{N} |H(\lambda_i)|^2 = 1, \tag{8.41}$$

where $K < N$ determines the pass band "width" of the filter.

Following the formulation in Section 8.3, (8.41) can be reformulated as the following generalized Rayleigh quotient problem:

$$\max_{\mathbf{h}} \qquad \mathbf{h}^{\mathrm{H}}\, \mathbf{S}(K)\, \mathbf{h} \qquad \text{s.t.} \qquad \mathbf{h}^{\mathrm{H}}\, \mathbf{S}(N)\, \mathbf{h} = 1, \tag{8.42}$$

where

$$\big(\mathbf{S}(K)\big)_{m,n} = \frac{1}{N} \sum_{i=1}^{K} \lambda_i^{m+n-2}. \tag{8.43}$$

Then, the optimum filter and the maximum amount of energy compaction can be found as the dominant eigenpair of the following generalized eigenvalue problem:

$$\mathbf{S}(K)\, \mathbf{h} = \rho\, \mathbf{S}(N)\, \mathbf{h}. \tag{8.44}$$

Although the problems in (8.14) and (8.42) have the same form, their characteristics differ from each other in two respects. Firstly, K in (8.42) is a discrete parameter as opposed to σ in (8.14) being a continuous. Nevertheless, they can be conceptually related as $\sigma = K/N$, which denotes the fraction of eigenvalues in the baseband of the graph spectrum. Secondly, and more importantly, the spectrum of a graph has a finite number of possibly repeated eigenvalues. Thus, the matrix $\mathbf{S}(N)$ in (8.42) may have a null-space unlike $\mathbf{Q}(1)$. More precisely, we have the following lemma:

Lemma 8.2. *Let $\bar{N}$ denote the number of distinct eigenvalues of the graph operator. If $L < \bar{N}$, then $\mathbf{S}(N) \succ \mathbf{0}$; if otherwise, $\mathbf{S}(N)$ has a null-space, hence positive semi-definite.*

When the matrix $\mathbf{S}(N)$ has a null-space it can be shown that the problem in (8.42) does not have a unique maximizer. Thus, the optimal energy compaction filter is not unique. This means that the order of the polynomial filter is larger than what is necessary, and a lower order filter can obtain the same amount of energy compaction. In most applications low orders are preferred in order to have filters that are localized on the graph. So, the condition in Lemma 8.2 is almost always satisfied in practice yielding a positive definite $\mathbf{S}(N)$. Moreover, when the order of the filter satisfies

$L \geq \bar{N}$-1 *any* frequency response can be realized with a polynomial [153]. Thus, the maximum energy compaction becomes $\rho(K) = 1$ for all values of K.

Since $\mathbf{S}(N)$ depends on the eigenvalues of the underlying graph operator, a closed form expression for it does not exist in general. Nevertheless, it is possible to obtain closed forms in some specific cases. For example, if the graph is an undirected cycle of size N and its graph Laplacian is used as the operator, Lemma 9.2 of this thesis reveals that $\mathbf{S}(N)$ has the following closed form

$$\big(\mathbf{S}(N)\big)_{m,n} = \binom{2m+2n-4}{m+n-2} \tag{8.45}$$

as long as the order of the filter satisfies $L < N/2$.

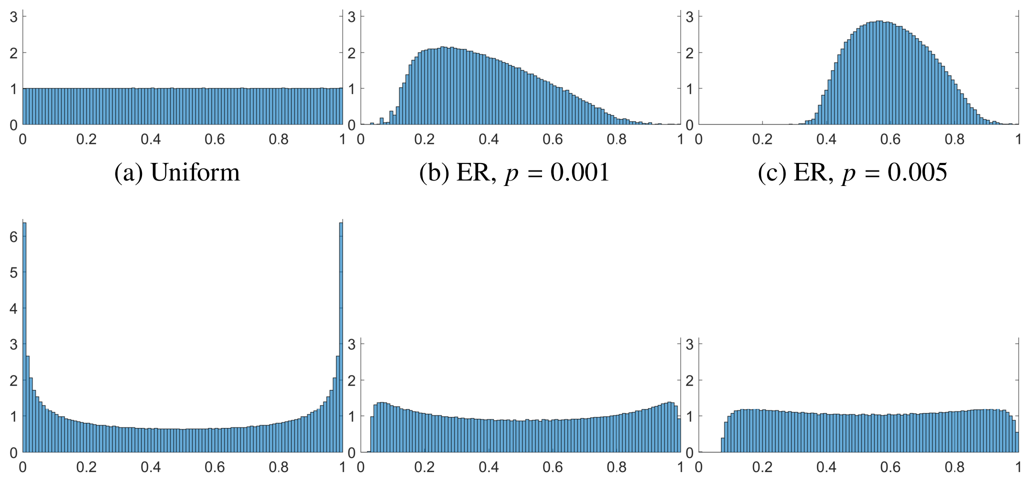
It is also important to note that $\mathbf{Q}(\sigma)$ and $\mathbf{S}(K)$ are asymptotically identical when the underlying eigenvalues are uniformly separated. That is, if $\lambda_i = i/N$ for $1 \leq i \leq N$, then

$$\lim_{N\to\infty} \mathbf{S}(\sigma N) = \mathbf{Q}(\sigma). \tag{8.46}$$

Thus, the problems in (8.14) and (8.42) are also asymptotically equivalent when the eigenvalues are separated uniformly. However, the spectrum of a graph is almost never distributed uniformly [159]. So, the solution to the energy compaction problem in (8.42) depends heavily on the underlying graph operator as we shall demonstrate next.

Figure 8.4 shows the histogram of the eigenvalues of the Laplacian of different examples of graphs including Erdős-Rényi (ER), random regular (RR) and the undirected cycle graph. We also consider the case of uniform eigenvalue separation (which does not correspond to a graph) as a reference. The size of the graphs is set to be $N = 10^4$, and the eigenvalues are scaled such that $0 \leq \lambda_i \leq 1$ since the scaling does not affect the generality of the results.

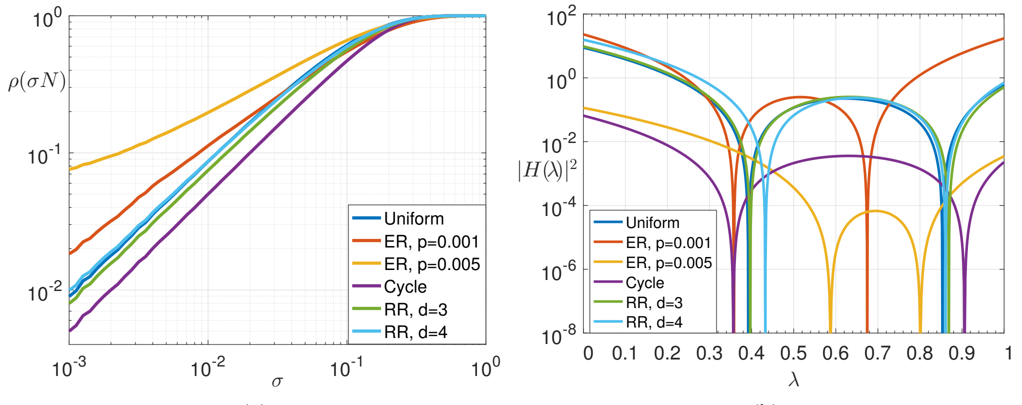
Figure 8.5 visualizes the numerical solution of (8.42) for the graphs considered above for filter order $L = 2$. In Figure 8.5a we consider the maximum amount of energy compaction with respect to $\sigma = K/N$. In Figure 8.5b we show the response of the optimum filters for $\sigma = 0.1$. As seen clearly from the figures, both the maximum energy compaction and the optimal filter are affected by the distribution of the eigenvalues. Among all considered examples, the compaction filter for the ER graph with $p = 0.005$ has the most "concentrated" spectrum, and Figure 8.5a shows that the optimum filter for the ER graph can confine more energy in a band compared to the other graphs. Similarly, Figure 8.5b shows that the zeros of the


(a) Uniform (b) ER, $p = 0.001$ (c) ER, $p = 0.005$

(d) Undirected cycle (e) RR, $d = 3$ (f) RR, $d = 4$

Figure 8.4: Histogram of the eigenvalues of (a) the uniform case, (b) Erdős-Rényi (ER) graph with $p = 0.001$, (c) ER graph with $p = 0.005$, (d) undirected cycle graph, (e) random regular (RR) graph with degree $d = 3$, and (f) RR graph with degree $d = 4$.

optimum polynomial are located where the eigenvalues are denser. On the other hand, the undirected cycle graph has the most "spread-out" spectrum with two different peaks. Figure 8.5a shows that the optimal filter on the undirected cycle graph has the least amount of energy confinement. Correspondingly, zeros of the optimum polynomial are also spread-out from each other in order to accommodate the spread-out in the spectrum.



(a) (b)

Figure 8.5: (a) Comparison of energy compaction on different graphs, (b) the optimal filters on different graphs for $\sigma = 0.1$.

8.5 Concluding Remarks

In this chapter we investigated the spectral concentration problem for polynomial graph filters and considered two approaches. In the first one, we assumed that the graph spectrum is continuous, in which case it reduced to the polynomial concentration problem studied by Slepian. We re-visited the problem and compared its solution with the classical spectral concentration problem. In the second approach, we took the discrete graph spectrum into consideration and formulated the problem accordingly. We showed that the maximum amount of energy compaction and the optimum filter depend on the spectrum of the underlying graph.

Chapter 9

TIME ESTIMATION FOR HEAT DIFFUSION ON GRAPHS

9.1 Introduction

In this chapter we will consider signals that diffuse over the graph through time. Specifically, we will examine the heat diffusion process, which has been shown to be applicable for various problems [38, 111]. In particular, we will assume that there is a point heat source that starts to diffuse at a known location but at an unknown time. We sample the heat distribution over the graph and consider the estimation of the starting time of the diffusion process when there is additive noise.

Section 9.2 will provide the mathematical formulation of the problem. In Section 9.3 we will derive the Cramér-Rao lower bound for the time estimation problem. We will also analyze the relation between the bound and the underlying graph structure. In particular we will show that the estimation problem is more difficult on graphs with higher connectivity. In Section 9.4 we will derive closed form expressions of the lower bound for the case of cycle graphs, complete graphs and star graphs. In Section 9.5 we will consider the Maximum-Likelihood (ML) estimator. We will numerically verify that the ML estimator is unbiased and achieves the Cramér-Rao lower bound for various values of the time parameter.

We would like to note that the study in [189] considers the same heat diffusion model in a different setting where the problem is to estimate the graph itself from the observed data.

The content of this chapter is drawn from [186].

9.2 Problem Statement

In this study we consider graph signals that are evolving through time. We will assume that the evolution process of a signal is governed by the following differential equation:

$$\frac{\partial \, \mathbf{x}(t)}{\partial \, t} = -\mathbf{L} \, \mathbf{x}(t), \qquad \text{s.t.} \qquad \mathbf{x}(0) = \mathbf{x}_0, \tag{9.1}$$

where $\mathbf{L}$ is the Laplacian of the graph on which the signal $\mathbf{x}(t)$ resides. The model in (9.1) is known to be the *heat equation*, and it describes how an initial heat distribution diffuses over a geometry [39]. The solution of (9.1) is given as [39]

$$\mathbf{x}(t) = \exp(\text{-}\mathbf{L}\,t)\ \mathbf{x}_0, \qquad t \geq 0, \tag{9.2}$$

where $\exp(\cdot)$ denotes the matrix exponential, and the signal $\mathbf{x}(t)$ gives the distribution of the heat over the graph $\mathbf{L}$ after time t when the initial distribution is $\mathbf{x}_0$.

Starting with the diffusion model in (9.2) we consider the following inverse problem: assume that there is a unit point source located at node n, that is, the input distribution is $\mathbf{e}_n$. This point source starts to diffuse at time $\tilde{t}$ over a known graph with the Laplacian $\mathbf{L}$. We observe the heat signal over the graph at time $T \geq \tilde{t}$ with an additive i.i.d. Gaussian noise with variance σ^2. Thus, we can write the observed signal as

$$\mathbf{y} = \exp\left(\text{-}\mathbf{L}\,(T - \tilde{t})\right)\ \mathbf{e}_n + \mathbf{w}, \tag{9.3}$$

where $\mathbf{w}$ is the i.i.d. Gaussian noise. Given the observation vector $\mathbf{y}$, our task is to estimate the starting time $\tilde{t}$ and the location n of the point source. Notice that the observation vector can be written as

$$\mathbf{y} = \exp(\text{-}\mathbf{L}\,t)\ \mathbf{e}_n + \mathbf{w}, \tag{9.4}$$

where t denotes the *relative* time with respect to the sampling time T. Since the observation time T is known, it suffices to find t. The actual starting time can be found easily as $\tilde{t} = T - t$. Hence, our main problem is to estimate t and n from the vector $\mathbf{y}$ given in (9.4). Note that t is a (non-negative) continuous parameter, whereas n is an integer ranging from 1 to N.

In the rest of the chapter we will consider a simpler case where the location of the point source, n, is assumed to be known and study the problem of estimation of the time parameter t.

9.3 The Cramér-Rao Lower Bound

Given an observation vector $\mathbf{y}$, let $\widehat{t}(\mathbf{y})$ be an *unbiased* estimator of the parameter t, that is, $E[\ \widehat{t}(\mathbf{y})\ ] = t$. The variance of an unbiased estimator is lower bounded as

$$\text{var}\left(\ \widehat{t}(\mathbf{y})\ \right) \geq C_{n,t}, \tag{9.5}$$

where $C_{n,t}$ is the *Cramér-Rao lower bound* (CRLB).

Due to the data model in (9.4), the observation vector $\mathbf{y}$ has the following multivariate Gaussian distribution:

$$\mathbf{y} \sim \mathcal{N}\left(\exp(\text{-}\mathbf{L}\,t)\ \mathbf{e}_n,\ \sigma^2\mathbf{I}\right). \tag{9.6}$$

Then, the CRLB is given as [201]

$$C_{n,t} = \sigma^2 \,/\, \mathcal{I}_{n,t}, \tag{9.7}$$

where

$$\mathcal{I}_{n,t} = \left\| \frac{\mathrm{d}d}{\mathrm{d}t} \; \exp(-\mathbf{L}\,t)\; \mathbf{e}_n \right\|^2 = \left\| \exp(-\mathbf{L}\,t)\; \mathbf{L}\; \mathbf{e}_n \right\|^2 = \mathbf{e}_n^{\mathrm{H}} \; \exp(-2\,\mathbf{L}\,t)\; \mathbf{L}^2\; \mathbf{e}_n. \tag{9.8}$$

From now on we will refer to $\mathcal{I}_{n,t}$ as the *heat information* of the node n at time t. Notice that $\mathcal{I}_{n,t}$ is intrinsic to the graph structure and does not depend on the noise level σ.

Using the eigenvalue decomposition of **L**, the heat information can be equivalently written as

$$\mathcal{I}_{n,t} = \mathbf{e}_n^{\mathrm{H}} \left(\sum_{k=1}^{N} e^{-2\lambda_k t} \lambda_k^2 \, \mathbf{v}_k \mathbf{v}_k^{\mathrm{H}} \right) \mathbf{e}_n = \sum_{k=1}^{N} \; e^{-2\lambda_k t} \; \lambda_k^2 \; |V_{n,k}|^2, \tag{9.9}$$

where $V_{n,k} = \mathbf{e}_n^{\mathrm{H}} \, \mathbf{v}_k$ denotes the $(n, k)^{th}$ entry of the eigenvector matrix **V**.

One immediate observation regarding the heat information is that it converges to zero for large values of t. That is,

$$\lim_{t \to \infty} \; \mathcal{I}_{n,t} = 0. \tag{9.10}$$

This asymptotic behavior is the same for all the nodes no matter what the underlying graph is. The behavior in (9.10) is consistent with the intuition behind the heat dynamics in (9.1): large values of t mean that the initial source is already diffused all over the graph, hence it is not possible to estimate its location and time even if no noise is present.

Apart from its asymptotic behavior the heat information is tightly connected to the underlying graph structure. As a motivating example consider the star graph and a modified star graph in Figure 9.1a and 9.1b, respectively.

The heat information of the center node (labeled as node 1) in a star graph is as follows (from Corollary 9.1):

$$\mathcal{I}_{1,t} = N\,(N-1)\; e^{-2\,N\,t}, \tag{9.11}$$

where N is the total number of nodes in the graph. When the graph in Figure 9.1a is modified into the one in Figure 9.1b by moving only one edge, the heat information of the center node becomes

$$\mathcal{I}_{1,t} = \alpha_1\, e^{-2\,s_1\,t} + \alpha_2\, e^{-2\,s_2\,t} + \alpha_3\, e^{-2\,s_3\,t}, \tag{9.12}$$

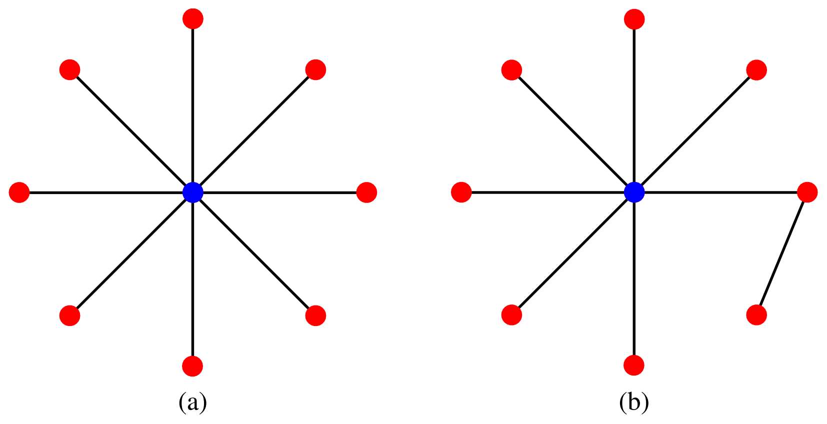


Figure 9.1: (a) Star graph on $N = 9$ nodes, (b) modified star graph.

for some constant α_i's, which possibly depend on N. Here s_i's are the roots of the following polynomial $p(s) = s^3 - (N+2)\, s^2 + (3N-2)\, s - N$. These roots can be well approximated as follows:

$$s_1 \approx N - 1, \qquad s_2 \approx \frac{3+\sqrt{5}}{2}, \qquad s_3 \approx \frac{3-\sqrt{5}}{2}. \tag{9.13}$$

The mere purpose of this example is to show the tight connection between the graph structure and the heat information of the nodes. Motivated with this example in the following sub-sections we will analyze the relation between the heat information and the underlying graph structure.

9.3.1 Average Heat Information over the Graph

In order to understand the general characteristics of the heat information in (9.8), we shall first consider the *average* heat information over all the nodes. Hence, we define the following:

$$\mathcal{I}_t := \frac{1}{N} \sum_{n=1}^{N} \mathcal{I}_{n,t}. \tag{9.14}$$

When we substitute (9.9) into the definition in (9.14) we obtain the following:

$$\mathcal{I}_t = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{N} e^{-2\lambda_k t}\, \lambda_k^2\, |V_{n,k}|^2 = \frac{1}{N} \sum_{k=1}^{N} e^{-2\lambda_k t}\, \lambda_k^2, \tag{9.15}$$

where we use the fact $\sum_{n=1}^{N} |V_{n,k}|^2 = \|\mathbf{v}_k\|^2 = 1$. Notice that the average heat information in (9.15) depends only on the spectrum, λ_k's, of the graph, but not the eigenvectors.

When the nodes of a graph have similar neighborhood structures $\mathcal{I}_t$ is expected to reflect the average behavior of the heat information on individual nodes. Note that nodes with peculiar structures (e.g. center node of a star graph) may behave differently than $\mathcal{I}_t$. Nevertheless, $\mathcal{I}_t$ is useful to understand the characteristics of the heat diffusion over a given graph. In particular, an upper bound on $\mathcal{I}_t$ is provided in the following theorem:

Theorem 9.1 (Effect of the Connectivity)**.** *Let λ_2 be the smallest non-zero eigenvalue of the Laplacian of a simple and connected graph. Then, the average heat information is bounded as*

$$\mathcal{I}_t \ \leq \ e^{-2\lambda_2 t}\, \mathcal{I}_0, \tag{9.16}$$

where $\mathcal{I}_0$ is the average heat information at time $t = 0$.

Proof. Observe the following:

$$\mathcal{I}_t = \frac{1}{N}\sum_{k=2}^{N} e^{-2\lambda_k t}\lambda_k^2 \leq \frac{1}{N}\sum_{k=2}^{N} e^{-2\lambda_2 t}\lambda_k^2 = e^{-2\lambda_2 t}\frac{1}{N}\sum_{k=2}^{N}\lambda_k^2, \tag{9.17}$$

where we use the fact that $\lambda_1 = 0$ and $\lambda_2 \leq \lambda_k$ for $k \geq 2$. Also notice that $\mathcal{I}_0$ is given as $\mathcal{I}_0 = 1/N\ \sum_{k=1}^{N}\lambda_k^2$. □

We would like to note that the average heat information at time $t = 0$ is related to the graph Laplacian as follow:

$$\mathcal{I}_0 = \frac{1}{N}\sum_{k=1}^{N}\lambda_k^2 = \frac{1}{N}tr(\mathbf{L}^2) = \frac{1}{N}\|\mathbf{L}\|_F^2. \tag{9.18}$$

The second smallest eigenvalue, λ_2, of the graph Laplacian is known as the *algebraic connectivity* of the graph [63]. Roughly speaking graphs with larger λ_2 tend to be more "connected" than the others: a complete graph has $\lambda_2 = N$; a star graph has $\lambda_2 = 1$; a cycle graph has $\lambda_2 = 2 - 2\cos(2\pi/N) \approx 4\pi^2/N^2$ (for large values of N), and $\lambda_2 = 0$ for graphs with disconnected components. Many more results on λ_2 and connectivity of a graph can be found in [47].

From the connectivity perspective the inequality in (9.16) states the following: a graph with higher connectivity is expected to have smaller average heat information at any given t. This statement agrees with the dynamics of the heat equation in

(9.1). If a graph has higher connectivity a point source is expected to diffuse easily on the graph, hence, the estimation of its location and time becomes a more difficult problem.

Even though the bound in Theorem 9.1 is tight for some graphs (see Section 9.4.2), it is *not* tight in general. Nevertheless, it is useful to provide some qualitative understanding of the difficulty of the problem in hand.

9.3.2 Power Series Representation

The expression in (9.9) describes the heat information in terms of the spectral characterization of the graph Laplacian. In the following we will describe the heat information in terms of algebraic properties of the graph. For this purpose we start with the Taylor series expansion of e^t at $t = 0$, that is,

$$\exp(t) = \sum_{k=0}^{\infty} \frac{t^k}{k!}. \tag{9.19}$$

Using the expansion (9.19) in (9.8), we obtain the following:

$$\mathcal{I}_{n,t} = \mathbf{e}_n^{\mathrm{H}} \left(\sum_{k=0}^{\infty} \frac{(\text{-}2\mathbf{L}\,t)^k}{k} \mathbf{L}^2 \right) \mathbf{e}_n = \sum_{k=0}^{\infty} \frac{(\text{-}2t)^k}{k}\, d_{n,\,k+2}, \tag{9.20}$$

where we define $d_{n,k}$ as the n^{th} diagonal term of the k^{th} power of $\mathbf{L}$, that is,

$$d_{n,k} := \mathbf{e}_n^{\mathrm{H}}\, \mathbf{L}^k\, \mathbf{e}_n = \left(\mathbf{L}^k\right)_{n,n}. \tag{9.21}$$

It is important to notice that $d_{n,k}$'s are local parameters of the node n in the sense that $d_{n,k}$ depends only on the connectivity of the k-hop neighborhood of the node n. In particular $d_{n,1} = d_n$ is the degree of the node n. Furthermore we have the following:

$$d_{n,2} = d_n^2 + d_n, \qquad d_{n,3} = d_n^3 + 2d_n^2 + w_{n,2} - c_{n,3}. \tag{9.22}$$

where $w_{n,2}$ and $c_{n,3}$ are the number of walks of length 2 and the number of *closed* walks of length 3, starting at the node n, respectively.

Given a graph, $d_{n,k}$'s are simple to compute numerically. However, closed form expression of $d_{n,k}$ in terms of the neighborhood structure is not available for an arbitrary graph. Nevertheless, closed form expression of $d_{n,k}$ can be derived for some specific cases. Our first result is as follows:

Lemma 9.1. *Let* $\mathbf{L}$ *be the graph Laplacian of an arbitrary simple graph on N nodes. Assume that the* n^{th} *node is connected to every other node, that is,* $d_n = N\text{-}1$. *Then,*

$$d_{n,k} = \left(\mathbf{L}^k\right)_{n,n} = (N\text{-}1)\, N^{k\text{-}1}. \tag{9.23}$$

Proof. Without loss of any generality assume that $n = 1$. Given an arbitrary graph with a fully connected node, we can write its adjacency matrix as follows:

$$\mathbf{A} = \begin{bmatrix} 0 & \mathbf{1}^{\mathrm{T}} \\ \mathbf{1} & \bar{\mathbf{A}} \end{bmatrix}, \tag{9.24}$$

where $\bar{\mathbf{A}} \in \mathbb{R}^{(N\text{-}1)\times(N\text{-}1)}$ represents the adjacency matrix of the induced graph on the nodes $2, \cdots, N$. Then, the Laplacian of this graph can be written as

$$\mathbf{L} = \begin{bmatrix} N\text{-}1 & \text{-}\mathbf{1}^{\mathrm{T}} \\ \text{-}\mathbf{1} & \bar{\mathbf{L}} + \mathbf{I} \end{bmatrix}, \tag{9.25}$$

where $\bar{\mathbf{L}}$ denotes the Laplacian of the induced graph on the nodes $2, \cdots, N$, and $\mathbf{1}$ is the all-1 vector of size N-1. We postulate that the k^{th} power of $\mathbf{L}$ has the following form:

$$\mathbf{L}^k = \begin{bmatrix} a_k & \text{-}b_k\, \mathbf{1}^{\mathrm{T}} \\ \text{-}b_k\, \mathbf{1} & c_k\, \mathbf{1}\mathbf{1}^{\mathrm{T}} + (\bar{\mathbf{L}} + \mathbf{I})^k \end{bmatrix}. \tag{9.26}$$

Notice that this holds true for $k = 1$ with $a_1 = N\text{-}1$, $b_1 = 1$ and $c_1 = 0$. We will now assume that (9.26) is true for an arbitrary k-1 and validate it for k. Using the fact that $\mathbf{L}^k = \mathbf{L}^{k\text{-}1}\, \mathbf{L}$ we obtain the following:

$$\mathbf{L}^k = \begin{bmatrix} a_{k\text{-}1} & \text{-}b_{k\text{-}1}\, \mathbf{1}^{\mathrm{T}} \\ \text{-}b_{k\text{-}1}\, \mathbf{1} & c_{k\text{-}1}\, \mathbf{1}\mathbf{1}^{\mathrm{T}} + (\bar{\mathbf{L}} + \mathbf{I})^{k\text{-}1} \end{bmatrix} \begin{bmatrix} N\text{-}1 & \text{-}\, \mathbf{1}^{\mathrm{T}} \\ \text{-}\, \mathbf{1} & \bar{\mathbf{L}} + \mathbf{I} \end{bmatrix} \tag{9.27}$$

$$= \begin{bmatrix} (a_{k\text{-}1}+b_{k\text{-}1})(N\text{-}1) & \text{-}(a_{k\text{-}1}+b_{k\text{-}1})\, \mathbf{1}^{\mathrm{T}} \\ \text{-}(a_{k\text{-}1}+b_{k\text{-}1})\, \mathbf{1} & (b_{k\text{-}1} + c_{k\text{-}1})\, \mathbf{1}\mathbf{1}^{\mathrm{T}} + (\bar{\mathbf{L}} + \mathbf{I})^k \end{bmatrix}, \tag{9.28}$$

where we use the fact $\mathbf{1}^{\mathrm{T}}\, \bar{\mathbf{L}} = \mathbf{0}$. Notice that the form in (9.28) is the same as the form in (9.26), which proves (by recursion) that (9.26) is in fact correct. The result in (9.28) further reveals that the coefficients in (9.26) can be found recursively as follows:

$$a_k = (N\text{-}1)(a_{k\text{-}1} + b_{k\text{-}1}), \quad b_k = a_{k\text{-}1} + b_{k\text{-}1}, \quad c_k = b_{k\text{-}1} + c_{k\text{-}1},$$

which in particular shows that $a_k = (N\text{-}1)\, b_k$. Therefore,

$$a_k = (N\text{-}1)\left(a_{k\text{-}1} + \frac{a_{k\text{-}1}}{N\text{-}1}\right) = N\, a_{k\text{-}1}, \qquad b_k = N\, b_{k\text{-}1}. \tag{9.29}$$

Remember that $a_1 = N\text{-}1$, $b_1 = 1$ and $c_1 = 0$. Hence,

$$a_k = (N\text{-}1)\, N^{k\text{-}1}, \qquad b_k = N^{k\text{-}1}. \tag{9.30}$$

Furthermore notice that

$$c_k = c_1 + \sum_{l=1}^{k-1} b_l = \sum_{l=0}^{k-2} N^l = \frac{N^{k-1} - 1}{N - 1}. \tag{9.31}$$

As a result,

$$d_{1,k} = \mathbf{e}_1^{\mathrm{H}} \, \mathbf{L}^k \, \mathbf{e}_1 = a_k = (N\text{-}1)\, N^{k-1}, \tag{9.32}$$

which proves the result in (9.23). □

An immediate result of Lemma 9.1 is as follows:

Corollary 9.1. *Let* **L** *be the graph Laplacian of an arbitrary simple graph on N nodes. Assume that the* n^{th} *node has* $d_n = N$-1*. Then, the heat information for the* n^{th} *node is*

$$\mathcal{I}_{n,t} = N(N\text{-}1)\; e^{-2Nt}. \tag{9.33}$$

Proof. Using (9.23) in (9.20) we get

$$\mathcal{I}_{n,t} = \sum_{k=0}^{\infty} \frac{(\text{-}2t)^k}{k} (N\text{-}1)\, N^{k+1} = (N\text{-}1)\, N \sum_{k=0}^{\infty} \frac{(\text{-}2tN)^k}{k} = N(N\text{-}1)\; e^{-2Nt}. \tag{9.34}$$

□

Corollary 9.1 states that the heat information of a fully connected node does *not* depend on the remaining graph structure (except for the size of the graph). This result will be useful to analyze some examples of graphs (see Section 9.4.2 and 9.4.3).

Closed form expression of $d_{n,k}$ is obtained for cycle graphs as follows:

Lemma 9.2. *For the cycle graph of size N, we have*

$$d_{n,k} = \left(\mathbf{L}^k\right)_{n,n} = \binom{2k}{k} = \frac{(2k)!}{k!\; k!} \qquad \textit{for} \qquad k < N/2. \tag{9.35}$$

Proof. Assume $k < N/2$. For the adjacency matrix of a cycle graph we have $\mathbf{A} = \mathbf{C} + \mathbf{C}^{-1}$, where $\mathbf{C}$ is the circular shift matrix. Therefore $\mathbf{A}^2 = 2\mathbf{I} + \mathbf{C}^2 + \mathbf{C}^{-2}$. Hence,

$$\mathbf{e}_n^{\mathrm{H}} \, \mathbf{A}^{2k} \, \mathbf{e}_n = \sum_{k_1+k_2+k_3=k} \binom{k}{k_1, k_2, k_3} 2^{k_1} \, \mathbf{e}_n^{\mathrm{H}} \, \mathbf{C}^{2(k_2\text{-}k_3)} \, \mathbf{e}_n, \tag{9.36}$$

$$= \sum_{\substack{k_1+k_2+k_3=k \\ k_2=k_3}} \binom{k}{k_1, k_2, k_3} 2^{k_1}, \tag{9.37}$$

where the last equality follows from the fact that $\mathbf{e}_n^{\mathrm{H}}\mathbf{C}^{2(k_2\text{-}k_3)}\mathbf{e}_n = \delta(k_2 - k_3)$ for $k_2, k_3 < N/2$.

Also note that the Laplacian of a cycle graph can be written as $\mathbf{L} = 2\mathbf{I} - \mathbf{C} - \mathbf{C}^{\text{-}1}$. Hence,

$$\mathbf{e}_n^{\mathrm{H}}\, \mathbf{L}^k\, \mathbf{e}_n = \sum_{k_1+k_2+k_3=k} \binom{k}{k_1, k_2, k_3} 2^{k_1}\, \mathbf{e}_n^{\mathrm{H}}\, (\text{-}\mathbf{C})^{(k_2\text{-}k_3)}\, \mathbf{e}_n \tag{9.38}$$

$$= \sum_{\substack{k_1+k_2+k_3=k \\ k_2=k_3}} \binom{k}{k_1, k_2, k_3} 2^{k_1}. \tag{9.39}$$

As a result we get $\mathbf{e}_n^{\mathrm{H}}\, \mathbf{A}^{2k}\, \mathbf{e}_n = \mathbf{e}_n^{\mathrm{H}}\, \mathbf{L}^k\, \mathbf{e}_n = d_{n,k}$ for $k < N/2$. Furthermore notice that $\mathbf{e}_n^{\mathrm{H}}\mathbf{A}^{2k}\mathbf{e}_n$ is the number of closed walks of length $2k$ on the cycle graph of size $N > 2k$. In order to have a closed walk of length $2k$ we need k left-shifts and k right-shifts in total. However they can be in any order. Therefore, the total number of such walks is given as $(2k)!/(k!\,k!)$. This proves (9.35). $\square$

We want to note that Lemma 9.2 is proved for $k < N/2$, however we have numerically verified that (9.35) holds true for $k < N$ as well.

9.3.3 Short-Time Approximations

Power series representation of the heat information is especially useful to understand the behavior of $\mathcal{I}_{n,t}$ for small values of t, in which case $\mathcal{I}_{n,t}$ is expected to depend only on the local properties of the n^{th} node. In the following we will show that this is in fact the case.

Using only the first three terms of (9.20), the heat information for small values of t can be approximated as follows:

$$\mathcal{I}_{n,t} \approx d_{n,2} - 2\,t\,d_{n,3} + 2\,t^2\,d_{n,4}, \tag{9.40}$$

which depends only on 4-hop neighborhood of the node n. Moreover, in the limit, the heat information has the following:

$$\lim_{t\to0^+} \mathcal{I}_{n,t} = \mathbf{e}_n^{\mathrm{H}}\, \mathbf{L}^2\, \mathbf{e}_n = d_{n,2} = d_n^2 + d_n, \tag{9.41}$$

which depends only on the degree (1-hop neighbors) of the node n.

9.4 Closed Form Expressions of the Heat Information for Some Graphs

In this section we will present closed form expressions of the heat information for cycle, complete and star graphs. See Figure 9.2 for visual representations of these graphs.

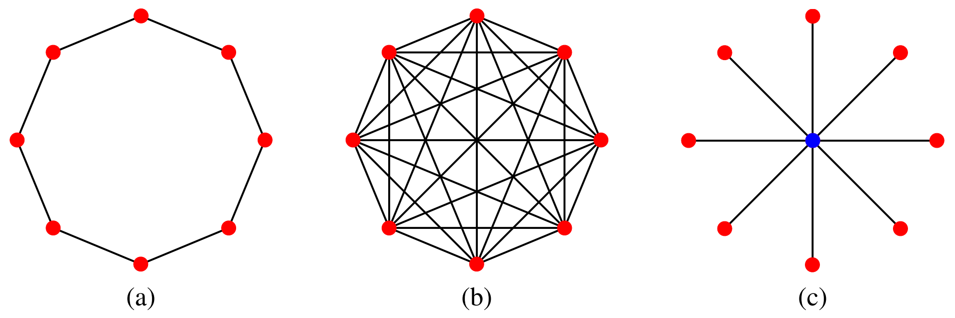


Figure 9.2: Visualizations of a (a) cycle graph, (b) complete graph, and (c) star graph.

9.4.1 Cycle Graphs

Using the expression in (9.9), the heat information of the n^{th} node on the cycle graph can be written as

$$\mathcal{I}_{n,t} = \sum_{k=1}^{N} e^{-2\lambda_k t}\, \lambda_k^2\, |V_{n,k}|^2 = \frac{1}{N}\sum_{k=1}^{N} e^{-2\lambda_k t}\, \lambda_k^2, \tag{9.42}$$

where we use the fact that a cycle graph is a circulant graph, hence the graph Laplacian can be diagonalized by the unitary DFT matrix of size N. Therefore $|V_{n,k}|^2 = 1/N$ for all (n,k).

It is important to note that $\mathcal{I}_{n,t}$ does *not* depend on the node n. As a result, the heat information is equivalent to the average: $\mathcal{I}_{n,t} = \mathcal{I}_t$. In fact this equivalence holds true for any circulant graph.

It is known that the Laplacian spectrum of a cycle graph is $\lambda_k = 2\,(1\text{-}\cos(2\pi(k\text{-}1)/N))$ for $1 \le k \le N$ [29]. Therefore,

$$\mathcal{I}_{n,t} = \frac{4\,e^{-4t}}{N}\sum_{k=0}^{N-1} e^{4t\,\cos(2\pi k/N)}\,\big(1\text{-}\cos(2\pi k/N)\big)^2. \tag{9.43}$$

We can consider the expression in (9.43) as an approximation in the form of a Riemann sum. Therefore, for sufficiently large values of N, the heat information can be well approximated with the following integral:

$$\mathcal{I}_{n,t} \approx 4\,e^{-4t}\int_0^1 e^{4t\,\cos(2\pi u)}\,\big(1-\cos(2\pi u)\big)^2\,\mathrm{d}u, \tag{9.44}$$

whose closed form solution is given by

$$\mathcal{I}_{n,t} \approx e^{-4t}\Big(6\,I_0(4t) - 8\,I_1(4t) + 2\,I_2(4t)\Big), \tag{9.45}$$

where $I_\nu(t)$ denotes the modified Bessel function of the first kind of order ν of t.

Using the asymptotic approximation of modified Bessel functions of the first kind (see Eq. (9.7.1) of [1]), the heat information in (9.45) can be further approximated as

$$\mathcal{I}_{n,t} \approx \frac{3}{32\sqrt{2\pi}} \; t^{-5/2} \tag{9.46}$$

for large values of t. On the other hand, for small values of t, (9.40) gives an approximation of the heat information as follows:

$$\mathcal{I}_{n,t} \approx d_{n,2} - 2\,t\,d_{n,3} + 2\,t^2\,d_{n,4} = 6 - 40\,t + 140\,t^2, \tag{9.47}$$

where we use the fact that $d_{n,k} = \binom{2k}{k}$ from Lemma 9.2.

The heat information in (9.43) together with the approximations in (9.46) and (9.47) are provided in Figure 9.3 for various different values of N. It is clear that the short-time approximation is valid regardless of N, whereas the long-time approximation is valid for sufficiently large values of N.

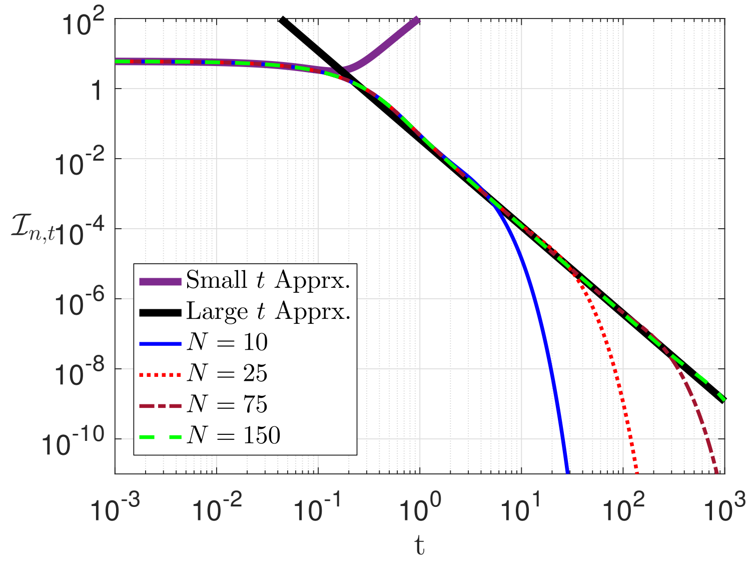


Figure 9.3: The heat information on cycle graphs for various values of N together with the short-time and the long-time approximations.

9.4.2 Complete Graphs

Similar to cycle graphs, a complete graph is a circulant graph as well. Hence, the heat information can be written as in (9.42). For the complete graph of size N it is well known that $\lambda_1 = 0$ and $\lambda_k = N$ for $k \geq 2$. As a result, the heat information can be found as

$$\mathcal{I}_{n,t} = \frac{1}{N} \sum_{k=2}^{N} e^{-2Nt} N^2 = N(N\text{-}1)\, e^{-2Nt}. \tag{9.48}$$

We would like to note that this result is consistent with Corollary 9.1 since any node of a complete graph is connected to every other node. Hence, (9.48) and (9.33) are the same. Furthermore, for the average heat information we have $\mathcal{I}_t/\mathcal{I}_0 = e^{-2Nt}$ for complete graphs. Since $\lambda_2 = N$, the bound in Theorem 9.1 is achieved with equality for complete graphs.

9.4.3 Star Graphs

For the star graph of size N, let's assume that the center node is labeled as 1. Since the center node is connected to every other node of the graph, we can apply Corollary 9.1 and obtain that $\mathcal{I}_{1,t} = N(N\text{-}1)\, e^{-2Nt}$. For the remaining nodes of the star graph we have the following:

$$d_{n,k} = \frac{N^{k\text{-}1} + N - 2}{N - 1} \qquad \text{for} \qquad n \geq 2. \tag{9.49}$$

The result in (9.49) can be seen from the proof of Lemma 9.1 by letting $\bar{\mathbf{L}} = \mathbf{0}$ and observing that $d_{n,k} = c_k{+}1$ where c_k is given by (9.31). The use of (9.49) in (9.20) results in the following:

$$\mathcal{I}_{n,t} = \sum_{k=0}^{\infty} \frac{(\text{-}2t)^k}{k!} \, \frac{N^{k+1}{+}N\text{-}2}{N\text{-}1} = \frac{N\, e^{-2Nt} + (N\text{-}2)\, e^{-2t}}{N\text{-}1}. \tag{9.50}$$

These examples of graphs justify the interpretation of Theorem 9.1. The cycle graph (having the least connectivity among three) has the largest heat information: $\mathcal{I}_{n,t}$ decays with the 2.5^{th} power of t. The complete graph has the highest connectivity, and $\mathcal{I}_{n,t}$ decays exponentially with both t and the size of the graph. The star graph lies in the middle: the center node behaves the same as the nodes of a complete graph (see Corollary 9.1). For the remaining nodes $\mathcal{I}_{n,t}$ decays exponentially with t, but it is not affected by the size of the graph (asymptotically).

9.5 The Maximum-Likelihood Estimator

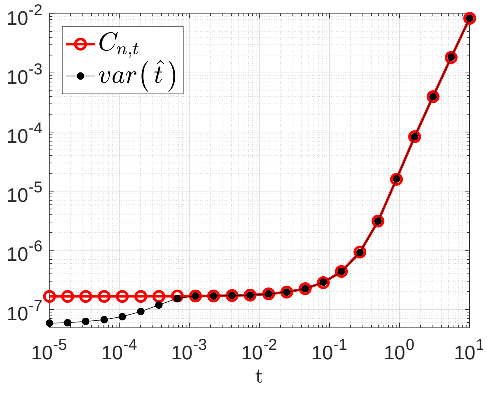
The observation vector $\mathbf{y}$ has the distribution given in (9.6). As a result, the Maximum-Likelihood (ML) estimator of the parameter t can be derived as follows:

$$\hat{t}_{\mathrm{ML}}(\mathbf{y}) = \arg\min_{t \geq 0} \left\| \mathbf{y} - \exp(\text{-}\mathbf{L}\, t)\, \mathbf{e}_n \right\|^2, \tag{9.51}$$

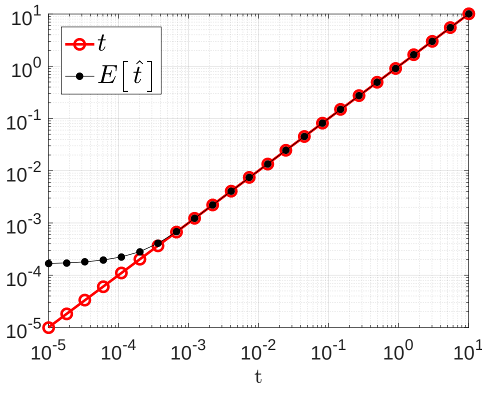
where the location of the source, n, is assumed to be known.

In general, ML estimators are not guaranteed to be unbiased, and they may not achieve the Cramér-Rao lower bound. In the following, we will numerically verify that the ML estimator in (9.51) is in fact unbiased and achieves the Cramér-Rao lower bound for various different values of the parameter t.

In the following experiment we consider the cycle graph of size $N = 150$ and take the observation noise level to be $\sigma = 10^{-3}$. We select a node and fix it for the whole experiment. We generate data according to the model in (9.4) for various different values of t and estimate the time parameter according to (9.51) using a dense grid search. We repeat this procedure 10^5 times. The variance and the mean of the simulated estimation results together with the theoretical bounds are given in Figure 9.4a and Figure 9.4b, respectively.



(a)



(b)

Figure 9.4: Performance of the ML estimator in (9.51): (a) variance, (b) mean.

As discussed in [67], the quantity $C_{n,t} \,/\, t^2$ provides intuitive interpretations for the sensitivity and the normalized mean-squared-error of the estimation problem. In the case of $C_{n,t} \geq t^2$, we do not expect the ML estimator to achieve the CRLB. According to (9.41) we have $C_{n,t} \approx \sigma^2(d_n^2 + d_n)^{-1}$ for small values of t. Hence, the ML estimator is not expected to meet the CRLB when

$$t \leq \sigma \ (d_n^2 + d_n)^{-1/2} \leq \sigma / d_n, \tag{9.52}$$

assuming that σ is small enough (high SNR regime).

For the case of cycle graphs we have $d_n = 2$. Hence, (9.52) becomes $t \leq 5 \cdot 10^{-4}$ in our experimental setting. This cut-off point can be seen clearly from Figure 9.4, before which the ML estimator is *biased*, and its variance is *lower* than the CRLB.

9.6 Concluding Remarks

In this chapter we assumed that there is a point source at a known vertex of a given graph. This source starts to diffuse according to the heat equation at an unknown time. We studied the estimation of the starting time from a noisy measurement of the signal over the graph. In particular we derived the Cramér-Rao lower bound for the problem. We showed that for graphs with higher connectivity the problem has a larger lower bound making the estimation problem more difficult. We simulated the performance of the ML estimator and showed that it is unbiased and achieves the CRLB for a wide range of parameters. We also characterized the case where the variance of the ML estimator deviates from the CRLB.

In future work, we will consider the case of unknown source location. We will develop convex optimization based techniques for simultaneous estimation of the time and the location of a point source. Furthermore, we will study the case when there are multiple sources. We will analyze the change in the CRLB in the presence of multiple sources, and develop techniques for the estimation of source parameters.

Chapter 10

CONCLUSIONS AND FUTURE DIRECTIONS

In this thesis, we studied various aspects of graph signal processing. In Chapter 2, we presented a randomized and asynchronous version of the graph shift. In Chapter 3 we studied an asynchronous implementation of graph filters. In Chapter 4, we extended the randomized model to include time-varying input signal, and studied its behavior from a linear system theory viewpoint. Then, in Chapter 5 we showed that switching system viewpoint provides a useful tool set for the analysis of random asynchronous model and randomized Kaczmarz and Gauss-Seidel algorithms. In Chapter 6, we studied an extension of multirate signal processing and filter banks to the case of graph. In Chapter 7, we studied discrete uncertainty principles on graphs and presented the existence of sparse eigenvectors. In Chapter 8, we studied energy compaction filters on graphs for optimal polynomial filter design, and in Chapter 9 we studied estimation of the starting time of a diffusion over a graph.

In addition to the results presented in this thesis we believe that there are many interesting research problems for future studies, which will be described next:

Further Aspects of Randomized Asynchronous Models: In the work presented in Chapters 2 and 3 nodes are assumed to communicate with each other asynchronously *but reliably*. However, in a more realistic scenario randomly broadcasted messages may not be received by some of the recipients due to unreliable communication between the nodes, in which case the theoretical analysis becomes inconclusive. Nevertheless, initial numerical findings showed that convergence can be achieved even in the case of unreliable communications. Therefore, it is possible to incorporate unreliable communications into the asynchronous model and provide rigorous results on convergence properties. Since unreliable communications can be interpreted as time-varying networks, it is possible to consider such results further for asynchronous communications on time-varying networks.

The results presented in Chapters 2, 3, 4, and 5 have considered the convergence (or stability) of randomized asynchronous updates on *a given system* with *a given set of update probabilities.* Since the necessary and sufficient conditions for the convergence (and stability) depend on the system matrix (graph) and the index (node) update probabilities, it is very natural to consider the *design* of such randomized systems. This is an interesting and challenging task due to the conditions being nonlinear (even non-convex) in terms of the underlying parameters. In this regard, it is possible to study the optimal probabilities as well as the design of the network itself for different objectives, including, *but not limited to*, maximizing the rate of convergence, minimizing the effect of the input noise, and obtaining robustness to ambiguities in system parameters. Each such problem requires a careful attention and is expected to open up even more interesting research questions.

Nonlinear Updates: Recurrent Neural Networks: In addition to the applications in graph signal processing, the randomized asynchronous update model studied in this thesis can also be interpreted as *single layer linear recurrent neural networks* [84], where the underlying graph determines the type of network (e.g., edges with weights ± 1 correspond to a Hopfield network). Therefore, it is possible to consider extensions of these results to the case of nonlinear updates, where the nonlinearity corresponds to *the activation function* of neural networks. Considering the elevated interest in neural networks in recent years, we believe that analysis of recurrent neural networks from the viewpoint of system theory carries a significant potential for further developments. It is also possible to study extensions of the randomized asynchronous model to the emerging field of geometric deep learning (graphical neural networks), which aims to extend the neural networks to non-Euclidean domains such as graphs [28].

Arbitrarily Sparse Eigenvectors and Sparse Graph Fourier Basis: Chapter 7 of this thesis considered the existence of sparse eigenvectors from the uncertainty viewpoint, and the main focus was to find the most sparse eigenvector, namely 2-sparse case. Although the chapter have provided the necessary and sufficient condition for the existence of 2-sparse and 3-sparse eigenvectors, necessary conditions for the existence of an arbitrary K-sparse eigenvector remains unknown. More interestingly, 2-sparse and 3-sparse eigenvectors are necessarily localized on the graph, whereas an arbitrary K-sparse eigenvector need *not* be localized. Therefore, it would be interesting to reveal the intricate relation between the sparsity and the locality properties of the eigenvectors of graphs. In addition, earlier results on this problem considered the existence of a single sparse eigenvector while the Fourier transform on a graph depends on all of the eigenvectors. So, it would also be interesting to study sparse graph Fourier basis in detail.

BIBLIOGRAPHY

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions*. Dover Publications, 1965.

[2] A. Agaskar and Y. M. Lu. “A Spectral Graph Uncertainty Principle”. In: *IEEE Trans. on Inf. Theory* 59.7 (July 2013), pp. 4338–4356.

[3] A. Agaskar, C. Wang, and Y. M. Lu. “Randomized Kaczmarz algorithms: Exact MSE analysis and optimal sampling probabilities”. In: *IEEE Global Conf. Signal and Inf. Process. (GlobalSIP)*. 2014, pp. 389–393.

[4] A. Anis, A. Gadde, and A. Ortega. “Towards a sampling theorem for signals on arbitrary graphs”. In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. May 2014, pp. 3864–3868.

[5] Bengt Aspvall and John R. Gilbert. “Graph Coloring Using Eigenvalue Decomposition”. In: *SIAM Journal on Algebraic Discrete Methods* 5.4 (1984), pp. 526–538.

[6] Konstantin Avrachenkov, Vivek S. Borkar, and Krishnakant Saboo. “Distributed and Asynchronous Methods for Semi-supervised Learning”. In: *Int. Workshop on Alg. and Models for the Web-Graph*. 2016, pp. 34–46.

[7] Konstantin Avrachenkov, Vivek S. Borkar, and Krishnakant Saboo. *Parallel and Distributed Approaches for Graph Based Semi-supervised Learning*. Tech. rep. 8767. Inria, Aug. 2015.

[8] Konstantin Avrachenkov et al. “Generalized Optimization Framework for Graph-based Semi-supervised Learning”. In: *SIAM International Conf. Data Mining*. 2012, pp. 966–974.

[9] Haim Avron, Alex Druinsky, and Anshul Gupta. “Revisiting Asynchronous Linear Solvers: Provable Convergence Rate Through Randomization”. In: *J. ACM* 62.6 (Dec. 2015), 51:1–51:27. ISSN: 0004-5411.

[10] Baruch Awerbuch. “Complexity of Network Synchronization”. In: *J. ACM* 32.4 (Oct. 1985), pp. 804–823.

[11] Anirban Banerjee and Jurgen Jost. “On the spectrum of the normalized graph Laplacian”. In: *Lin. Alg. and its Appl.* 428 (2008), pp. 3015–3022.

[12] C. Barnes and A. Fam. “Minimum norm recursive digital filters that are free of overflow limit cycles”. In: *IEEE Trans. on Circuits and Systems* 24.10 (Oct. 1977), pp. 569–574.

[13] S. Basagni. “Distributed clustering for ad hoc networks”. In: *Parallel Architectures, Algorithms, and Networks*. 1999, pp. 310–315.

[14] Gérard M. Baudet. “Asynchronous Iterative Methods for Multiprocessors”. In: *J. ACM* 25.2 (Apr. 1978), pp. 226–244.

[15] Bernhard Beckermann. “The condition number of real Vandermonde, Krylov and positive definite Hankel matrices”. In: *Numerische Mathematik* 85.4 (June 2000), pp. 553–577.

[16] F. K. Bell and P. Rowlinson. “On the multiplicities of graph eigenvalues”. In: *Bull. of the London Math. Soc.* 35.3 (May 2003), pp. 401–408.

[17] J. J. Benedetto and P. J. Koprowski. “Graph theoretic uncertainty principles”. In: *Int. Conf. Sampling Theory and Appl. (SampTA)*. May 2015, pp. 357–361.

[18] M. W. Berry et al. “An algorithm to compute a sparse basis of the null space”. In: *Numerische Mathematik* 47.4 (1985), pp. 483–504.

[19] Dimitri P. Bertsekas. “Distributed asynchronous computation of fixed points”. In: *Mathematical Programming* 27.1 (Sept. 1983), pp. 107–120.

[20] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[21] Amit Bhaya and Eugenius Kaszkurewicz. “On discrete-time diagonal and D-stability”. In: *Linear Algebra and its Applications* 187 (1993), pp. 87–104.

[22] P. Bianchi and J. Jakubowicz. “Convergence of a Multi-Agent Projected Stochastic Gradient Algorithm for Non-Convex Optimization”. In: *IEEE Trans. Autom. Control* 58.2 (Feb. 2013), pp. 391–405.

[23] R. Bitmead and B. Anderson. “Lyapunov techniques for the exponential stability of linear difference equations with random coefficients”. In: *IEEE Transactions on Automatic Control* 25.4 (Aug. 1980), pp. 782–787.

[24] V. D. Blondel et al. “Convergence in Multiagent Coordination, Consensus, and Flocking”. In: *Proceedings of the 44th IEEE Conference on Decision and Control*. Dec. 2005, pp. 2996–3000.

[25] Eric Bonabeau, Guy Theraulaz, and Marco Dorigo. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[26] John Adrian Bondy and U.S.R. Murty. *Graph Theory With Applications*. Elsevier Science Ltd/North-Holland, 1976.

[27] S. Boyd et al. “Randomized gossip algorithms”. In: *IEEE Trans. on Inf. Theory* 52.6 (June 2006), pp. 2508–2530.

[28] M. M. Bronstein et al. “Geometric Deep Learning: Going beyond Euclidean data”. In: *IEEE Signal Processing Magazine* 34.4 (July 2017), pp. 18–42.

[29] Andries E. Brouwer and Willem H. Haemers. *Spectra of Graphs*. Springer, 2011.

[30] Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. Cengage Learning, 2010.

[31] Andrzej Cegielski. *Iterative Methods for Fixed Point Problems in Hilbert Spaces*. Springer, 2012.

[32] D. Chazan and W. Miranker. "Chaotic relaxation". In: *Linear Algebra and its Applications* 2 (Apr. 1969), pp. 199–222.

[33] Chi-Tsong Chen. *Linear System Theory and Design*. 3rd. Oxford University Press, Inc., 1998.

[34] Siheng Chen, Aliaksei Sandryhaila, and Jelena Kovacevic. "Sampling theory for graph signals". In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. Apr. 2015, pp. 3392–3396.

[35] Siheng Chen et al. "Discrete Signal Processing on Graphs: Sampling Theory". In: *IEEE Trans. Signal Process.* 63.24 (Dec. 2015), pp. 6510–6523.

[36] Xuemei Chen and Alexander M. Powell. "Almost Sure Convergence of the Kaczmarz Algorithm with Random Measurements." In: *J. Fourier Anal Appl* 18 (2012), pp. 1195–1214.

[37] Man-Duen Choi. "Tricks or Treats with the Hilbert Matrix". In: *The American Mathematical Monthly* 90.5 (May 1983), pp. 301–312.

[38] Fan Chung. "The heat kernel as the pagerank of a graph". In: *Proc. National Academy of Sciences* 104.50 (2007), pp. 19735–19740.

[39] Fan R. K. Chung. *Spectral Graph Theory*. AMS, 1997.

[40] Thomas F. Coleman and Alex Pothen. "The Null Space Problem I. Complexity". In: *SIAM J. on Algebraic Discrete Methods* 7.4 (1986), pp. 527–537.

[41] Thomas F. Coleman and Alex Pothen. "The Null Space Problem II. Algorithms". In: *SIAM J. on Algebraic Discrete Methods* 8.4 (1987), pp. 544–563.

[42] Oswaldo Luiz Valle Costa, Ricardo Paulino Marques, and Marcelo Dutra Fragoso. *Discrete-Time Markov Jump Linear Systems*. Springer, 2005.

[43] Ronald E. Crochiere and Lawrence R. Rabiner. *Multirate Digital Signal Processing*. Prentice Hall, 1987.

[44] Dragos Cvetkovic, Peter Rowlinson, and Slobodan Simic. *Eigenspaces of Graphs*. Cambridge University Press, 2008.

[45] Dragos M. Cvetkovic, Michael Doob, and Horst Sachs. *Spectra of Graphs: Theory and Application (Pure & Applied Mathematics)*. Academic Press, 1980.

[46] L. Dai, M. Soltanalian, and K. Pelckmans. "On the Randomized Kaczmarz Algorithm". In: *IEEE Signal Processing Letters* 21.3 (Mar. 2014), pp. 330–333.

[47] Nair Maria Maia De Abreu. “Old and new results on algebraic connectivity of graphs”. In: *Linear algebra and its app.* 423.1 (2007), pp. 53–73.

[48] Morris H. Degroot. “Reaching a Consensus”. In: *Journal of the American Statistical Association* 69.345 (1974), pp. 118–121.

[49] J. A. Deri and J. M. F. Moura. “Spectral Projector-Based Graph Fourier Transforms”. In: *IEEE J. Sel. Topics Signal Process.* 11.6 (Sept. 2017), pp. 785–795.

[50] J. A. Deri and J. M. F. Moura. “Taxi data in New York city: A network perspective”. In: *Asilomar Conf. on Signals, Systems and Computers*. Nov. 2015, pp. 1829–1833.

[51] *Dgraph*. 2019. URL: `https://dgraph.io` (visited on 09/30/2019).

[52] A. G. Dimakis et al. “Gossip Algorithms for Distributed Signal Processing”. In: *Proceedings of the IEEE* 98.11 (Nov. 2010), pp. 1847–1864.

[53] A. Domínguez. “Highlights in the History of the Fourier Transform”. In: *IEEE Pulse* 7.1 (2016), pp. 53–61.

[54] David L. Donoho and Philip B. Stark. “Uncertainty Principles and Signal Recovery”. In: *SIAM J. Appl. Math* 49.3 (1989), pp. 906–931.

[55] Petros Drineas and Michael W Mahoney. “Lectures on randomized numerical linear algebra”. In: *The Mathematics of Data* 25 (2018), p. 1.

[56] V.N. Ekambaram et al. “Circulant structures and graph signal processing”. In: *Int. Conf. on Image Process. (ICIP)*. Sept. 2013, pp. 834–838.

[57] V.N. Ekambaram et al. “Critically-sampled perfect-reconstruction spline-wavelet filterbanks for graph signals”. In: *Global Conf. on Signal and Information Process. (GlobalSIP)*. Dec. 2013, pp. 475–478.

[58] V.N. Ekambaram et al. “Spline-Like Wavelet Filterbanks for Multiresolution Analysis of Graph-Structured Data”. In: *IEEE Trans. Signal Inf. Process. Netw.* 1.4 (Dec. 2015), pp. 268–278.

[59] M. Elad and A. M. Bruckstein. “A generalized uncertainty principle and sparse representation in pairs of bases”. In: *IEEE Trans. Inf. Theory* 48.9 (Sept. 2002), pp. 2558–2567.

[60] Michael Elad. *Sparse and Redundant Representations*. Springer, 2013.

[61] David E. Evans and Raphael Høegh-Krohn. “Spectral Properties of Positive Maps on C*-Algebras”. In: *Journal of the London Mathematical Society* 17.2 (1978), pp. 345–355.

[62] Isabel Faria. “Multiplicity of integer roots of polynomials of graphs”. In: *Linear Algebra and its Applications* 229 (1995), pp. 15–35.

[63] Miroslav Fiedler. “Algebraic connectivity of graphs”. In: *Czechoslovak mathematical journal* 23.2 (1973), pp. 298–305.

[64] Miroslav Fiedler. *Special Matrices and Their Applications in Numerical Mathematics: Second Edition*. Dover Publications, 2008. ISBN: 0486466752.

[65] Gerald B. Folland and Alladi Sitaram. "The uncertainty principle: A mathematical survey". In: *J. Fourier Anal. Appl.* 3.3 (1997), pp. 207–238.

[66] A. Gadde and A. Ortega. "A probabilistic interpretation of sampling theory of graph signals". In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. Apr. 2015, pp. 3257–3261.

[67] W. Gardner. "Likelihood sensitivity and the Cramer-Rao bound". In: *IEEE Trans. Inf. Theory* 25.4 (July 1979), pp. 491–491.

[68] A. Gavili and X. Zhang. "On the Shift Operator, Graph Frequency, and Optimal Filtering in Graph Signal Processing". In: *IEEE Trans. Signal Process.* 65.23 (Dec. 2017), pp. 6303–6318.

[69] E. Gilbert and D. Slepian. "Doubly Orthogonal Concentrated Polynomials". In: *SIAM Journal on Mathematical Analysis* 8.2 (Apr. 1977), pp. 290–319.

[70] John R. Gilbert and Michael T. Heath. "Computing a Sparse Basis for the Null Space". In: *SIAM Journal on Algebraic Discrete Methods* 8.3 (1987), pp. 446–459.

[71] *Giraph*. 2019. URL: `https://giraph.apache.org` (visited on 09/30/2019).

[72] Robert M. Gower and Peter Richtárik. "Randomized Iterative Methods for Linear Systems". In: *SIAM Journal on Matrix Analysis and Applications* 36.4 (2015), pp. 1660–1690.

[73] Robert Grone. "On the geometry and Laplacian of a graph". In: *Linear Algebra and its Applications* 150 (1991), pp. 167–178.

[74] Will Hamilton, Zhitao Ying, and Jure Leskovec. "Inductive Representation Learning on Large Graphs". In: *Neural Information Processing Systems (NIPS)*. 2017, pp. 1024–1034.

[75] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. *The Spectral Graph Wavelets Toolbox*. URL: `http://wiki.epfl.ch/sgwt` (visited on 01/15/2015).

[76] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. "Wavelets on graphs via spectral graph theory". In: *Applied and Computational Harmonic Analysis* 30.2 (2011), pp. 129–150.

[77] Frank Harary and Edgar M Palmer. *Graphical Enumeration*. Academic Press, 1973.

[78] Moritz Hardt and Aaron Roth. "Beyond Worst-case Analysis in Private Singular Vector Computation". In: *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*. 2013, pp. 331–340.

[79] Fredric J. Harris. *Multirate Signal Processing for Communication Systems*. Prentice Hall, 2004.

[80] A. Hefny, D. Needell, and A. Ramdas. "Rows versus Columns: Randomized Kaczmarz or Gauss–Seidel for Ridge Regression". In: *SIAM Journal on Scientific Computing* 39.5 (2017), S528–S542.

[81] G. T. Herman and L. B. Meyer. "Algebraic reconstruction techniques can be made computationally efficient". In: *IEEE Trans. Medical Imaging* 12.3 (Sept. 1993), pp. 600–609.

[82] David Hilbert. "Ein Beitrag zur Theorie des Legendre'schen Polynoms". In: *Acta Mathematica* 18 (Jan. 1894), pp. 155–159.

[83] Remco van der Hofstad. *Random Graphs and Complex Networks: Volume 1*. Cambridge University Press, 2016.

[84] J J Hopfield. "Neural networks and physical systems with emergent collective computational abilities". In: *Proceedings of the National Academy of Sciences* 79.8 (1982), pp. 2554–2558.

[85] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[86] Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1994.

[87] E. Isufi et al. "Autoregressive Moving Average Graph Filtering". In: *IEEE Trans. on Sig. Process.* 65.2 (Jan. 2017), pp. 274–288.

[88] E. Isufi et al. "Filtering Random Graph Processes Over Random Time-Varying Graphs". In: *IEEE Trans. Signal Process.* 65.16 (Aug. 2017), pp. 4406–4421.

[89] F. Iutzeler et al. "Asynchronous distributed optimization using a randomized alternating direction method of multipliers". In: *IEEE Conf. Dec. Cont.* Dec. 2013, pp. 3671–3676.

[90] Matthew Jackson. *Social and Economic Networks*. Princeton Uni. Press, 2008.

[91] A. Jadbabaie, Jie Lin, and A. S. Morse. "Coordination of groups of mobile autonomous agents using nearest neighbor rules". In: *IEEE Trans. Autom. Control* 48.6 (June 2003), pp. 988–1001.

[92] Raphaël Jungers. *The Joint Spectral Radius: Theory and Applications*. Springer, 2009.

[93] Stefan Kaczmarz. "Angenaherte auflosung von systemen linearer gleichungen". In: *Bull. Internat. Acad. Polon. Sci. Letters A* (1937), pp. 335–357.

[94] Thomas Kailath. *Linear Systems*. Prentice Hall, 1980.

[95] Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Prentice Hall, 2000.

[96] S. Kar and J. Moura. "Distributed Consensus Algorithms in Sensor Networks With Imperfect Communication: Link Failures and Channel Noise". In: *IEEE Trans. Signal Process.* 57.1 (Jan. 2009), pp. 355–369.

[97] S. Kar and J. M. F. Moura. "Sensor Networks With Random Links: Topology Design for Distributed Consensus". In: *IEEE Trans. Signal Process.* 56.7 (July 2008), pp. 3315–3326.

[98] S. Kar, J. M. F. Moura, and K. Ramanan. "Distributed Parameter Estimation in Sensor Networks: Nonlinear Observation Models and Imperfect Communication". In: *IEEE Trans. on Inf. Theory* 58.6 (June 2012), pp. 3575–3605.

[99] Samuel Karlin. "Positive Operators". In: *Journal of Mathematics and Mechanics* 8.6 (1959), pp. 907–937.

[100] E. Kaszkurewicz, A. Bhaya, and D.D. Siljak. "On the convergence of parallel asynchronous block-iterative computations". In: *Linear Algebra and its Applications* 131 (1990), pp. 139–160.

[101] Eugenius Kaszkurewicz and Amit Bhaya. *Matrix Diagonal Stability in Systems and Computation*. Birkhäuser, 2000.

[102] D. Kempe, A. Dobra, and J. Gehrke. "Gossip-based computation of aggregate information". In: *IEEE Symp. Found. Comp. Sci.* Oct. 2003, pp. 482–491.

[103] Donald E. Knuth. *The Stanford GraphBase: A Platform for Combinatorial Computing*. Addison-Wesley Professional, 1993.

[104] Paul J. Koprowski. "Graph theoretic uncertainty and feasibility". In: *arXiv: 1603.02059* (Feb. 2016).

[105] Vijay Kumar, Daniela Rus, and Gaurav S. Sukhatme. "Networked Robots". In: *Springer Handbook of Robotics*. Springer Berlin Heidelberg, 2008, pp. 943–958.

[106] Qi Lei, Kai Zhong, and Inderjit S Dhillon. "Coordinate-wise Power Method". In: *Advances in Neural Inf. Process. Systems (NIPS)*. 2016, pp. 2064–2072.

[107] D. Leventhal and A. S. Lewis. "Randomized Methods for Linear Constraints: Convergence Rates and Conditioning". In: *Mathematics of Operations Research* 35.3 (2010), pp. 641–654.

[108] A. Loukas, A. Simonetto, and G. Leus. "Distributed Autoregressive Moving Average Graph Filters". In: *IEEE Sig. Process. Letters* 22.11 (Nov. 2015), pp. 1931–1935.

[109] Andreas Loukas. "Distributed Graph Filters". PhD thesis. Delft University of Technology, Mar. 2015.

[110] Anna Ma, Deanna Needell, and Aaditya Ramdas. "Convergence Properties of the Randomized Extended Gauss–Seidel and Kaczmarz Methods". In: *SIAM Journal on Matrix Analysis and Applications* 36.4 (2015), pp. 1590–1604.

[111] Hao Ma et al. "Mining Social Networks Using Heat Diffusion Processes for Marketing Candidates Selection". In: *ACM Conf. Inf. and Knowledge Management*. ACM, 2008, pp. 233–242.

[112] Wilhelm Magnus. "On the Spectrum of Hilbert's Matrix". In: *American Journal of Mathematics* 72.4 (1950), pp. 699–704.

[113] A. Marques et al. "Sampling of graph signals with successive local aggregations". In: *IEEE Trans. Signal Process.* 64.7 (Apr. 2016), pp. 1832–1843.

[114] A. G. Marques et al. "Stationary Graph Processes and Spectral Estimation". In: *IEEE Trans. Signal Process.* 65.22 (Nov. 2017), pp. 5911–5926.

[115] Per-Gunnar Martinsson and Joel Tropp. "Randomized Numerical Linear Algebra: Foundations and Algorithms". In: *arXiv:2002.01387v1* (Feb. 2020).

[116] Roy Mathias. "The spectral norm of a nonnegative matrix". In: *Linear Algebra and its Applications* 139 (1990), pp. 269–284.

[117] Russell Merris. "Laplacian graph eigenvectors". In: *Linear Algebra and its Applications* 278.1 (1998), pp. 221–236.

[118] Russell Merris. "Laplacian matrices of graphs: a survey". In: *Linear Algebra and its Applications* 197 (1994), pp. 143–176.

[119] C. D. Meyer. *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics, 2000.

[120] W. Mills, C. Mullis, and R. Roberts. "Digital filter realizations without overflow oscillations". In: *IEEE Trans. on Acoustics, Speech, and Signal Process.* 26.4 (Aug. 1978), pp. 334–338.

[121] Ioannis Mitliagkas, Constantine Caramanis, and Prateek Jain. "Memory Limited, Streaming PCA". In: *Advances in Neural Information Processing Systems 26*. 2013, pp. 2886–2894.

[122] Jacob D. Moorman et al. "Randomized Kaczmarz with Averaging". In: *arXiv* 2002.04126v1 (2020).

[123] C. Mullis and R. Roberts. "Synthesis of minimum roundoff noise fixed point digital filters". In: *IEEE Trans. on Circ. Sys.* 23.9 (Sept. 1976), pp. 551–562.

[124] Mikhail Muzychuk and Mikhail Klin. "On graphs with three eigenvalues". In: *Discrete Mathematics* 189.1-3 (1998), pp. 191–207.

[125] S.K. Narang, A. Gadde, and A. Ortega. "Signal processing techniques for interpolation in graph structured data". In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. May 2013, pp. 5445–5449.

[126] S.K. Narang and A. Ortega. “Compact Support Biorthogonal Wavelet Filterbanks for Arbitrary Undirected Graphs”. In: *IEEE Trans. Signal Process.* 61.19 (Oct. 2013), pp. 4673–4685.

[127] S.K. Narang and A. Ortega. “Downsampling graphs using spectral theory”. In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. May 2011, pp. 4208–4211.

[128] S.K. Narang and A. Ortega. *Graph Bior wavelet toolbox*. 2013. URL: `http://biron.usc.edu/wiki/index.php/Graph_Filterbanks` (visited on 01/15/2015).

[129] S.K. Narang and A. Ortega. “Perfect Reconstruction Two-Channel Wavelet Filter Banks for Graph Structured Data”. In: *IEEE Trans. Signal Process.* 60.6 (June 2012), pp. 2786–2799.

[130] A. Nedic and A. Ozdaglar. “Distributed Subgradient Methods for Multi-Agent Optimization”. In: *IEEE Trans. Autom. Control* 54.1 (Jan. 2009), pp. 48–61.

[131] A. Nedić and A. Olshevsky. “Distributed Optimization Over Time-Varying Directed Graphs”. In: *IEEE Trans. Autom. Control* 60.3 (Mar. 2015), pp. 601–615.

[132] Deanna Needell. “Randomized Kaczmarz solver for noisy linear systems”. In: *BIT Numerical Mathematics* 50.2 (June 2010), pp. 395–403.

[133] M. E. J. Newman. *Network data*. 2013. URL: `http://www-personal.umich.edu/~mejn/netdata/` (visited on 04/06/2016).

[134] M. E. J. Newman. *Networks: An Introduction*. Oxford Uni. Press, 2010.

[135] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. “On Spectral Clustering: Analysis and an algorithm”. In: *NIPS 14*. 2002, pp. 849–856.

[136] H.Q. Nguyen and M.N. Do. “Downsampling of Signals on Graphs Via Maximum Spanning Trees”. In: *IEEE Trans. Signal Process.* 63.1 (Jan. 2015), pp. 182–191.

[137] R. Olfati-Saber, J. A. Fax, and R. M. Murray. “Consensus and Cooperation in Networked Multi-Agent Systems”. In: *Proceedings of the IEEE* 95.1 (Jan. 2007), pp. 215–233.

[138] A. Ortega et al. “Graph Signal Processing: Overview, Challenges, and Applications”. In: *Proceedings of the IEEE* 106.5 (May 2018), pp. 808–828.

[139] Lawrence Page et al. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report. Stanford InfoLab, Nov. 1999.

[140] B. Pasdeloup et al. “Toward an uncertainty principle for weighted graphs”. In: *European Signal Process. Conf. (EUSIPCO)*. Aug. 2015, pp. 1496–1500.

[141] Z. Peng et al. “ARock: An Algorithmic Framework for Asynchronous Parallel Coordinate Updates”. In: *SIAM Journal on Scientific Computing* 38.5 (2016), A2851–A2879.

[142] Bethany Percha et al. “Transition from local to global phase synchrony in small world neural network and its possible implications for epilepsy”. In: *Phys. Rev. E* 72 (3 Sept. 2005), p. 031909.

[143] Nathanael Perraudin et al. “Global and local uncertainty principles for signals on graphs”. In: *APSIPA Trans. Signal Inf. Process.* 7 (2018), e3.

[144] Qing Qu, Ju Sun, and John Wright. “Finding a sparse vector in a subspace: Linear sparsity using alternating directions”. In: *Advances in Neural Information Processing Systems 27*. 2014, pp. 3401–3409.

[145] B. Ricaud and B. Torrésani. “Refined Support and Entropic Uncertainty Inequalities”. In: *IEEE Trans. Inf. Theory* 59.7 (July 2013), pp. 4272–4279.

[146] D. Romero, V. N. Ioannidis, and G. B. Giannakis. “Kernel-Based Reconstruction of Space-Time Functions on Dynamic Graphs”. In: *IEEE J. Sel. Topics Signal Process.* 11.6 (Sept. 2017), pp. 856–869.

[147] D. Romero, M. Ma, and G. B. Giannakis. “Kernel-Based Reconstruction of Graph Signals”. In: *IEEE Trans. Signal Process.* 65.3 (Feb. 2017), pp. 764–778.

[148] Gian-Carlo Rota and Gilbert Strang. “A note on the joint spectral radius”. In: *Proceedings of the Netherlands Academy* 22.4 (1960), pp. 379–381.

[149] S. Safavi and U. A. Khan. “Revisiting Finite-Time Distributed Algorithms via Successive Nulling of Eigenvalues”. In: *IEEE Sig. Process. Letters* 22.1 (Jan. 2015), pp. 54–57.

[150] A. Sandryhaila, S. Kar, and J. M. F. Moura. “Finite-time distributed consensus through graph filters”. In: *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*. May 2014, pp. 1080–1084.

[151] A. Sandryhaila and J. M. F. Moura. “Big Data Analysis with Signal Processing on Graphs: Representation and processing of massive data sets with irregular structure”. In: *IEEE Signal Process. Mag.* 31.5 (Sept. 2014), pp. 80–90.

[152] A. Sandryhaila and J. M. F. Moura. “Discrete Signal Processing on Graphs”. In: *IEEE Trans. Signal Process.* 61.7 (Apr. 2013), pp. 1644–1656.

[153] A. Sandryhaila and J. M. F. Moura. “Discrete Signal Processing on Graphs: Frequency Analysis”. In: *IEEE Trans. Signal Process.* 62.12 (June 2014), pp. 3042–3054.

[154] S. Segarra, A. G. Marques, and A. Ribeiro. “Distributed implementation of linear network operators using graph filters”. In: *Allerton Conference on Communication, Control, and Computing*. Sept. 2015, pp. 1406–1413.

[155] Ohad Shamir. “A stochastic PCA and SVD algorithm with an exponential convergence rate”. In: *Proc. of the 32st Int. Conf. Machine Learning (ICML 2015)*. 2015, pp. 144–152.

[156] Ohad Shamir. “Fast Stochastic Algorithms for SVD and PCA: Convergence Properties and Convexity”. In: *Proc. of the 33st Int. Conf. Machine Learning (ICML 2016)*. 2016, pp. 248–256.

[157] X. Shi et al. “Infinite Impulse Response Graph Filters in Wireless Sensor Networks”. In: *IEEE Sig. Process. Letters* 22.8 (Aug. 2015), pp. 1113–1117.

[158] D. I. Shuman et al. “Distributed Signal Processing via Chebyshev Polynomial Approximation”. In: *IEEE Trans. on Sig. and Inf. Process. Net.* 4.4 (Dec. 2018), pp. 736–751.

[159] D. I. Shuman et al. “Spectrum-Adapted Tight Graph Wavelet and Vertex-Frequency Frames”. In: *IEEE Trans. Signal Process.* 63.16 (Aug. 2015), pp. 4223–4235.

[160] D.I. Shuman et al. “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains”. In: *IEEE Signal Process. Mag.* 30.3 (May 2013), pp. 83–98.

[161] D. Slepian. “Prolate spheroidal wave functions, Fourier analysis, and uncertainty - V: the discrete case”. In: *The Bell System Technical Journal* 57.5 (May 1978), pp. 1371–1430.

[162] D. Slepian and H. O. Pollak. “Prolate spheroidal wave functions, Fourier analysis and uncertainty - I”. In: *The Bell System Technical Journal* 40.1 (Jan. 1961), pp. 43–63.

[163] *Spark*. 2019. URL: `https://spark.apache.org` (visited on 09/30/2019).

[164] Gilbert Strang and Truong Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.

[165] Thomas Strohmer and Roman Vershynin. “A Randomized Kaczmarz Algorithm with Exponential Convergence”. In: *J. of Fourier Anal. and Appl.* 15 (2009), pp. 262–278.

[166] Zhendong Sun and Shuzhi Sam Ge. *Stability theory of switched dynamical systems*. Springer, 2011.

[167] A. Tahbaz-Salehi and A. Jadbabaie. “A Necessary and Sufficient Condition for Consensus Over Random Networks”. In: *IEEE Trans. on Auto. Control* 53.3 (Apr. 2008), pp. 791–795.

[168] Oguzhan Teke and Palghat P. Vaidyanathan. “Asynchronous Nonlinear Updates on Graphs”. In: *Asilomar Conference on Signals, Systems, and Computers*. Oct. 2018, pp. 998–1002. DOI: `10.1109/ACSSC.2018.8645351`.

[169] Oguzhan Teke and Palghat P. Vaidyanathan. “Discrete uncertainty principles on graphs”. In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2016, pp. 1475–1479. DOI: 10.1109/ACSSC.2016.7869622.

[170] Oguzhan Teke and Palghat P. Vaidyanathan. “Energy Compaction Filters on Graphs”. In: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Nov. 2018, pp. 783–787. DOI: 10.1109/GlobalSIP.2018.8646570.

[171] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending classical multirate signal processing theory to graphs”. In: *Proceedings SPIE, Wavelets and Sparsity XVII*. Vol. 10394. Aug. 2017, pp. 354–365. DOI: 10.1117/12.2272362.

[172] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending Classical Multirate Signal Processing Theory to Graphs – Part I: Fundamentals”. In: *IEEE Transactions on Signal Processing* 65.2 (Jan. 2017), pp. 409–422. DOI: 10.1109/TSP.2016.2617833.

[173] Oguzhan Teke and Palghat P. Vaidyanathan. “Extending Classical Multirate Signal Processing Theory to Graphs – Part II: M-Channel Filter Banks”. In: *IEEE Transactions on Signal Processing* 65.2 (Jan. 2017), pp. 423–437. DOI: 10.1109/TSP.2016.2620111.

[174] Oguzhan Teke and Palghat P. Vaidyanathan. “Fundamentals of multirate graph signal processing”. In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2015, pp. 1791–1795. DOI: 10.1109/ACSSC.2015.7421460.

[175] Oguzhan Teke and Palghat P. Vaidyanathan. “Graph filter banks with M-channels, maximal decimation, and perfect reconstruction”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2016, pp. 4089–4093. DOI: 10.1109/ICASSP.2016.7472446.

[176] Oguzhan Teke and Palghat P. Vaidyanathan. “IIR Filtering on Graphs with Random Node-Asynchronous Updates”. In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 3945–3960. DOI: 10.1109/TSP.2020.3004912.

[177] Oguzhan Teke and Palghat P. Vaidyanathan. “Linear systems on graphs”. In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Dec. 2016, pp. 385–389. DOI: 10.1109/GlobalSIP.2016.7905869.

[178] Oguzhan Teke and Palghat P. Vaidyanathan. “Node-asynchronous Implementation of Rational Filters on Graphs”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2019, pp. 7530–7534. DOI: 10.1109/ICASSP.2019.8682946.

[179] Oguzhan Teke and Palghat P. Vaidyanathan. “Node-Asynchronous Spectral Clustering on Directed Graphs”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2020, pp. 5325–5329. DOI: `10.1109/ICASSP40776.2020.9054241`.

[180] Oguzhan Teke and Palghat P. Vaidyanathan. “Random Asynchronous Linear Systems: Frequency Response and Mean-Squared Stability”. In: *Submitted to IEEE Transactions on Signal Processing* (2020).

[181] Oguzhan Teke and Palghat P. Vaidyanathan. “Random Node-Asynchronous Updates on Graphs”. In: *IEEE Transactions on Signal Processing* 67.11 (June 2019), pp. 2794–2809. DOI: `10.1109/TSP.2019.2910485`.

[182] Oguzhan Teke and Palghat P. Vaidyanathan. “Randomized Asynchronous Recursions with a Sinusoidal Input”. In: *Asilomar Conference on Signals, Systems, and Computers*. Nov. 2019, pp. 1491–1495. DOI: `10.1109/IEEECONF44664.2019.9048998`.

[183] Oguzhan Teke and Palghat P. Vaidyanathan. “Sparse Eigenvectors of Graphs”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2017, pp. 3904–3908. DOI: `10.1109/ICASSP.2017.7952888`.

[184] Oguzhan Teke and Palghat P. Vaidyanathan. “The Asynchronous Power Iteration: A Graph Signal Perspective”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Apr. 2018, pp. 4059–4063. DOI: `10.1109/ICASSP.2018.8461872`.

[185] Oguzhan Teke and Palghat P. Vaidyanathan. “The Random Component-Wise Power Method”. In: *Proceedings of SPIE, Wavelets and Sparsity XVIII*. Vol. 11138. Sept. 2019, pp. 473–488. DOI: `10.1117/12.2530511`.

[186] Oguzhan Teke and Palghat P. Vaidyanathan. “Time Estimation for Heat Diffusion on Graphs”. In: *Asilomar Conference on Signals, Systems, and Computers*. Oct. 2017, pp. 1963–1967. DOI: `10.1109/ACSSC.2017.8335709`.

[187] Oguzhan Teke and Palghat P. Vaidyanathan. “Uncertainty Principles and Sparse Eigenvectors of Graphs”. In: *IEEE Transactions on Signal Processing* 65.20 (Oct. 2017), pp. 5406–5420. DOI: `10.1109/TSP.2017.2731299`.

[188] Text to Matrix Generator. *http://scgroup20.ceid.upatras.gr:8000/tmg/*. 2019. URL: `http://scgroup20.ceid.upatras.gr:8000/tmg/` (visited on 06/30/2019).

[189] Dorina Thanou et al. “Learning heat diffusion graphs”. In: *IEEE Trans. Signal Inf. Process. Netw.* 3.3 (Sept. 2017), pp. 484–499.

[190] M. J. Todd. *The Computation of Fixed Points and Applications*. Springer, 1976.

[191] Linh V. Tran, Van H. Vu, and Ke Wang. “Sparse random graphs: Eigenvalues and eigenvectors”. In: *Random Structures & Algorithms* 42.1 (2013), pp. 110–134.

[192] Nicolas Tremblay and Pierre Borgnat. “Subgraph-based filterbanks for graph signals”. In: *IEEE Trans. Signal Process.* 64.15 (Aug. 2016), pp. 3827–3840.

[193] K. I. Tsianos, S. Lawlor, and M. G. Rabbat. “Push-Sum Distributed Dual Averaging for convex optimization”. In: *IEEE Conf. Decision Control*. Dec. 2012, pp. 5453–5458.

[194] J. Tsitsiklis, D. Bertsekas, and M. Athans. “Distributed asynchronous deterministic and stochastic gradient optimization algorithms”. In: *IEEE Trans. on Auto. Control* 31.9 (Sept. 1986), pp. 803–812.

[195] M. Tsitsvero, S. Barbarossa, and P. Di Lorenzo. “Signals on Graphs: Uncertainty Principle and Sampling”. In: *IEEE Trans. Signal Process.* 64.18 (Sept. 2016), pp. 4845–4860.

[196] Peter J. Uhlhaas and Wolf Singer. “Neural Synchrony in Brain Disorders: Relevance for Cognitive Dysfunctions and Pathophysiology”. In: *Neuron* 52.1 (2006), pp. 155–16.

[197] P. Vaidyanathan and V. Liu. “An improved sufficient condition for absence of limit cycles in digital filters”. In: *IEEE Trans. on Circuits and Systems* 34.3 (Mar. 1987), pp. 319–322.

[198] P. P. Vaidyanathan. “Low-Noise and Low-Sensitivity Digital Filters”. In: *Handbook of Digital Signal Processing*. Ed. by Douglas F. Elliott. Academic Press, 1987, pp. 359–479.

[199] P. P. Vaidyanathan. “Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial”. In: *Proceedings of the IEEE* 78.1 (Jan. 1990), pp. 56–93.

[200] P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Englewood Cliffs, N.J. Prentice Hall, 1993.

[201] Harry L. Van Trees. *Detection, Estimation, and Modulation Theory, Part I*. Wiley, 1968.

[202] Harry L. Van Trees. *Optimum Array Processing*. Wiley, 2002.

[203] Arun Venkitaraman, Saikat Chatterjee, and Peter Händel. “Kernel Regression for Signals over Graphs”. In: *arXiv: 1706.02191v4* (July 2018).

[204] Martin Vetterli and Jelena Kovacevic. *Wavelets and Subband Coding*. Prentice-Hall, 1995.

[205] D. Van De Ville, R. Demesmaeker, and M. G. Preti. “When Slepian Meets Fiedler: Putting a Focus on the Graph Spectrum”. In: *IEEE Signal Process. Lett.* 24.7 (July 2017), pp. 1001–1004.

[206] Jialei Wang et al. “Efficient coordinate-wise leading eigenvector computation”. In: *arXiv* 1702.07834 (2017).

[207] Xiaohan Wang, Pengfei Liu, and Yuantao Gu. “Local-Set-Based Graph Signal Reconstruction”. In: *IEEE Trans. Signal Process.* 63.9 (May 2015), pp. 2432–2444.

[208] Manfred K. Warmuth and Dima Kuzmin. “Randomized Online PCA Algorithms with Regret Bounds that are Logarithmic in the Dimension”. In: *The Journal of Machine Learning Research* 9 (2008), pp. 2287–2320.

[209] David S. Watkins. “Product Eigenvalue Problems”. In: *SIAM Review* 47.1 (2005), pp. 3–40.

[210] Stephen J. Wright. “Coordinate descent algorithms”. In: *Mathematical Programming* 151.1 (June 2015), pp. 3–34.

[211] Lin Xiao and Stephen Boyd. “Fast linear iterations for distributed averaging”. In: *Systems & Control Letters* 53.1 (2004), pp. 65–78.

[212] O. Younis and S. Fahmy. “HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks”. In: *IEEE Transactions on Mobile Computing* 3.4 (Oct. 2004), pp. 366–379.

[213] Yong Zeng and Shu Wu. *State-Space Models: Applications in Economics and Finance*. Springer-Verlag New York, 2013.

[214] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. “Semi-supervised Learning Using Gaussian Fields and Harmonic Functions”. In: *International Conference on Machine Learning (ICML)*. 2003, pp. 912–919.