CRITICALLY SAMPLED WAVELET FILTERBANKS ON GRAPHS

by

Sunil K. Narang

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

August 2012

Dedication

To my teachers who taught me to be honest and hardworking.

Acknowledgements

It is a pleasure to thank those who made this dissertation possible: my teachers, collaborators, family and friends. First and foremost, I would like to thank my advisor, Professor Antonio Ortega, whose expertise, continuous guidance, and understanding helped me evolve from a naive undergraduate student into a practitioner of critical thinking. The best thing I liked about Antonio is that he gave me the freedom to explore on my own, and at the same time guidance to remain focused on the big picture and not getting lost in minor details. I could not have wished for a better or friendlier supervisor. I would also like to thank Professor Bhaskar Krishnamachari and Professor Yan Liu for serving on my dissertation committee, as well as to Professor C.-C. Jay Kuo and Professor Srikanth Narayanan for being members in my qualifying exam committee. It is a privilege to have their advice on my work.

I am also indebted to my colleagues in the Compression Research Group, who taught me many things through various interactions and discussions. Specially, I owe my gratitude to Dr. Godwin Shen from whom I learnt so much about distributed compression, and whose PhD research became the foundation and motivation for my own research. Your friendship both within and outside of our research world has been a great source of strength for me. Special thanks is also due to Dr. Woo-Shik Kim, with whom I had a lot of memorable collaborations and who first introduced me to the Korean cuisine. I would also like to acknowledge the inputs and collaborations of Sungwon Lee, Yenting (Greg) Lin, Yung-Hsuan (Jessie) Chao, and Akshay Gadde, without which this thesis would have been incomplete.

Furthermore, I owe special thanks to Professor Urbashi Mitra, Professor Bhaskar Krishnamachari and Dr. Marco Levorato, for giving me a wonderful opportunity to collaborate with you. The time spent in our joint efforts has truly enriched my experience at USC. I would also like to thank Dr. Grzegorz M Swirszcz, and Dr. Tomasz Nowicki from IBM Watson research lab, for our discussions on distributed compression during my summer internship in New York. I am also

grateful to Xavier Perez Trufero from Polytechnic University of Catalonia and Apostol T. Gjika from Politecnico Di Torino, for all the wonderful discussions and collaborations.

Finally, and most importantly, I would like to thank my wife Shanu, whose encouragement, quiet patience and unwavering love has been the driving force throughout my PhD, and before. A big hug and a heartfelt "thanks" to you. I also thank my dad, Rambeer Singh, for his faith in me and for allowing me and supporting me in every way to be as ambitious as I wanted.

Table of Contents

List of Tables

List of Figures

Abstract

Emerging data mining applications will have to operate on datasets defined on graphs. Examples of such datasets include online document networks, social networks, and transportation networks etc. The data on these graphs can be visualized as a finite collection of samples, a *graph-signal* which can be defined as the information attached to each node (scalar or vector values mapped to the set of vertices/edges) of the graph. Major challenges are posed by the size of these datasets, making it difficult to visualize, process, analyze and act on the information available. Wavelets have been popular for traditional signal processing problems (e.g., compression, segmentation, denoising) because they allow signal representations where a variety of trade-offs between spatial (or temporal) resolution and frequency resolution can be achieved. In this research, we seek to leverage novel basic wavelet techniques for graph data, and apply them to realistic information analytics problems. The primary contribution of this thesis is to design *critically sampled wavelet filterbanks on graphs*, which provide a local analysis in the graph (localized within a few hops of a target node), while capturing spectral/frequency information of the graph-signals. The graphs in our study are simple undirected graphs. We first design "one-dimensional" two-channel filterbanks on *bipartite graphs*, and then extend them to any arbitrary graph. The filterbanks come in two flavors, depending upon the chosen downsampling method: i) lifting wavelet filterbanks and ii) spectral wavelet filterbanks. For bipartite graphs we define a *spectral folding* phenomenon, analogous to *aliasing* in regular signals, that helps us define filterbank constraints in simple terms. For arbitrary graphs we propose two choices: a) to approximate the graph as a single bipartite graph and apply "one-dimensional" filterbanks, or b) to decompose the graph into multiple bipartite subgraphs and apply "multi-dimensional" filterbanks. All our proposed filterbanks designs are critically sampled and perfect reconstruction. To the best of our knowledge, no such filterbanks have been proposed before. The tools proposed in this thesis make it possible to develop i) multiresolution representations of graphs, ii) *edge-aware* processing of regular signals, iii) anomaly detection in datasets, and iv) sampling of large networks.

Chapter 1

Introduction

1.1 Motivation

Our work is focused on constructing linear wavelet-like transforms for functions defined on the vertices of an arbitrary finite weighted graph. Graphs provide a very flexible model for representing data in many domains. Many networks such as biological networks [48], social networks [7, 13] and sensor networks [38, 47] etc. have a natural interpretation in terms of finite graphs with vertices as data-sources and links established based on connectivity, similarity, ties etc. The data on these graphs can be visualized as a finite collection of samples, which we term *graph-signals*. For example, graphs can be used to represent irregularly sampled datasets in Euclidean spaces such as regular grids with missing samples. In many machine learning applications multi-dimensional datasets can be represented as *point-clouds* of vectors and links are established between data sources based on the distance between their feature-vectors. In computer vision, meshes are polygon graphs in 2D/3D space and the attributes of the sampled points (coordinates, intensity etc) constitute the graph-signals. The graph-signal formulation can also be used to solve systems of partial differential equations using finite element analysis (grid based solution). The sizes (number of nodes) of the graphs in these applications can be very large, which presents computational and technical challenges for the purpose of storage, analysis etc. In some other applications such as wireless sensor-networks, the data-exchanges between far-off nodes can be expensive (bandwidth, latency, energy constraints issues). Therefore, instead of operating on the original graph, it would be desirable to find and operate on smaller graphs with fewer nodes and data representing a

smooth[1] approximation of the original data. Moreover, such systems need to employ localized operations which could be computed at each node by using data from a small neighborhood of nodes around it. Multi-channel wavelet filterbanks, widely used as a signal processing tool for the sparse representation of signals, possess both these features (i.e. smooth approximations and localized operations[2]). For example, a two channel wavelet transform splits the sample space into an approximation subspace which contains a smoother (coarser) version of the original signal and a detail subspace containing additional details required to perfectly reconstruct the original signal. A discussion of the construction and analysis of wavelet filterbanks for regular signals can be found in standard textbooks such as [44]. While wavelet transform-based techniques would seem well suited to provide efficient local analysis, a major obstacle to their application to graphs is that these, unlike images, are not regularly structured. For graphs, traditional notions of dimensions along which to filter the data do not hold.

1.2 Background

Transform techniques for graph analysis can be broadly divided into a) global methods, e.g., those using concepts of graph spectral theory, and b) wavelet like localized methods which are supported on a local neighborhood around each node. Global methods are often based on the Laplacian matrix, whose eigenvalues and eigenvectors contain global information about the shape of the graph. Common applications of global methods include, graph partitioning (graph-cuts) [13], simplification, graph based feature extraction [24] and graph matching [51]. A comprehensive discussion of global methods can be found in [3] and [26]. In addition to uncovering mostly global information, global methods usually do not scale well as the graph size increases, e.g., the time required to perform the eigenvalue decomposition can be significant. The most expensive component in the computation of global methods is eigenvalue decomposition(EVD) of graph matrices, which normally require $\mathcal{O}(N^3)$ arithmetic operations. Note that, decentralized algorithm have been proposed which can greatly simplify the computation of a partial set of principal eigenvectors in graphs. For example, the algorithm proposed by Kempe [20], computes k principal eigenvectors of the graph in $\mathcal{O}(\tau_{mix}N^2)$ decentralized steps, where τ_{mix} is the mixing time of a random walk on a network, and with $\mathcal{O}(k^3)$ computations per node in each round. While these

[1] more generally, it could be any sparse approximation of the original data.
[2] In case of FIR wavelet filters

approaches are good for structural analysis of graph, requiring only a partial set of eigenvectors, they have limited use in applications such as compression and denoising which require full spectral decomposition of the graph.

Researchers have recently focused on developing localized transforms specifically for data defined on graphs. Crovella and Kolaczyk [7] designed wavelet like functions on graphs which are localized in space and time. These graph functions $\psi_{j,k}$ are composed of either shifts or dilations of a single generating function ψ. Wang and Ramchandran [47] proposed graph dependent basis functions for sensor network graphs, which implement an invertible 2-channel like filterbank. There exists a natural spectral interpretation of graph-signals in terms of eigen-functions and eigen-values of graph Laplacian matrix $\mathbf{L}$. Maggioni and Coifman [6] introduced "diffusion wavelets" as the localized basis functions of the eigenspaces of the dyadic powers of a diffusion operator. Hammond et al. [14] construct a class of wavelet operators in the *graph spectral domain*, i.e., the space of eigenfunctions of the graph Laplacian matrix $\mathbf{L}$. These eigenfunctions provide a spectral decomposition for data on a graph similar to the Fourier transform for standard signals. A common drawback of all of these filterbank designs is that they are not critically sampled : the output of the transform is not downsampled and there is oversampling by a factor equal to the number of channels in the filterbank. Unlike classical wavelet transforms which have well-understood downsampling/upsampling operations, there is no obvious way in graphs to downsample nodes in a regular manner, since the neighboring nodes vary in number. Lifting based wavelet transforms have been proposed in [45, 18] for graphs in an Euclidean Space. However, these transforms require a Euclidean embedding of the graph. Shen and Ortega [38, 40] have applied wavelet lifting transforms on spanning trees for a wireless sensor network application, where invertibility is guaranteed for any tree, as long as nodes in the tree are partitioned into two sets (even and odd nodes) and the transform is structured by modifying even nodes based on odd nodes (and vice versa).

1.3 Contributions

The objective of this thesis is to design critically-sampled wavelet-filterbanks on graphs. The building blocks in our proposed designs are *two channel wavelet filterbanks on bipartite graphs*, which provide a decomposition of any graph-signal into a low-pass (smooth) graph-signal, and a

high-pass (detail) graph-signal. These designs come in two flavors: i) lifting wavelet filterbanks, and ii) spectral wavelet filterbanks. We choose bipartite graphs because they are a natural choice for implementing lifting wavelet filterbanks, and provide easy-to-interpret perfect reconstruction conditions for spectral wavelet filterbanks. For arbitrary graphs we have two choices: a) we can either implement our proposed wavelet filterbanks on a bipartite graph approximation of the original graph, which provides a "one-dimensional" analysis, or b) we can decompose the graph into multiple edge-disjoint bipartite subgraphs, and apply our proposed filterbanks iteratively on each subgraph, leading to a "multi-dimensional" analysis. Our contributions in this thesis can be divided into three major part which we describe in what follows.

1.3.1 Sampling operations in graphs

One of the desired properties of wavelet transforms on graphs is critical sampling: the output wavelet coefficients are equal in number to the input samples. The critical sampling can be achieved by partitioning the set of vertices in the graphs into two subsets (say L and H), such that nodes in L only sample the output of low-pass channel and nodes in H only sample the output of highpass channel. Algebraically, we describe a linear *downsample then upsample* (DU) operation on graphs, in which a set of nodes in the graph are first downsampled (removed) and then upsampled (replaced) by inserting zeros. Especially, we show that for all undirected bipartite graphs, the DU operations lead to a spectral decomposition of the graph-signal where spectral coefficients are reproduced at mirror graph-frequencies around a central frequency. This is a phenomenon we term as *spectrum folding in graphs* as it is analogous to the frequency-folding or "aliasing" effect for regular one-dimensional signals. We utilize this property to define critically sampled operations for implementing wavelet filterbanks on graphs. This is described in detail in Chapter 4.

1.3.2 Two-channel wavelet filterbanks on bipartite graphs

We propose two-channel wavelet filterbanks on bipartite graphs which are critically sampled, and perfect reconstruction. The filterbanks come in two flavors, depending upon the chosen downsampling method: a) lifting wavelet filterbanks, and b) spectral wavelet filterbanks. The block diagrams of these two designs are shown in Figures 1.1 and 1.2, respectively.

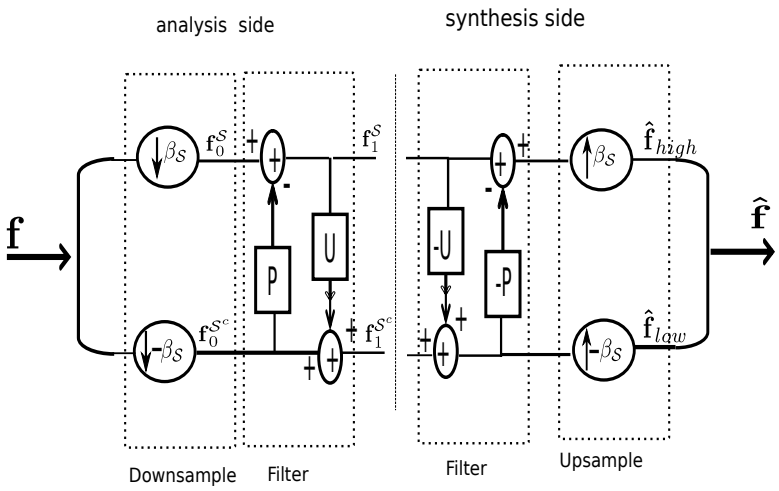


Figure 1.1: Lifting Scheme: Downsampling followed by filtering

The *lifting wavelet filterbanks* are critically sampled and invertible by construction. These transforms require splitting the vertex set into two disjoint sets, often called the *even*, and *odd* sets, given which the transform is computed only on the links between nodes in different sets. The previous splitting schemes for lifting transforms required either the coordinates of the nodes in some Euclidean embedding (eg. in [45]), or a specific graph structure (eq., trees in [38, 40]). Our contribution is that we formulate the problem of splitting nodes as a bipartite subgraph approximation problem, and provide greedy heuristics to compute optimal subgraphs. We also apply these graph-based lifting transforms in a data-gathering application in wireless sensor networks (WSN).

Next, we propose *spectral wavelet filterbanks*, in which filtering and downsampling is chosen so as to guarantee perfect reconstruction. We design *spectral filters*, and use the *spectral folding phenomenon* to provide *necessary and sufficient* conditions for aliasing cancellation, perfect-reconstruction and orthogonality in these filterbanks. As a practical solution we propose a *graph-quadrature mirror filterbank* (referred to as graph-QMF) design for bipartite graphs which has all the above mentioned properties. However, the exact realizations of the graph-QMF filters do not have *compact support* on the graph, and approximations of exact solutions as compactly supported solutions incur small reconstruction error and loss of orthogonality. As an alternative, we design biorthogonal wavelet filterbanks which have compact support and yet provide perfect reconstruction of any graph signals.

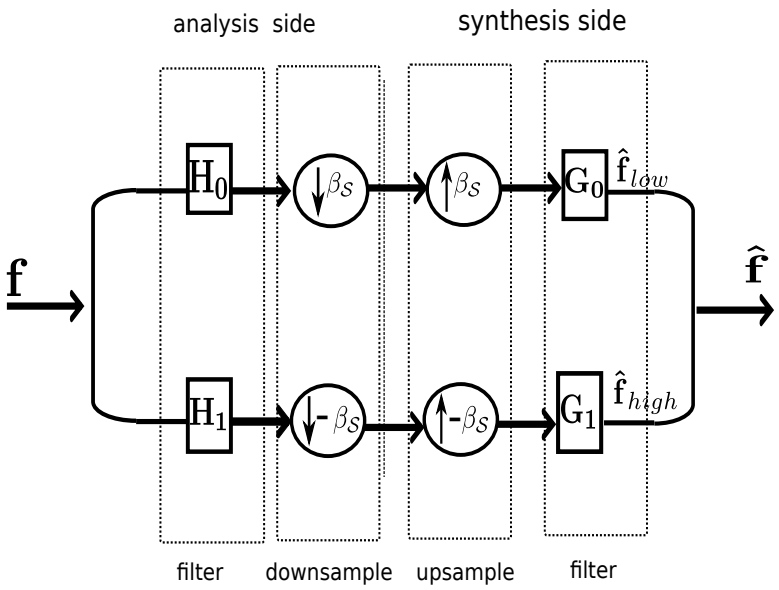


Figure 1.2: Spectral Scheme: Filtering followed by downsampling

1.3.3 Bipartite subgraph decomposition

In order to extend two-channel wavelet filterbanks on bipartite graphs to arbitrary graphs, we formulate a bipartite subgraph decomposition problem, which provides an edge-disjoint collection of K bipartite subgraphs, each with the same vertex set $\mathcal{V}$ and whose union is the original graph. Each of these subgraphs is then used as a separate "dimension" to filter and downsample, leading to a K-dimensional separable wavelet filterbank design. The bipartite subgraph decomposition of graphs is not unique, therefore, we propose metrics to identify optimal decompositions, and propose algorithms to compute them.

1.4 Thesis Statement and Research Questions

It is possible to extend standard DSP techniques for data defined on graphs. In particular, critically sampled wavelet transforms can be designed on graphs which have a spectral interpretation, and provide perfect reconstruction of any signal defined on the graph. The research questions that we answer are:

1. How to define downsampling and upsampling operations on graphs?

2. How to design wavelet filters on graphs, given a set of spatial and spectral constraints?

3. Using these concepts, how to design critically sampled wavelet filterbanks on graphs?

1.5 Publications

The work in this thesis has resulted in following published articles:

- **S. K. Narang** and Antonio Ortega, "*Perfect Reconstruction Two-Channel Wavelet Filter-Banks For Graph Structured Data*", IEEE Transactions on Signal Processing, June 2012
- **S.K. Narang**, Y. H. Chao and A. Ortega, "*Graph-wavelet filterbanks for edge-aware image processing*", to appear in IEEE Statistical Signal Processing (SSP'12)
- **S.K. Narang** and A. Ortega, "*Multi-dimensional separable critically sampled wavelet filterbanks on arbitrary graphs*", IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP'12)
- **S.K. Narang** and A. Ortega, "*Downsampling Graphs Using Spectral Theory*", IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP'11).
- J. P.-Trufero, **S.K. Narang** and A. Ortega, "*Distributed Transforms for Efficient Data Gathering in Arbitrary Networks*", Intl. Conf. on Image Proc. (ICIP'11).
- **S.K. Narang** and A. Ortega, "*Local two-channel critically-sampled filter-banks on graphs*", Intl. Conf. on Image Proc. (ICIP'10).
- **S.K. Narang**, G. Shen and A. Ortega, "*Unidirectional Graph-based Wavelet Transforms for Efficient Data Gathering in Sensor Networks*". IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP'10).
- **S.K. Narang** and A. Ortega, "*Lifting based wavelet transforms on graphs*", Asia-Pacific Sig. and Information Proc. Association (APSIPA ASC'09).
- G. Shen, **S.K. Narang** and A. Ortega, "*Adaptive Distributed Transforms for Irregularly Sampled Wireless Sensor Networks*". In Proc. of 2009 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP'09).

1.6 Summary

In this chapter, we have described the objectives of this thesis, the motivation behind and a summary of our contributions. The rest of the thesis is organized as follows: In Chapter 2, we describe the basic framework required to understand wavelet filterbanks on graphs, and evaluate some of the existing work on wavelet-like transforms on graph, based on the proposed framework. In Chapter 3, we propose critically sampled lifting wavelet transforms on any arbitrary graphs, and discuss a distributed data-gathering application, in which proposed lifting filterbanks are useful. In Chapter, 4 we introduce the theory behind downsampling/upsampling operations on graphs. In particular, we describe a *spectral folding* phenomenon in bipartite graphs which is analogous to *aliasing* in standard regular signals. We use these concepts in Chapter 5, to design two-channel critically sampled wavelet filterbanks on bipartite graphs. In this chapter, we first propose *graph-QMF* filterbanks which are orthogonal and perfect reconstruction but do not have compact support. Then, we considered 1-*hop localized filterbanks* in which the analysis filters are exactly 1-hop localized, but synthesis filters do not have compact support. As an alternative, we propose *graph-Bior* wavelet filterbanks on bipartite graphs in which are perfect reconstruction and in which both analysis and synthesis filters have compact support. These filters are not orthogonal but can be designed to mostly preserve energy. In Chapter 6, we extend the filterbank designs proposed for bipartite graphs to arbitrary graphs via bipartite subgraph decomposition. In Chapter 7, we discuss some applications of the proposed filterbanks. Finally, in Chapter 8, we conclude and describe our future work.

Chapter 2

Basic Theory

In this chapter, we describe the basic theory, helpful to understand the construction of graph wavelet filterbanks. We will use the following notations for the rest of the thesis: we represent matrices and vectors with bold letters, mathematical sets with calligraphic capital letters and scalars with normal letters. A graph can be denoted as $G = (\mathcal{V}, E)$ with vertices (or nodes) in set $\mathcal{V}$ and links (or edges) as tuples (i, j) in E. The graphs considered in our research work are undirected graphs without self-loops and without multiple edges between nodes. The edges can only have positive weights. The size of the graph $N = |\mathcal{V}|$ is the number of nodes and the geodesic distance metric is given as $d(v, m)$, which represents sum of edge weights along the shortest path between nodes u and v, and is considered infinite if u and v are disconnected . The j-hop neighborhood $\mathcal{N}_{j,n} = \{v \in \mathcal{V} : d(v, n) \leq j\}$ of node n is the set of all nodes which are at most j-hop distance away from node n. Algebraically, a graph can be represented with the node-node adjacency matrix $\mathbf{A}$ such that the element $A(i, j)$ is the weight of the edge between node i and j (0 if no edge). The value d_i is the degree of node i, which is the sum of weights of all edges connected to node i, and $\mathbf{D} = diag(\{d_i\})$ denotes the diagonal degree matrix whose i^{th} diagonal entry is d_i. The Laplacian matrix of the graph is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. The Laplacian matrix also has a symmetric normalized form $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, and an asymmetric normalized form $\mathcal{L}_a = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$, where $\mathbf{I}$ is the identity matrix. We denote $< \mathbf{f}_1, \mathbf{f}_2 >$ as the inner-product between vectors $\mathbf{f}_1$ and $\mathbf{f}_2$.

The rest of the chapter is organized as follows: In Section 2.1, we formally define graph signals and graph transforms. In Section 2.2, we describe the spectral representation of graph signal and graph-transforms in terms of eigenvalues and eigenvectors of graph Laplacian matrix. In

Section 2.3, we introduce downsampling upsampling operations as graph transforms on graphs, and in Section 2.4 we utilize these concepts to define a general framework for *two-channel wavelet filterbanks on the graph*. In Section 2.5 we compare and evaluate existing graph transforms based on our proposed framework, and finally we conclude the chapter in Section 2.6.

2.1 Spatial Representation of Graph Signals

A graph signal is a real-valued scalar function $f : \mathcal{V} \to \mathbb{R}$ defined on graph $G = (\mathcal{V}, E)$ such that $f(v)$ is the sample value of function at vertex $v \in \mathcal{V}$.[1] On a finite graph, the graph-signal can be viewed as a sequence or a vector $\mathbf{f} = [f(0), f(1), ..., f(N)]^t$, where the order of arrangement of the samples in the vector is arbitrary and neighborhood (or nearness) information is provided separately by the adjacency matrix $\mathbf{A}$. Graph-signals can, for example, be a set of measured values by sensor network nodes [47] or traffic measurement samples on the edges of an Internet graph [7] or information about the actors in a social network. Further, a graph based transform is defined as a linear transform $\mathbf{T} : \mathbb{R}^N \to \mathbb{R}^M$ applied to the N-node graph-signal space, such that the operation at each node n is a linear combination of the value of the graph-signal $f(n)$ at the node n and the values $f(m)$ on nearby nodes $m \in \mathcal{N}_{j,n}$, i.e.,

$$y(n) =< \mathbf{t}_n \ \mathbf{f} >= T(n,n)f(n) + \sum_{m \in \mathcal{N}_{j,n}} T(n,m)f(m), \tag{2.1}$$

where $\mathbf{t}_n$ is the n^{th} row of the transform $\mathbf{T}$. In analogy to the 1-D regular case, we would sometimes refer to graph-transforms as graph-filters and the elements $T(n,m)$ for $m = 1, 2, ...N$ as the filter coefficients at the n^{th} node [2] A desirable feature of graph filters is *spatial localization*, which typically means that the energy of each basis (i.e., each row) of the graph filter is concentrated in a local region around a node. Let us define $\Delta_k^2(\mathbf{t}_n)$ given as:

$$\Delta_k^2(\mathbf{t}_n) = \frac{1}{||\mathbf{t}_n||^2} \sum_{l \in \mathcal{N}_{k,n}} T(n,l)^2, \tag{2.2}$$

[1] The extension to complex or vector sample values $f(v)$ is straightforward but is not considered in this work.

[2] Not every linear transform is a graph-transform, since graph-transforms, by definition, are defined along the edges in the graph. For example, filter-coefficient $T(n,m)$ can be non-zero only if nodes n and m are connected, i.e., $d(n,m) < \infty$, and the magnitude of $T(n,m)$ usually decreases with increasing distance $d(n,m)$.

to be the fraction of energy of n^{th} basis function (i.e., n^{th} row $\mathbf{t}_n$), in the k-hop neighborhood around node n. A graph transform is said to be strictly k-hop localized, or having a *compact support* in the spatial domain, if $\Delta_k^2(\mathbf{t}_n) = 1$ for all $n = 1, 2, ..N$. Note that spatial localization can also be applied in a weaker sense in which $\Delta_k^2(\mathbf{t}_n)$ is not exactly 1 but very close to it for all $n = 1, 2, ...N$. Our focus in this thesis is to propose graph-filters with *compact support*. .

2.2 Spectral Representation of Graph Signals

The spectral decomposition of graph G is defined can be defined in terms of the set of eigenvalues $\sigma(G)$, and the corresponding eigenvectors $\mathbf{u}_\lambda$, $\lambda \in \sigma(G)$ of the graph Laplacian matrix. The Laplacian matrices $\mathbf{L}$ and $\mathcal{L}$ are both symmetric positive semidefinite matrices and therefore, from the spectral projection theorem, there exists a real unitary matrix $\mathbf{U}$ which diagonalizes $\mathcal{L}$, such that $\mathbf{U}^t\mathcal{L}\mathbf{U} = \Lambda = diag\{\lambda_i\}$ is a non-negative diagonal matrix. In our proposed designs, we use the symmetric normalized form of Laplacian matrix $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$, which is more closely related to random walks in the graphs, and is more appropriate in dealing with non-regular graphs[3]. This leads to an *eigenvalue decomposition* of matrix $\mathcal{L}$ given as

$$\mathcal{L} = \mathbf{U}\Lambda\mathbf{U}^t = \sum_{i=1}^{N} \lambda_i \mathbf{u}_i \mathbf{u}_i^t, \tag{2.3}$$

where the eigenvectors $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_N$, which are columns of $\mathbf{U}$, form a basis in $\mathbb{R}^N$ and $\{0 \leq \lambda_1 \leq \lambda_2 ... \leq \lambda_N\}$ are corresponding eigenvalues. Thus, every graph-signal $\mathbf{f} \in \mathbb{R}^N$ can be decomposed into a linear combination of eigenvectors $\mathbf{u}_i$ given as $\mathbf{f} = \sum_{n=1}^{N} \bar{f}(n)\mathbf{u}_n$. It has been shown in [3, 26] that the eigenvectors of Laplacian matrix provide a harmonic analysis of graph signals which gives a Fourier-like interpretation. The eigenvectors act as the *natural vibration modes* of the graph, and the corresponding eigenvalues as the associated *graph-frequencies*[1]. The *spectrum* $\sigma(G)$ of a graph is defined as the set of eigen-values of its normalized Laplacian matrix, and it is always a

[3]This is because $\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A}$ is the transition matrix of a Markov chain which has the same eigenvalues as $\mathbf{I} - \mathcal{L}$. The eigenvalues of $\mathcal{L}$ are in "normalized" form, i.e, if $\lambda \in \sigma(G)$ then $0 \leq \lambda \leq 2$, and are thus consistent with the eigenvalues in the stochastic processes. Further, the normalization reweighs the edges of graph G so that the degree of each node is equal to 1. Refer to [3] for details.

[1]The mapping $\mathbf{u}_n \to \mathcal{V}$ associates the real numbers $u_n(i), i = \{1, 2, ..., N\}$, with the vertices $\mathcal{V}$ of G. The numbers $u_n(i)$ will be positive, negative or zero. The frequency interpretation of eigenvectors can thus be understood in terms of number of zero-crossings (pair of connected nodes with different signs) of eigenvector $\mathbf{u}_n$ on the graph G. For any finite graph the eigenvectors with large eigenvalues have more zero-crossings (hence high-frequency) than eigenvectors with small eigenvalues. These results are related to 'nodal domain theorems' and readers are directed to [8] for more details.

subset of closed set $[0, 2]$ for any graph G. For the purpose of this thesis, an eigenvector $\mathbf{u}_\lambda$ is either considered to be a "lowpass" eigenvector if eigenvalue $\lambda \leq 1$, or "highpass" eigenvector if $\lambda > 1$. The *graph Fourier transform* (GFT), denoted as $\bar{\mathbf{f}}$, is defined in [14] as the projections of a signal $\mathbf{f}$ on the graph G onto the eigenvectors of G, i.e.,

$$\bar{f}(\lambda) =< \mathbf{u}_\lambda \ , \ \mathbf{f} >= \sum_{i=1}^{N} f(i) u_\lambda(i). \tag{2.4}$$

Note that GFT is an energy preserving transform. A signal is considered "lowpass" (or "highpass") if the energy $|\bar{f}(\lambda)|^2 \approx 0$ for all $\lambda > 1$ (or for all $\lambda \leq 1$). In case of eigenvalues with multiplicity greater than 1 (say $\lambda_1 = \lambda_2 = \lambda$) the eigenvectors $\mathbf{u}_1, \mathbf{u}_2$ are unique up to a unitary transformation in the eigenspace $V_\lambda = V_{\lambda_1} = V_{\lambda_2}$. In this case, we can choose $\lambda_1 \mathbf{u}_1 \mathbf{u}_1^t + \lambda_2 \mathbf{u}_2 \mathbf{u}_2^t = \lambda \mathbf{P}_\lambda$ where $\mathbf{P}_\lambda$ is the projection matrix for eigenspace V_λ. Note that for all symmetric matrices, the dimension of eigenspace V_λ (geometric multiplicity) is equal to the multiplicity of eigenvalue λ (algebraic multiplicity) and the spectral decomposition in (2.3) can be written as

$$\mathcal{L} = \sum_{\lambda \in \sigma(G)} \lambda \sum_{\lambda_i = \lambda} \mathbf{u}_i \mathbf{u}_i^t = \sum_{\lambda \in \sigma(G)} \lambda \mathbf{P}_\lambda. \tag{2.5}$$

The eigenspace projection matrices are idempotent and $\mathbf{P}_\lambda$ and $\mathbf{P}_\gamma$ are orthogonal if λ and γ are distinct eigenvalues of the Laplacian matrix, i.e.,

$$\mathbf{P}_\lambda \mathbf{P}_\gamma = \delta(\lambda - \gamma) \mathbf{P}_\lambda, \tag{2.6}$$

where $\delta(\lambda)$ is the Kronecker delta function.

2.3 Downsampling in Graphs

A downsampling operation on the graph $G = (\mathcal{V}, E)$ can be defined as choosing a subset H of vertex set $\mathcal{V}$ such that all samples of the graph signal $\mathbf{f}$, corresponding to indices not in H, are discarded. A subsequent upsampling operation projects the downsampled signal back to original $\mathbb{R}^N$ space by inserting zeros in place of discarded samples in $H^c = L$. Given such a set H we define a *downsampling function* $\boldsymbol{\beta}_H \in \{-1, +1\}$ given as

$$\beta_H(n) = \begin{cases} 1 & \text{if } n \in H \\ -1 & \text{if } n \in L \end{cases} \tag{2.7}$$

and a diagonal *downsampling matrix* $\mathbf{J}_{\beta_H} = diag\{\beta_H(n)\}$.Note that, by definition $\beta_L(n) = -\beta_H(n)$, therefore everything we derive for β_H can also be derived for β_L with appropriate sign changes. The overall 'downsample then upsample' (DU) operation , using β_H can then be algebraically represented as

$$\begin{aligned} f_{du}(n) &= \frac{1}{2}(1+\beta_H(n))f(n) \\ &= \frac{1}{2}(f(n)+\beta_H(n)f(n)). \end{aligned} \tag{2.8}$$

Thus the signal after DU operation is the sum of the original signal and the signal modulated with $\beta_H(n)$. We can write (2.8) in the matrix form as:

$$\begin{aligned} \mathbf{f}_{du} &= \frac{1}{2}(\mathbf{I}+\mathbf{J}_{\beta_H})\mathbf{f} \\ &= \frac{1}{2}(\mathbf{f}+\mathbf{J}_{\beta_H}\mathbf{f}) \end{aligned} \tag{2.9}$$

Note that $\mathbf{J}_{\beta_H}$ is a symmetric matrix such that $\mathbf{J}^2_{\beta_H} = \mathbf{I}$ (identity matrix). Since the graph-signal after DU operation also belongs to $\mathbb{R}^N$ space, it too has a GFT decomposition $\bar{\mathbf{f}}_{du}$ according to (2.4). The relationship between the GFTs of $\mathbf{f}$ and $\mathbf{f}_{du}$ is given as:

$$\bar{f}_{du}(l) =< \mathbf{u}_l \ , \ \mathbf{f}_{du} >= \frac{1}{2}(< \mathbf{u}_l \ , \ \mathbf{f} > + < \mathbf{u}_l \ , \ \mathbf{J}_{\beta_H}\mathbf{f} >) \tag{2.10}$$

The inner-product $< \mathbf{u}_l \ , \ \mathbf{J}_{\beta_H}\mathbf{f} >$ can also be written as $< \mathbf{J}_{\beta_H}\mathbf{u}_l \ , \ \mathbf{f} >$, which represents the projection of input signal $\mathbf{f}$ onto a modulated eigenvector $\mathbf{J}_{\beta_H}\mathbf{u}_l$. We define this projection as a modulated spectral coefficient $\bar{\mathbf{f}}^d(l)$ and (2.10) can be written as:

$$\bar{f}_{du}(l) = \frac{1}{2}(\bar{f}(l) + < \mathbf{J}_{\beta_H}\mathbf{u}_l \ , \ \mathbf{f} >) = \frac{1}{2}(\bar{f}(l) + \bar{f}^d(l)) \tag{2.11}$$

We describe the effect of this modulation in the spectral domain of graph in Chapter 4. We show that in case of bipartite graphs, the spectrum of the graph is symmetric, and the modulated eigenvectors are also the eigenvectors of the same graph. This phenomenon, which we term as

spectral folding, forms the basis of our two-channel spectral filterbank framework, and will be described in detail in Chapter 4.

2.4 Two-Channel Filterbanks on Graph

A two-channel wavelet filterbank on a graph provides a decomposition of any graph-signal into a lowpass (smooth) graph-signal and a highpass (detail) graph-signal component. The two channels of the filterbanks are characterized by the graph-filters $\{\mathbf{H}_i, \mathbf{G}_i\}_{i\in\{0,1\}}$ and the downsampling operations $\boldsymbol{\beta}_H$ and $\boldsymbol{\beta}_L$ as shown in Figure 2.1. The transform $\mathbf{H}_0$ acts as a lowpass filter, i.e., it transfers the contributions of the low-pass graph-frequencies, which are below some cut-off, and attenuates significantly the graph-frequencies which are above the cut-off. The highpass transform $\mathbf{H}_1$ does the opposite, i.e, it attenuates significantly, the graph-frequencies below some cut-off frequency. The filtering operations in each channel are followed by downsampling operations $\boldsymbol{\beta}_H$ and $\boldsymbol{\beta}_L$, which means that the nodes with membership in the set H store the output of highpass channel while the nodes in the set L store the output of lowpass channel. For critically sampled output we have: $|H| + |L| = N$. Using (2.9), it is easy to see from Figure 2.1 that the output

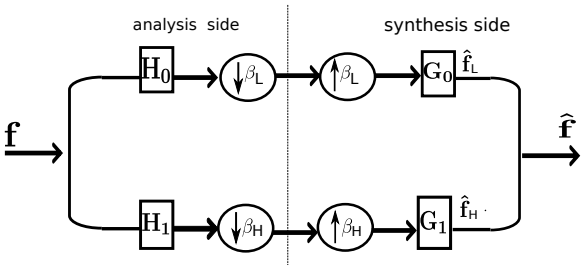


Figure 2.1: Block diagram of a two-channel wavelet filterbank on graph.

signals in the lowpass and highpass channels after reconstruction are given as

$$\begin{aligned}\hat{\mathbf{f}}_L &= \frac{1}{2}\mathbf{G}_0(\mathbf{I}+\mathbf{J}_{\beta_L})\mathbf{H}_0\mathbf{f} \\ &= \frac{1}{2}\mathbf{G}_0(\mathbf{H}_0\mathbf{f}+\mathbf{J}_{\beta_L}\mathbf{H}_0\mathbf{f}),\end{aligned} \tag{2.12}$$

and

$$\begin{aligned}\hat{\mathbf{f}}_H &= \frac{1}{2}\mathbf{G}_1(\mathbf{I}+\mathbf{J}_{\beta_H})\mathbf{H}_1\mathbf{f} \\ &= \frac{1}{2}\mathbf{G}_1(\mathbf{H}_1\mathbf{f}+\mathbf{J}_{\beta_H}\mathbf{H}_1\mathbf{f}),\end{aligned} \tag{2.13}$$

respectively. Thus, $\hat{\mathbf{f}}_L$ is the sum of product of $\mathbf{G}_0$ with signal $\mathbf{H}_0\mathbf{f}$ and a modulated signal $\mathbf{J}_{\beta_L}\mathbf{H}_0\mathbf{f}$. Similarly, $\hat{\mathbf{f}}_H$ is the product of $\mathbf{G}_1$ with signal $\mathbf{H}_1\mathbf{f}$ and a modulated signal $\mathbf{J}_{\beta_H}\mathbf{H}_1\mathbf{f}$. Note that without the DU operations, the output of two channels are simply $\hat{\mathbf{f}}_L = \mathbf{G}_0\mathbf{H}_0\mathbf{f}$ and $\hat{\mathbf{f}}_H = \mathbf{G}_1\mathbf{H}_1\mathbf{f}$, respectively. Thus the modulated signals $\mathbf{G}_0\mathbf{J}_{\beta_L}\mathbf{H}_0\mathbf{f}$ and $\mathbf{G}_1\mathbf{J}_{\beta_H}\mathbf{H}_1\mathbf{f}$ in (2.12) and (2.13), respectively, can be interpreted as producing distortion (or aliasing) in the two channels. The overall output $\hat{\mathbf{f}}$ of the filterbank is the sum of outputs of the two channels, i.e., $\hat{\mathbf{f}} = \hat{\mathbf{f}}_L + \hat{\mathbf{f}}_H = \mathbf{T}\mathbf{f}$, where $\mathbf{T}$ is the overall transfer function of the filterbank. Combining (2.12) and (2.13) $\hat{\mathbf{f}}$ can be written as:

$$\hat{\mathbf{f}} = \frac{1}{2}\mathbf{G}_0(\mathbf{I}+\mathbf{J}_{\beta_L})\mathbf{H}_0\mathbf{f} + \frac{1}{2}\mathbf{G}_1(\mathbf{I}+\mathbf{J}_{\beta_H})\mathbf{H}_1\mathbf{f}. \tag{2.14}$$

Separating out modulation terms in (2.14), we get

$$\hat{\mathbf{f}} = \underbrace{\frac{1}{2}(\mathbf{G}_0\mathbf{H}_0+\mathbf{G}_1\mathbf{H}_1)}_{\mathbf{T}_{eq}}\mathbf{f} + \underbrace{\frac{1}{2}(\mathbf{G}_0\mathbf{J}_{\beta_L}\mathbf{H}_0+\mathbf{G}_1\mathbf{J}_{\beta_H}\mathbf{H}_1)}_{\mathbf{T}_{alias}}\mathbf{f}. \tag{2.15}$$

where $\mathbf{T}_{eq}$ is the transfer function of the filterbank without the *DU* operation and $\mathbf{T}_{alias}$ is another transform which arises primarily due to the downsampling in the two channels. For perfect reconstruction $\mathbf{T}$ should be equal to identity which can be ensured by requiring $\mathbf{T}_{eq}$ to be a scalar multiple of identity and $\mathbf{T}_{alias} = \mathbf{0}$. *Thus the two-channel filterbank on a graph provides distortion-free perfect reconstruction if*

$$\begin{aligned}\mathbf{G}_0\mathbf{J}_{\beta_L}\mathbf{H}_0 + \mathbf{G}_1\mathbf{J}_{\beta_H}\mathbf{H}_1 &= \mathbf{0} \\ \mathbf{G}_0\mathbf{H}_0 + \mathbf{G}_1\mathbf{H}_1 &= c\mathbf{I}\end{aligned} \tag{2.16}$$

In order to design perfect reconstruction filterbanks we need to determine a) how to design filtering operations $\mathbf{H}_i, \mathbf{G}_i, i = \{0,1\}$, and b) the downsampling functions β_L and β_H. In Chapter

4, we show that the spectral folding phenomenon in bipartite graphs leads to an aliasing interpretation of (2.16) and we design filterbanks which cancel aliasing and lead to perfect reconstruction of any graph-signal.

2.5 Literature Review

There has been some work in the past few years on developing localized transforms for data defined on graphs. While some of these works do not take into account the "filterbank perspective", our goal in this section is to place these designs in the common framework described so far in this chapter. We broadly divide the existing graph-transform designs into two types, namely, spatial and spectral designs. We first describe spatial wavelet transform designs which are based on the spatial features of the graph, i.e., in terms of node connectivity and distances between nodes. Next, we describe spectral wavelet transforms on graphs which are based on the spectral features of the graph, i.e. in terms of the eigenvalues and eigenvectors of a matrix defined on the graph. In order to understand these designs we introduce some additional notation. We define $\partial\mathcal{N}_{h,k}$ to be an h-hop neighborhood ring around node k (i.e., the set of all nodes v such that the shortest hop distance between v and k is exactly equal to h), a j-hop adjacency matrix $\mathbf{A}_j$ s.t. $\mathbf{A}_j(n,m) = 1$ only if $m \in \mathcal{N}_{j,n}$, a j-hop diagonal degree matrix with $D_j(k,k) = |\mathcal{N}_{j,k}|$ s.t. $d_{j,k} = |\mathcal{N}_{j,k}|$ and a j-hop uniform Laplacian matrix $\mathbf{L}_j = \mathbf{D}_j - \mathbf{A}_j$. Similarly we define a ring adjacency matrix $\partial\mathbf{A}_h$ such that $\partial\mathbf{A}_j(n,m) = 1$ only if $m \in \partial\mathcal{N}_{j,n}$ and corresponding ring degree matrix $\partial\mathbf{D}_h = diag\{\partial d_{j,k}\}$ s.t. $\partial d_{j,k} = |\partial\mathcal{N}_{h,k}|$.

2.5.1 Spatial Designs

2.5.1.1 Random transforms

Wang and Ramchandran [47] proposed spatially localized graph transforms for sensor network graphs with binary links (i.e., links which have weight either 0 or 1). The transforms proposed in [47] either compute a weighted average given as

$$y(n) = (1 - a + \frac{a}{d_{j,k}+1})x(n) + \sum_{m \in \mathcal{N}_{j,n}} \frac{a}{d_{j,k}+1} x(m), \tag{2.17}$$

or a weighted difference given as

$$y(n) = (1 + b - \frac{b}{d_{j,k}+1})x(n) - \sum_{m \in \mathcal{N}_{j,n}} \frac{b}{d_{j,k}+1} x(m), \tag{2.18}$$

in a j-hop neighborhood around each node in the graph. The corresponding transform matrices can be represented for a given j as

$$\begin{aligned} \mathbf{T}_j &= \mathbf{I} - a(\mathbf{I} + \mathbf{D}_j)^{-1}\mathbf{L}_j \\ \mathbf{S}_j &= \mathbf{I} + b(\mathbf{I} + \mathbf{D}_j)^{-1}\mathbf{L}_j. \end{aligned} \tag{2.19}$$

This approach intuitively defines a two-channel wavelet filter-bank on the graph consisting of two types of linear filters: a) *approximation filters* as given in (2.17) and b) *detail filters* as given in (2.18). Note that these transforms are oversampled and produce output of the size twice that of the input, since no downsampling is applied after filtering. Further none of the transforms can be called a wavelet filter since both transforms have a non-zero DC response.

2.5.1.2 Graph wavelets

Crovella and Kolaczyk [7] designed wavelet like transforms on graphs that are localized in space. They defined a collection of functions $\psi_{j,n} : \mathcal{V} \to \mathbb{R}$, localized with respect to a range of scale/location indices (j, n), which at a minimum satisfy $\sum_{m \in \mathcal{V}} \psi_{j,n}(m) = 0$ (i.e. a zero DC response). Each function $\psi_{j,n}$ is constant within hop rings $\partial\mathcal{N}_{h,n}$ and can be written as:

$$y(n) = a_{j,0}x(n) + \sum_{h=1}^{j} \sum_{m \in \mathcal{N}_{h,n}} \frac{a_{j,h}}{\partial d_{j,n}} x(m) \tag{2.20}$$

In matrix form the j-hop wavelet transform $\mathbf{T}_j$ can be written as:

$$\mathbf{T}_j = a_{j,0}\mathbf{I} + a_{j,1}\partial\mathbf{D}_1^{-1}\partial\mathbf{A}_1 + ...a_{j,j}\partial\mathbf{D}_j^{-1}\partial\mathbf{A}_j \tag{2.21}$$

Further, the constants $a_{j,h}$ satisfy $\sum_{h=0}^{h=j} a_{j,h} = 0$, which allows the wavelet filters to have zero DC response. and can be computed from any continuous wavelet function $\psi(x)$ supported on the interval $[0, 1)$ by taking $a_{j,h}$ to be the average of $\psi(x)$ on the sub-intervals $I_{j,h} = [\frac{h}{j+1}, \frac{h+1}{j+1}]$. Though these transforms are local and provide a multi-scale summarized view of the graph, they

do not have approximation filters and are not invertible in general.

2.5.1.3 Lifting wavelet transforms

Lifting based wavelet transforms have been proposed for graphs with a Euclidean embedding in [45], and for arbitrary routing trees in [38]. In [18], lifting transform is designed in an iterative way by *lifting one coefficient at a time*. These filterbanks provide a natural way of constructing local two-channel critically sampled filter-banks on graph-signals. A block-diagram of lifting wavelet filter-bank is shown in Figure 2.2. In this approach the vertex set is first partitioned into

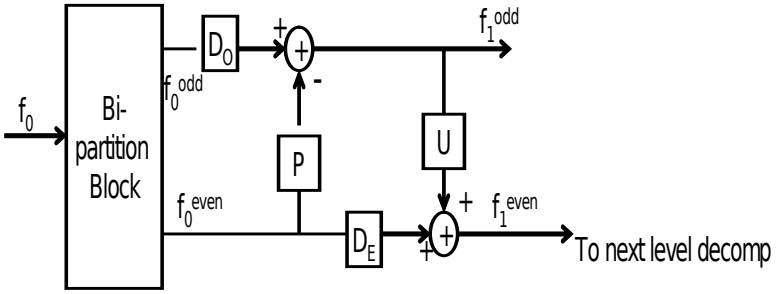


Figure 2.2: Block diagram of two-channel lifting wavelet filter-banks

sets of even and odd nodes $\mathcal{V} = \mathcal{O} \cup \mathcal{E}$. The odd nodes compute their *prediction* coefficients using their own data and data from their even neighbors followed by even nodes computing their *update* coefficients using their own data and prediction coefficient of their neighboring odd nodes. The equivalent transform in matrix-form can be written as:

$$\mathbf{T}^{lift} = \overbrace{\begin{bmatrix} \mathbf{I}_{\mathcal{O}} & 0 \\ \mathbf{U} & \mathbf{D}_{\mathcal{E}} \end{bmatrix}}^{\tilde{\mathbf{U}}} \overbrace{\begin{bmatrix} \mathbf{D}_{\mathcal{O}} & -\mathbf{P} \\ 0 & \mathbf{I}_{\mathcal{E}} \end{bmatrix}}^{\tilde{\mathbf{P}}} \tag{2.22}$$

where $\mathbf{D}_{\mathcal{O}}$ and $\mathbf{D}_{\mathcal{E}}$ are diagonal matrices of size $|\mathcal{O}|$ and $|\mathcal{E}|$ respectively. Thus, the lifting transforms are critically sampled by design. However, the partitioning schemes for these lifting transforms required either the coordinates of the nodes in some Euclidean embedding (eg. in [45]), or specific structure of the graph (eq., trees in [38, 40]). In Chapter 3, we formulate the problem of partitioning nodes as a bipartite subgraph approximation problem, and provide greedy heuristics to compute optimal subgraphs.

2.5.2 Spectral Designs

2.5.2.1 Diffusion wavelets

Maggioni and Coifman [6] introduced diffusion wavelets, a general theory for wavelet decompositions based on compressed representations of powers of a diffusion operator. Their construction interacts with the underlying graph or manifold space through repeated applications of a diffusion operator $\mathbf{T}$, such as the graph Laplacian matrix $\mathbf{L}$. The localized basis functions at each resolution level are orthogonalized and downsampled appropriately to transform sets of orthonormal basis functions through a variation of the Gram-Schmidt orthonormalization (GSM) scheme. Although this local GSM method orthogonalizes the basis functions (filters) into well localized 'bump-functions' in the spatial domain, it does not provide guarantees on the size of the support of the filters it constructs. Further the diffusion wavelets form an over-complete basis and there is no simple way of representing the corresponding transform $\mathbf{T}$.

2.5.2.2 Spectral graph wavelets

Hammond et al [14] defined spectral graph wavelet transforms that are determined by the choice of a kernel function $g : \mathbb{R}^+ \to \mathbb{R}^+$. The kernel $g(\lambda)$, is a continuous bandpass function in spectral domain with $g(0) = 0$ and $\lim_{\lambda \to \lambda_{max}} g(\lambda) = 0$, where λ_{max} is the highest magnitude eigenvalue of the Laplacian matrix $\mathbf{L}$ (or $\mathcal{L}$). The corresponding wavelet operator $\mathbf{T}_g = g(\mathcal{L}) = \mathbf{U} g(\Lambda) \mathbf{U}^t$ acts on a graph signal $\mathbf{f}$ by modulating each Fourier mode as

$$\mathbf{T}_g \mathbf{f} = \sum_{k=1}^{N} g(\lambda_k) \bar{f}(k) \mathbf{u}_k \tag{2.23}$$

The kernel can be scaled as $g(t\lambda)$ by a continuous scalar t. For spatial localization, the authors design filters by approximating the kernels $g(\lambda)$ with smooth polynomials functions. The approximate transform with polynomial kernel of degree k is given by $\mathbf{T}_{\hat{g}} = \hat{g}(\mathbf{L}) = \sum_{l=0}^{k} a_l \mathbf{L}^l$ and is exactly k-hop localized in space. By construction the spectral wavelet transforms have zero DC response, hence in order to stably represent the low frequency content of signal $\mathbf{f}$ a second class of kernel function $h : \mathbb{R}^+ \to \mathbb{R}^+$ is introduced which acts as a lowpass filter, and satisfies $h(0) > 0$ and $\lim_{\lambda \to \lambda_{max}} h(\lambda) = 0$. Thus a multi-channel wavelet transform can be constructed from the choice of a low pass kernel $h(\lambda)$ and J band-pass kernels $\{g(t_1\lambda), ..., g(t_J\lambda)\}$ and it

has been shown that the perfect reconstruction of the original signal is assured if the quantity $G(\lambda) = h(\lambda)^2 + \sum_{k=1}^{J} g(t_i\lambda)^2 > 0$ for all eigenvalues λ in the spectrum of $\mathbf{L}$. However, these transforms are overcomplete, for example, a J-scale decomposition of graph-signal of size N produces $(J+1)N$ transform coefficients. As a result, the transform is invertible only by the least square projection of the output signal onto a lower dimension subspace.

2.6 Summary

In this chapter, we formally introduced graph signals, graph transforms and their spectral domain representation. Further, we introduced downsampling-upsampling (DU) operations on the graphs. These concepts provide a framework for the design of critically sampled two-channel filterbanks on the graphs, and we stated the necessary conditions for these filterbanks to provide perfect reconstruction of any graph signal. Further, we analyzed and evaluated some of the existing graph based transforms, by representing them using the framework introduced in Chapter 2. A common drawback with most of the existing transforms, is oversampling, i.e., number of output wavelet coefficients generated are more than the number of input coefficients. The lifting wavelet transforms are exceptions to this, as they are critically sampled by construction. However, existing lifting based transforms require graph simplification (i.e., approximation of graph to a bipartite graph), which results in the loss of graph properties. In Chapters 5, we will describe two filterbank designs, namely *graph-QMF* filterbanks, and *graph-Bior* filterbank, respectively which are critically sampled and do not require any graph-simplifications. To conclude this chapter, Table 2.1 presents a summary of existing methods and their properties.

[4]When designed using asymmetric normalized Laplacian matrix.

[5]The exact Graph-QMF solutions are perfect reconstruction and orthogonal, but they are not compact support. Localization is achieved with a matrix polynomial approximation of the original filters, which incurs some loss of orthogonality and reconstruction error. This error can be reduced to arbitrary small levels by increasing the degree of approximation.

Method	DC response	CS	PR	Comp	OE	GS
Wang & Ramchandran [47]	non-zero	No	Yes	Yes	No	No
Crovella & Kolaczyk [7]	zero	No	No	Yes	No	No
Lifting Scheme [18, 38, 45]	zero for wavelet basis	Yes	Yes	Yes	No	Yes
Diffusion Wavelets [6]	zero for wavelet basis	No	Yes	Yes	Yes	No
Spectral Wavelets [14]	zero for wavelet basis	No	Yes	Yes	No	No
graph-QMF filterbanks (Sec: 5.3)	zero for wavelet basis[4]	Yes	Yes	No[5]	Yes	No
graph-Bior filterbanks (Sec: 5.5)	zero for wavelet basis[4]	Yes	Yes	Yes	No	No

Table 2.1: Evaluation of graph wavelet transforms. CS: Critical Sampling, PR: Perfect Reconstruction, Comp: compact support, OE: Orthogonal Expansion, GS: Requires Graph Simplification.

Chapter 3

Lifting wavelet filterbanks on graphs

In this chapter, we describe the construction of two-channel lifting wavelet transforms on the vertices of a graph.[1] Lifting wavelet transforms, introduced earlier in Section 2.5, are comprised of three steps, namely, splitting step, prediction step and update step. In the context of graphs, the splitting step corresponds to splitting (labeling) the nodes in the graph into two sets, traditionally called *even set* and *odd set*, respectively. Then in the prediction step, odd set of nodes compute *detail coefficients* using data from their neighboring even nodes, and subsequently in the update step, the even nodes compute *update coefficients* using detail coefficients from their neighboring odd nodes. The overall lifting transform, written in matrix form in (2.22), is critically sampled and invertible by design. A block-diagram of lifting wavelet filter-bank is shown in Figure 2.2.

There are two important choices involved in designing lifting filterbanks: a) how to compute even-odd assignment of the nodes (i.e., splitting step), and b) how to design prediction and update filters. For the latter, there exist many choices. The prediction filters, for example, can be designed in a variety of ways, such as simple average filters [38], filters providing planar approximation [45], filters based on spectral properties [29], or data-adaptive filters [36], etc. Similarly, the update filters can be designed as simple smoothing filters [38], or filters providing orthogonality between update and detail coefficients [39], etc. However, the choice in (a), i.e, the problem of choosing even-odd assignment of the nodes, is relatively less studied in literature. While any even-odd assignment strategy will guarantee invertibility of the resulting filterbank, it is not clear what is an optimal split on the graph. An architecture for lifting wavelet analysis is introduced in [45] for irregular grids in 2-D or 3-D Euclidean spaces. However, the even-odd assignment strategy

[1] Parts of this research are jointly conducted with Dr. Godwin Shen, and J. Perez-Trufero. See [31], and [32] for details.

used there, requires location information of the nodes, and hence cannot be applied to general graphs. In [18], Jansen et al., introduce lifting wavelet transforms on general graphs based on the "lifting one coefficient at a time" theme. In this scheme a lifting filterbank is implemented iteratively in N stages (N is the number of nodes in the graph), such that in each stage only one node is assigned an odd parity, while the remaining nodes are all declared even nodes. Thus, the algorithm produces only one wavelet (i.e., detail) coefficient at each scale, and this can be very slow in decomposing graphs with a large number of nodes. Recently, Shen et al. [38] proposed lifting transforms on trees, in which even-odd assignment at a node depends on the shortest hop distance of the node from the root node. Their assignment strategy defines roughly 50% of nodes as odd nodes, at each scale.

The starting point for this work is the observation, that the idea in [38] can be extended to arbitrary graphs, no longer constrained to be planar and acyclic, as long as suitable even/odd assignment algorithms on the graph can be identified. Our main contribution in this chapter is that we formulate the even-odd assignment problem on the graph as a *bipartite subgraph approximation* problem,which is to approximate the original graph as a single bipartite subgraph, or as a collection of bipartite subgraphs. Each of these subgraphs is defined on the original set of vertices and a subset of edges.. This formulation helps us define optimal even/odd assignment strategies for various applications. The outline of the rest of the chapter is as follows: in Section 3.1 we formulate the problem of even-odd assignment in lifting wavelet transforms on graphs as a bipartite subgraph approximation. In Section 3.2, we propose an even-odd assignment based on maximum bipartite subgraph approximation in the original graph, and propose a greedy heuristic based algorithm to obtain such an assignment. This approach is appropriate for applications where we want to minimize the loss in the quality of lifting filters due to bipartite subgraph approximation, and we discuss a graph-denoising example where these lifting transforms are useful. In Section 3.3, we propose an even-odd assignment based on finding a bipartite subgraph with a dominating set as one of its natural partitions. This strategy is appropriate in compression applications, where the data stored on the even nodes require more bits for storage or transmission than the data on odd nodes, and therefore it is desired to have the minimum number of even nodes in the network. We use it to implement lifting transforms in a data gathering application in wireless sensor networks (WSN), and show performance gains. Finally, we summarize the chapter in Section 3.4.

3.1 Problem Formulation

The starting point of this work is the design of a unidirectional 2D lifting transform along arbitrary trees in a wireless sensor network application, proposed by Shen and Ortega [38]. Given a tree graph, the authors split the nodes into even and odd nodes based on their minimum hopping distance from the root node (see the tree defined by solid lines in Figure 3.1 as an example). A lifting transform is then applied locally on the tree using these assignments. Since trees are acyclic planar graphs, the even-odd assignment of nodes is well-defined and no pair of directly connected nodes is assigned identical (even/odd) parity. To apply this idea to arbitrary graphs (in general cyclic and non-planar) would require selecting an even-odd assignment on these graphs. Referring

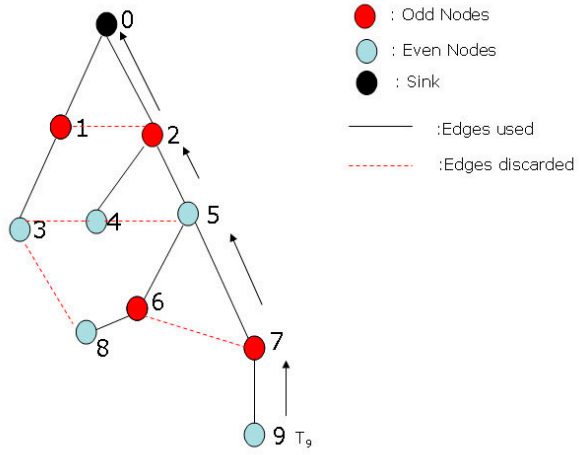


Figure 3.1: Even Odd Assignment in routing trees designed in [38].The dashed lines show the edges not used by the transform though they are within radio-range

again to Figure 3.1 if we now consider a graph that includes both solid and dashed lines it can be seen that nodes that are neighbors in the graph are no longer guaranteed to have opposite parity (e.g., 4 is even and connected to 3 and 5 which are both even as well).Let $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{U}}$ be the transform matrices in the prediction and update step as defined in (2.22) and let $\mathcal{E}$ denote the set of even nodes (blue nodes), and $\mathcal{O}$ denote the set of odd nodes (red nodes). Note that $\mathcal{E}$ and $\mathcal{O}$ are disjoint sets and $\mathcal{E} \cup \mathcal{O} = \mathcal{V}$. Given this even-odd assignment of nodes in the graph, the lifting transform is implemented as follows: we define $\mathbf{f}_{\mathcal{O}}$ and $\mathbf{f}_{\mathcal{E}}$ as the components of input signal on the sets $\mathcal{O}$, and $\mathcal{E}$, respectively, $\tilde{\mathbf{P}}_{\mathcal{O},\mathcal{E}}$ as the submatrix of $\tilde{\mathbf{P}}$ containing prediction weights from

nodes in $\mathcal{O}$ to nodes in $\mathcal{E}$, and $\tilde{\mathbf{U}}_{\mathcal{E},\mathcal{O}}$ as the submatrix of $\tilde{\mathbf{U}}$ containing update weights from nodes in $\mathcal{E}$ to nodes in $\mathcal{O}$. The forward lifting transform is then given as:

$$\begin{aligned} \mathbf{d}_{\mathcal{O}} &= \mathbf{f}_{\mathcal{O}} - \tilde{\mathbf{P}}_{\mathcal{O},\mathcal{E}}\mathbf{f}_{\mathcal{E}} \\ \mathbf{s}_{\mathcal{E}} &= \mathbf{f}_{\mathcal{E}} + \tilde{\mathbf{U}}_{\mathcal{E},\mathcal{O}}\mathbf{d}_{\mathcal{O}}. \end{aligned} \tag{3.1}$$

This transform is invertible and the original values can be recovered by following inverse lifting steps given as:

$$\begin{aligned} \mathbf{f}_{\mathcal{E}} &= \mathbf{s}_{\mathcal{E}} - \tilde{\mathbf{U}}_{\mathcal{E},\mathcal{O}}\mathbf{d}_{\mathcal{O}} \\ \mathbf{f}_{\mathcal{O}} &= \mathbf{d}_{\mathcal{O}} + \tilde{\mathbf{P}}_{\mathcal{O},\mathcal{E}}\mathbf{f}_{\mathcal{E}}. \end{aligned} \tag{3.2}$$

In some applications when we may want to have over-sampled transforms on the graph, we compute one more lifting transform, with the parity of even and odd nodes swapped. In this case, each node has one detail coefficient and one update coefficient value.

We observe that in the prediction step of lifting, odd nodes only use their even neighbors' data to compute prediction coefficients. Similarly, in the subsequent update step, even nodes only use their odd neighbors' data to compute their update coefficients. Thus, nodes of the same parity (even/odd), do not use each other's data, even if they are neighbors in the graph. In other words, links between any two even nodes or any two odd nodes, do not participate in computing the lifting transform, and can be considered non-existent for the purpose of implementing the filterbank. Therefore, the even-odd assignment problem can be formulated as a *bipartite subgraph approximation* problem. A bipartite graph $\mathcal{B} = (L, H, E)$ contains two natural clusters L and H, such that all the links connect nodes in L to nodes in H, and vice versa. The bipartite graphs are also called *two-colorable graphs*, since they have no conflicting edges. **Thus, given a graph $G = (\mathcal{V}, E)$, and an even-odd assignment which splits the vertex set $\mathcal{V}$ into an even set $\mathcal{E}$ and an odd set $\mathcal{O}$, the graph which is actually used to compute prediction and update filters is a bipartite graph $\mathcal{B} = (\mathcal{E}, \mathcal{O}, \hat{E})$, where $\hat{E} \subset E$ is the subset of all those edges, which connect an even node with an odd node.**

Note that this formulation results in edge losses, since the edges in set $E - \hat{E}$ do not participate in computing the transform. An alternative to this approach is to decompose the graph iteratively

into multiple bipartite subgraphs (say K), and implement the lifting transform in K stages, restricting the splitting and filtering operations in each stage to only one bipartite graph. The details of bipartite subgraph decomposition of a graph are presented in Chapter 6. In the present chapter, we only focus on optimizing one stage of lifting transforms (i.e., by approximating the original graph as a single bipartite graph). The optimality criteria for the bipartite approximation depends on the application. In the next section, we discuss optimality in terms of the quality of the lifting filters.

3.2 Maximum Bipartite Subgraph Approximation

Assume an even-odd assignment $\{\mathcal{E}, \mathcal{O}\}$, which assigns an even or an odd parity to each vertex of a graph $G = (\mathcal{V}, E)$ of size $|\mathcal{V}| = N$. Given this assignment the adjacency matrix $\mathbf{A}$ of G can be written as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{\mathcal{O},\mathcal{O}} & \mathbf{A}_{\mathcal{O},\mathcal{E}} \\ \mathbf{A}_{\mathcal{E},\mathcal{O}} & \mathbf{A}_{\mathcal{E},\mathcal{E}} \end{bmatrix} \tag{3.3}$$

where the submatrix $\mathbf{A}_{\mathcal{O},\mathcal{O}}$ of $\mathbf{A}$ is adjacency matrix of a subgraph containing odd nodes only. Similarly $\mathbf{A}_{\mathcal{E},\mathcal{E}}$ is a submatrix of a subgraph having even nodes only. These matrices contain edges which have conflicts since they connect nodes of same parity. The block matrices $\mathbf{A}_{\mathcal{E},\mathcal{O}}$ and $\mathbf{A}_{\mathcal{O},\mathcal{E}}$ contain edges which do not have conflicts. A lifting transform based on this even-odd assignment utilizes only the $\mathbf{A}_{\mathcal{E},\mathcal{O}}$ and $\mathbf{A}_{\mathcal{O},\mathcal{E}}$ matrices of adjacency matrix, in which case the adjacency matrix, actually used in computing filters is given as:

$$\hat{\mathbf{A}} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_{\mathcal{O},\mathcal{E}} \\ \mathbf{A}_{\mathcal{E},\mathcal{O}} & \mathbf{0} \end{bmatrix} \tag{3.4}$$

and corresponds to a bipartite subgraph $\mathcal{B} = (\mathcal{E}, \mathcal{O}, \hat{E})$, as explained in Section 3.1. This bipartite subgraph approximation affects the quality of the prediction and update filters. Therefore, we define a metric that measures the loss in the quality of lifting filters due to the bipartite subgraph decomposition, and choose optimization criteria to minimize this loss metric. In order to further expand upon this, we would need to choose a specific design of prediction and update filters. Let us choose prediction and update filters based on simple averages as defined in [38]. Using this filter-design, the *best* quality prediction filter at node i is achieved, when i can use data from *all*

of its neighbors. As a result, the "best" case weight applied to node j by the prediction filter at node i is given as:

$$w_p(i,j) = \begin{cases} 1 & \text{if } i = j \\ -\frac{A(i,j)}{D(i,i)} & \text{if } i \neq j \end{cases} \tag{3.5}$$

where $D(i,i)$ is the degree of node i ($D(i,i) = 1$ for isolated nodes). The operations in (3.5) can be written in matrix form as:

$$\mathbf{w}_p \quad = \quad \mathbf{I} - \mathbf{D}^{-1}\mathbf{A} \tag{3.6}$$

However, because of bipartite subgraph approximation some neighbors of each node can not be used in computing filters, and the the "approximate" case weight applied to node j by the filter node i, in the prediction step is given as:

$$\hat{w}_p(i,j) = \begin{cases} 1 & \text{if } i = j \\ -\frac{\hat{A}(i,j)}{\hat{D}(i,i)} & \text{if } i \neq j \end{cases} \tag{3.7}$$

and in the matrix form as:

$$\hat{\mathbf{w}}_p \quad = \quad \mathbf{I} - \hat{\mathbf{D}}^{-1}\hat{\mathbf{A}} \tag{3.8}$$

Therefore, an optimal bipartite subgraph decomposition, in this case, is the one that minimizes the difference between the best case filters and the approximate filters. Using (3.6) and (3.8), we get:

$$||\mathbf{w}_p - \hat{\mathbf{w}}_p||_1 \quad = \quad ||\mathbf{D}^{-1}\mathbf{A} - \hat{\mathbf{D}}^{-1}\hat{\mathbf{A}}||_1 = \Delta \tag{3.9}$$

where Δ is the entry-wise 1-norm of the difference between $\mathbf{w}_p$ and $\hat{\mathbf{w}}_p$. The filtering operation in the update step depends on the predict operations in the previous step, and therefore are non-linear. However, to keep the optimization linear, we assume that the prediction coefficients are computed according to the "best" case (i.e, by using data from all neighbors at each node). Thus, we ignore the contribution of predict operations, and focus only on the update step. Then, the update weight applied to the node j, by the update filter centered at node i is equal to

$w_u(i,j) = A(i,j)/(2D(i,i))$ in the "best" case and is equal to $\hat{w}_u(i,j) = \hat{A}(i,j)/(2\hat{D}(i,i))$ in the "approximate" case. Similar to the prediction case, the entry-wise 1-norm of the difference between best case filters and the approximate filters can be written as:

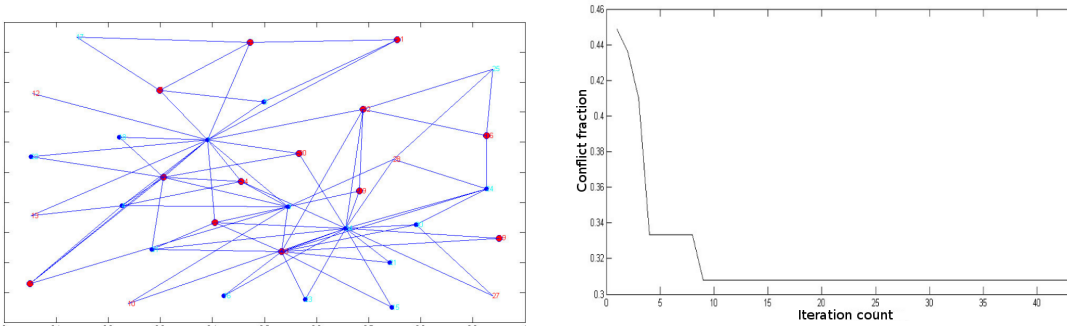
$$\begin{aligned} ||\mathbf{w}_u - \hat{\mathbf{w}}_u||_1 &= \sum_i \sum_j |w_u(i,j) - \hat{w}_u(i,j)| \\ &= \sum_i \sum_j |\frac{A(i,j)}{2D(i,i)} - \frac{\hat{A}(i,j)}{2\hat{D}(i,i)}| \\ &= ||\frac{1}{2}\mathbf{D}^{-1}\mathbf{A} - \frac{1}{2}\hat{\mathbf{D}}^{-1}\hat{\mathbf{A}}||_1 = \frac{1}{2}\Delta \end{aligned} \tag{3.10}$$

Thus, minimizing Δ in (3.9), and (3.10) minimizes the norm of the difference of both prediction and update operations. While this is applicable only for the average prediction and update filters, in general Δ provides a good measurement of loss in the quality of best case and approximate filters. Further, since degree matrix $\mathbf{D}$ itself is derived from $\mathbf{A}$, therefore if $\mathbf{A} \approx \hat{\mathbf{A}}$ then $\hat{\mathbf{D}} \approx \mathbf{D}$ and $\Delta = ||\mathbf{D}^{-1}\mathbf{A} - \hat{\mathbf{D}}^{-1}\hat{\mathbf{A}}||_1 \approx 0$. Thus, in order to minimize Δ, we minimize $\Theta = ||\mathbf{A} - \hat{\mathbf{A}}||_1$, which using (3.3), and (3.4), can be written as:

$$\Theta = ||\mathbf{A} - \hat{\mathbf{A}}||_1 = ||\mathbf{A}_{\mathcal{O},\mathcal{O}}||_1 + ||\mathbf{A}_{\mathcal{E},\mathcal{E}}||_1 \tag{3.11}$$

According to (3.11), Θ is equal to the number of conflicting edges in the graph, i.e., number of edges which have nodes of same parity on both its ends, and minimizing Θ corresponds to an even-odd assignment which minimizes the number of conflicts in the graph. The problem can also be formulated as a max-cut problem [1], for which many good centralized approximations available. However, here we choose a decentralized, synchronous algorithms called conservative fixed probability colorer (CFP) given in [12], which can be computed iteratively by using only 1-hop communications at each step. The algorithm operates in a distributed manner which is more efficient in the case of large graphs. The CFP colorer algorithm solves the corresponding problem of 2 colors graph coloring problem (2-GCP) so as to minimize the conflicts. This algorithm is based on a simple greedy local heuristics and gives competitive results as compared to other k-GCP algorithms [12]. The algorithm starts with each node choosing a parity (even/odd) at random. Then the nodes repeatedly update their colors in synchronized steps. In each step, each node decides whether or not to activate by comparing a randomly generated number with some

activation probability p. If the node activates, then it chooses a color that minimizes the number of conflicting edges that it has with its neighbors based on their parity in the previous step. Those nodes that change color inform their neighbors: all of a node's operations are thus based on information that it has available locally. Figure 3.2(a) shows a sample even-odd assignment for the Karate Data [50] and Figure 3.2(b) shows the reduction of conflicts with each iteration. The convergence of solution has been discussed in [12]. If the solution converges, it ensures



(a) Even-Odd assignment on Graph

(b) convergence of CFP algorithm. The x-axis is number of iterations. The value on y-axis is the fraction of conflicting edges.

Figure 3.2: Even-Odd assignment on Zachary Karate Data [50] using CFP algorithm.

in probability that there are no nodes having more than 50% neighbors of same parity. The algorithm can be easily extended to weighted edge graphs, where we minimize the total weight of the conflicting edges at each node, instead of the total number of edges.

3.2.1 Example: graph denoising

We address the even-odd assignment problem in the design of lifting transforms for a simple graph denoising application. Graph denoising problem refers to denoising data, defined on the vertices of a graph, using connectivity information, and may be applied as a preprocessing tool in analyzing real world graphs, e.g., protein interaction networks [33]. The motivation behind graph denoising is that if the original clean data is smooth, or piecewise smooth on the graph, then the samples at any two nodes, which are connected directly by a link, are correlated. Hence the sample at a node can be predicted using the samples of its 1-hop neighbors, and the prediction error corresponds to the noise as well as transition between two smooth regions. The lifting based wavelet transforms are useful in graph denoising, because the wavelet (detail) coefficients

obtained in the transforms are exactly these prediction errors. Thus, denoising can be performed by transforming the noisy data into the wavelet domain, applying thresholding in the wavelet domain, and inverse transforming the denoised wavelet coefficients. Since both forward and inverse lifting transforms have compact support (1-hop localized support at each node), they can be quite efficient in a distributed denoising application.

In this example, we use prediction and update filters based on simple averages as defined in [38]. The toy graphs of our experiment are similarity graphs (see [26], Section 2.2) with N uniformly sampled nodes from two partially overlapping Gaussian distributions. An edge $\{i, j\}$ between two vertices in the graph exists if the difference in the corresponding sample values is less than some threshold. In order to get optimal quality prediction and update filters, we formulate the even-odd assignment problem as a maximum bipartite subgraph approximation problem, and apply CFP algorithm, described above, to obtain a solution. For thresholding we apply universal threshold given by Donoho [10] $thr = \sqrt{2\log_2(N)}$ on the wavelet coefficients normalized to the noise level [19]. An example graph with $N = 200$ sample values is shown in Figure 3.3(a). Figures 3.3(b)-(f) show Voronoi tessellations of the distribution field with $N = 1500$ sampled points as Voronoi sites. For Figure 3.3(b) the value of each sample is the mean of the distribution from which it is drawn. In Figure 3.3(c) sample values are the actual noisy values. The intensity of each cell reflects the value of corresponding sample in the cell rescaled to the range between $[0, 1]$. Figure 3.3 (d),(e),(f) are the Voronoi tessellations of denoised samples. This problem can be seen as a 2D version of denoising of a general M-dimensional discrete data. While our results are preliminary they demonstrate promising performance as compared to simple, single-step methods operating on the Laplacian matrix that have been proposed in the literature. We compare our results to both short time and long time solutions of the diffusion heat equation ([3, 51]) on the graphs . The Voronoi tessellations of the field constructed from denoised values of the samples are drawn in Figure 3.3(c)-(f). The plots show that lifting transform based denoising results are closer to original distribution in Figure 3.3(b) than diffusion based methods. To quantitatively assess these results we use two quality metrics: peak signal to noise ratio (PSNR) and standard deviation(STD) of samples. Results are in Figure 3.5 and 3.4. As can be seen in Figure 3.5, PSNR achieved in lifting is higher than for diffusion based methods, with better results achieved with the oversampled approach. Note that gains from oversampling are only significant for relatively sparse graphs. In Figure 3.4 we can observe reduction STD with respect to the original signal and,

here too, we observe STD of oversampled transform to be lower than STD in critically sampled case.

3.3 Dominating Set Approximation

In the context of compression schemes which use distributed spatial transforms, the main objective is to compress the data into transform coefficients that require as few bits as possible. Note that in the case of the lifting transforms, even nodes in the network must transmit raw (original) data to their neighbors before any transform computations can take place. Moreover, the update coefficients computed on the even nodes, are also very similar to the original data, and take as many bits as required to encode the original data. Therefore, we also refer to even nodes as *raw data nodes*. Odd nodes then use this even node data to compute detail coefficients, hence, odd nodes can also be called *a*ggregating nodes. Therefore, in this section we shall refer to even-odd assignment as *raw-aggregating node assignment*, or simply RANA. If the sensed data is spatially correlated and a "sufficient" amount of data is received at aggregating nodes, the decorrelated data (i.e., the transform coefficients) at aggregating nodes will generally require significantly fewer bits than those needed to represent raw data. In this case, the objective of even-odd assignment is to minimize the number of even nodes as much as possible, and hence the maximum bipartite subgraph approximation described in Section 3.2, may not be optimal.

Consider an in-network transform where RANA leads to raw nodes in set $\mathcal{E}$ and aggregating nodes in set $\mathcal{O}$. Suppose that we compute the transform in a distributed manner using the method described above. For simplicity, suppose that data from each raw node m is encoded using B_r bits and that the transform coefficient from each aggregating node n is encoded using cB_r bits for some constant $c \in (0, 1]$. This could be the case in fairly dense networks since then each aggregating node is likely to receive the same amount of data from raw nodes. Thus, the amount of compression that can be achieved will be about the same for all aggregating nodes. Further, let $g(n)$ denote the cost of storing or transmitting a single bit at node n. The cost $g(n)$ could, for example, be the cost of transmitting a single bit from n to a single or multiple sinks in the network, or could be inversely proportional to the bandwidth available to node n. Then, the total

cost to compute a given transform in a distributed manner and route the resulting coefficients to the sink is simply

$$C(\mathcal{E}) = \sum_{m\in\mathcal{E}} B_r g(m) + \sum_{n\in\mathcal{O}} cB_r g(n). \tag{3.12}$$

Note that $\sum_{m\in\mathcal{V}} g(m) = \sum_{m\in\mathcal{E}} g(m) + \sum_{n\in\mathcal{O}} g(n)$ since $\mathcal{V} = \mathcal{E} \cup \mathcal{O}$ and $\mathcal{E} \cap \mathcal{O} = \emptyset$. Therefore, (3.12) becomes

$$C(\mathcal{E}) = (1-c)B_r \sum_{m\in\mathcal{E}} g(m) + cB_r \sum_{n\in\mathcal{V}} g(n). \tag{3.13}$$

Since the graph G is fixed, B_r is constant and c is constant, we have that both $(1-c)$ and $c\sum_{n\in\mathcal{V}} B_r g(n)$ are constant. Therefore, the only thing left to optimize is $\sum_{m\in\mathcal{E}} g(m)$, which is nothing more than the sum of the routing costs of the raw nodes. Note that a given assignment of $\mathcal{E}$ and $\mathcal{O}$ only makes sense if every aggregating node n has at least one raw data neighbor from which it can transform its own data, i.e., for all $n \in \mathcal{O}$, $\mathcal{N}_{1,n} \neq \emptyset$. This is equivalent to requiring that every aggregating node be "covered" by at least one raw data node, or more precisely, that $\cup_{m\in\mathcal{E}}\mathcal{N}_{1,m} = \mathcal{V}$. In other words, set $\mathcal{E}$ is a *dominating set*[2] in graph G, and $\cup_{m\in\mathcal{E}}\mathcal{N}_{1,m}$ provides a set cover for the nodes in G. Thus, finding the minimum cost $C(\mathcal{E})$ in (3.13) is equivalent to finding an assignment of raw data and aggregating nodes which minimizes $\sum_{m\in\mathcal{E}} g(m)$ under the constraint that $\cup_{m\in\mathcal{E}}\mathcal{N}_{1,m}$ provides a set cover for G. This can be formulated as a minimum weighted set cover (MWSC) problem described below.

Definition 1. Minimum Weight Set Cover Problem: *For graph $G = (V, E)$ denote closed neighborhood $n_{[v]} = n_{[v]} = \{v\} \cup \{u \in V : (v, u) \in E\}$ as a disk centered at node v with corresponding weight $g(v)$ for all nodes $v \in \mathcal{V}$. Given a collection $\mathcal{N}$ of all sets $\{n_{[v]}\}$, a set-cover $\mathcal{C} \subseteq \mathcal{N}$ is a sub collection of the sets whose union is V. The MWSC problem is to find a set-cover with minimum weights.*

Set-covering problem for unweighted undirected graphs is NP-hard in general. However it can be solved by a natural greedy algorithm that iteratively adds a set that covers the highest number of yet uncovered elements. It provides a good approximation [4] and can be implemented in a distributed way. The algorithm is same for directed graphs with the exception that sets with highest outdegree of central node are added first to the cover. The algorithm for choosing a greedy set cover in a graph is given in Algorithm 1.

[2]A dominating set for a graph $G = (\mathcal{V}, E)$ is a subset $\mathcal{E}$ of $\mathcal{V}$ such that every vertex not in $\mathcal{E}$ is joined to at least one member of $\mathcal{E}$ by some edge.

Algorithm 1 Greedy Minimum Set Cover

Require: $\mathcal{N} = \{n_{[v]}\}_{v \in V}$

1: Initialize $\mathcal{C} = \phi$. Define $f(\mathcal{C}) = |\bigcup_{n_{[v]} \in \mathcal{C}} n_{[v]}|$

2: **repeat**

3: Choose $v_j \in V$ maximizing the difference $[f(\mathcal{C} \cup \{n_{[v_j]}\}) - f(\mathcal{C})]$

4: Let $\mathcal{C} \leftarrow \mathcal{C} \cup \{n_{[v_j]}\}$

5: **until** $f(\mathcal{C}) = f(\mathcal{N})$

6: **return** $\mathcal{C}$

However, these set covering problems do not take into consideration the fact that even if the selected even nodes are small in number, the total cost $\sum_{m \in \mathcal{E}} g(m)$ of even nodes may be very high. To avoid this we propose a minimum weighted set covering problem. In the weighted set-covering problem, for each set $n_{[v]} \in \mathcal{N}$ a weight $w_v \geq 0$ is also specified, and the goal is to find a set cover $\mathcal{C}$ of minimum total weight. In the context of our problem weight $w_v = g(v)$ for node v. The greedy algorithm for weighted set cover builds a cover by repeatedly choosing a a set $n_{[v]} \in \mathcal{N}$ that minimizes the weight w_v divided by number of elements in $n_{[v]}$ not yet covered by chosen sets. The algorithm for choosing a greedy weighted set cover is given in Algorithm 2.

Algorithm 2 Greedy Minimum Weight Set Cover

Require: $\mathcal{N} = \{n_{[v]}\}_{v \in V}, \mathcal{W} = \{w_v\}_{v \in V}$

1: Initialize $\mathcal{C} = \phi$. Define $f(\mathcal{C}) = |\bigcup_{n_{[v]} \in \mathcal{C}} n_{[v]}|$

2: **repeat**

3: Choose $v_j \in V$ minimizing the cost per element $w_{v_j}/[f(\mathcal{C} \cup \{n_{[v_j]}\}) - f(\mathcal{C})]$

4: Let $\mathcal{C} \leftarrow \mathcal{C} \cup \{n_{[v_j]}\}$

5: **until** $f(\mathcal{C}) == f(\mathcal{N})$

6: **return** $\mathcal{C}$

3.3.1 Example: data gathering in WSN

In this section, we discuss an application of lifting transforms in which, we optimize spatial compression in wireless sensor networks (WSN) on arbitrary communication graphs. Since nodes in a wireless sensor network (WSN) are severely energy-constrained devices, it is essential to perform in-network compression for energy-efficient data gathering. The data gathering problem

in a single sink case is described as follows: suppose we have a network of N nodes and a sink node (indexed by $N+1$), and suppose that each node has some data $x(n)$ that it needs to forward to the sink. Assume that each node (indexed by $n \in \mathcal{I} = \{1, 2, \cdots, N\}$) transmits using a radio range of R_n and let $G = (V, E)$ be the directed communication graph which results from these choices of radio ranges. Let $\mathcal{N}_{R_n}(n)$ denote the set of nodes within R_n radio range of node n, i.e., $\mathcal{N}_{R_n}(n)$ is the set of all nodes that can hear transmissions from n. Furthermore, let $T = (V, E')$ denote a routing tree rooted at the sink node and let $g(n)$ denote the cost to route a single bit from node n to the sink along T. The cost $g(n)$ could, for example, be proportional to the number of hops, or to the sum of the squared distances between all nodes along the path from n to the sink, etc. Such a tree can be constructed using standard routing protocols such as the Collection Tree Protocol (CTP) [42], in which case the cost $g(n)$ is simply proportional to the number of hops to the sink. The objective of data-gathering is then to transmit data from all nodes to the sink with minimum transmission cost. For data-gathering application, the lifting transforms implemented on the routing tree T have been shown to perform better than other existing approaches [38]. Here, we extend these lifting transforms to the overall communication graph, using the optimal RANA strategy described above.

For this, we compute even and odd nodes $\mathcal{E}$ and $\mathcal{O}$, respectively, using the greedy set cover algorithms described in Algorithms 1 and 2. Given such sets, let the set of even neighbors that odd node n overhears be given by $\mathcal{H}_n$, i.e., $\mathcal{H}_n = \{m \in \mathcal{E} | n \in \mathcal{N}_{R_m}(m)\}$. Then node n can compute a prediction of its own data using data from nodes in $\mathcal{H}_n$, producing detail coefficient $d(n)$, i.e.,

$$d(n) = x(n) - \sum_{m \in \mathcal{H}_n} \mathbf{a}_n(m) x(m). \tag{3.14}$$

Note that if the prediction $\sum_{m \in \mathcal{H}_n} \mathbf{a}_n(m) x(m)$ is close to $x(n)$, then $d(n)$ will have small magnitude and so can be encoded using fewer bits than that those that would be needed for raw data $x(n)$. This will ultimately lead to cost reduction for odd nodes since they transmit fewer bits per coefficient. An update step can also be computed for data from each even node to produce smooth coefficients, but the number of bits needed to encode smooth coefficients is typically the same as the number of bits needed for raw data. Therefore, we do not use an update step in this work. Distributed computation of this transform proceeds as follows. Since odd nodes predict their data from even neighbors, it is only natural for the even nodes to transmit data first along

T. We assume that these raw data transmissions are broadcasted and that they are forwarded all the way to the sink. In this way, odd nodes can utilize data received from all even neighbors regardless of whether they are required to forward data from them. Odd nodes will then compute detail coefficients, encode them and then transmit them to the sink along T. Since the lifting transform is computed as the data flows towards the sink, it is termed as *unidirectional lifting transform*.

We now compare the unidirectional transforms with graph-based splits presented here against the transform with tree-based split (i.e., a 1-level transform) [38] and an extension of this transform where odd nodes perform additional levels of decomposition on data received from their even children (i.e., a multi-level transform). This multi-level transform is constructed in the same manner as the multi-level transform proposed in [41]. For all transforms, we use the data adaptive prediction filter design in [36]. Figure 3.7 compares the number of raw data transmissions required by a Haar-like lifting scheme and our proposed scheme, for networks of different sizes. It is clear that our proposed method leads to a significant reduction in raw data transmissions. Assuming a nearly uniform deployment of sensors, the distances between nodes are roughly equal. Hence reduction in the number of raw transmissions is directly proportional to the reduction in transmissions costs as shown in Fig. 3.6. Further in Fig. 3.6 the cost of raw data transmissions for weighted set cover based split is lower than the cost of raw data transmissions for unweighted set cover based split. This is to be expected since even nodes selected by weighted set cover algorithms now have lower costs of transmitting data to the sink.

We use an AR-2 model to generate (noise-free) simulation data with high spatial data correlation, e.g., the amount of data correlation between two nodes increases as the distance between them decreases. We also assume that raw measurements use 12 bits. A randomly generated 50 node network and a shortest path routing tree (SPT) is computed and used for routing. The SPT is shown in Fig. 3.8(a) along with the even and odd splitting described in [38]. We use the transmission schedule discussed in [41]. The structure of the graph-based transform presented here is shown in Fig. 3.8(b). Note that although both Figs. 3.8(a) and 3.8(b) have the same underlying routing structure (solid blue lines), the number of required even nodes is smaller for graph-based transform than for tree-based transform. This leads to reduction in raw-data transmission costs.

Performance comparisons are shown in Fig. 3.9, which plots energy consumption versus reconstruction quality (in terms of Signal to Quantization Noise Ratio). Energy consumption is

computed using the cost model in [46]. Each point corresponds to a different quantization level with adaptive arithmetic coding applied to blocks of 50 coefficients at each node. The transform proposed here is the best overall. This is to be expected since it seeks to minimize the number of nodes that must transmit raw data to their neighbors, thereby reducing the total energy consumed in the data gathering process. The 1-level and multi-level transforms with tree-based split do outperform simple raw data gathering, and the multi-level transform does better than the 1-level transform since more de-correlation is achieved in the network. However, both of these methods have roughly 50% raw data nodes, hence, they are not as efficient as the two transforms with graph-based splits (which have roughly 25% raw data nodes).

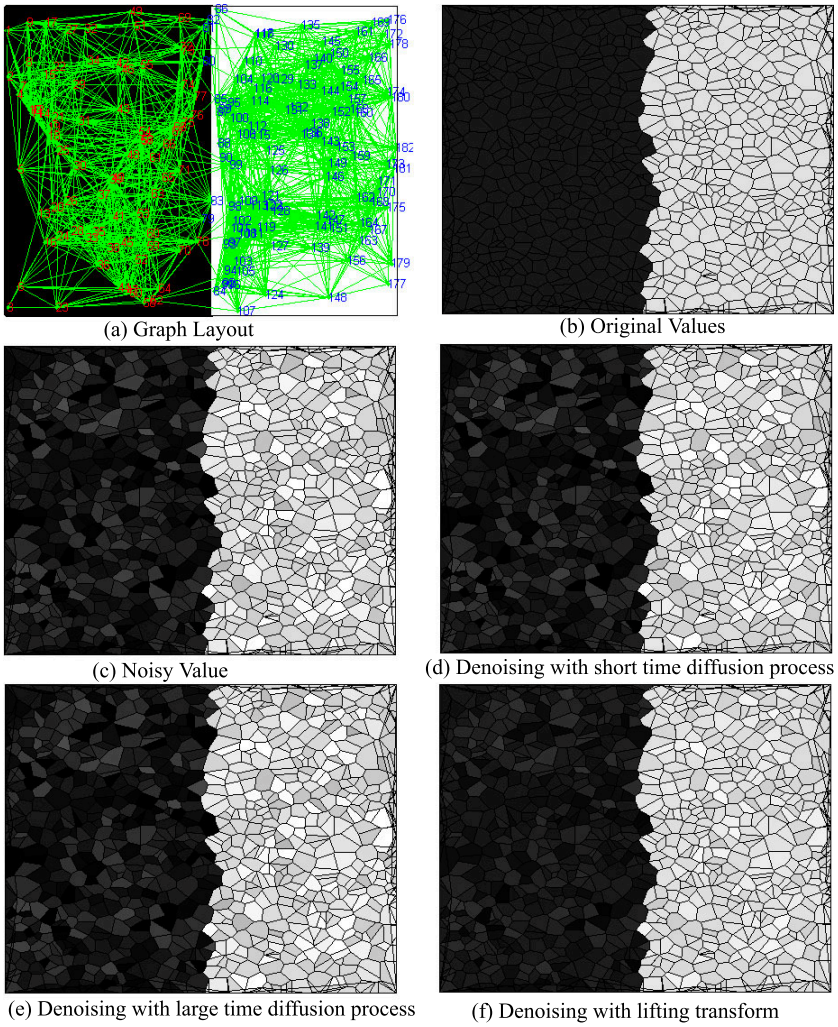
To see this distinction more clearly, consider the lossless coding numbers for this same network shown in Fig. 3.6. The cost for raw data forwarding and the total cost are shown separately. As we can see, the overall performance is greatly affected by the raw data forwarding cost in that lower raw data forwarding leads to lower total cost. In particular, the methods proposed here have the lowest raw data forwarding cost, hence, they also have the lowest overall cost.

Our proposed transform can be easily applied to any arbitrary WSN, since it is computed as the data is routed towards the sink. The schedule of computation and the even-odd assignment of nodes can be pre-fed into sensors at initialization. This transform design can be seen as precursor to a new class of algorithms which would focus on minimizing raw data transmissions in a WSN by jointly optimizing routing tree and even/odd partition (or raw nodes/aggregating nodes partition).

3.4 Summary

In this chapter, we proposed a novel lifting based wavelet transform for graphs. We extended the lifting transforms proposed in [38] for routing trees to any arbitrary graph. For this, we formulated the even-odd assignment problem in these transforms as a bipartite subgraph approximation problem. The definition of optimal bipartite graph depends on the application. In particular, we proposed two solutions: one based on maximum bipartite subgraph approximation and the second based on finding a bipartite graph with a dominating set as one of its natural partitions. For the former, we discussed a toy example of graph denoising, where lifting transform using proposed even-odd assignment are useful. For the latter, we discussed a data-gathering application in WSN, where proposed even-odd assignment leads to low number of raw data transmissions. These lifting

transforms provide a new way of applying signal processing tools on graph based data. In the next chapter, we introduce a theory behind downsampling upsampling operations on graphs. This will lead to the design of spectral wavelet filterbanks on graphs in Chapter 5.



(a) Graph Layout (b) Original Values

(c) Noisy Value (d) Denoising with short time diffusion process

(e) Denoising with large time diffusion process (f) Denoising with lifting transform

Figure 3.3: (a)Similarity graph with 200 sampled points from the underlying distribution.The nodes in shaded region are $\mathcal{N}(\mu_1, \sigma^2)$ and the nodes in white region are $\mathcal{N}(\mu_2, \sigma^2)$ (b)-(f) Voronoi Plots

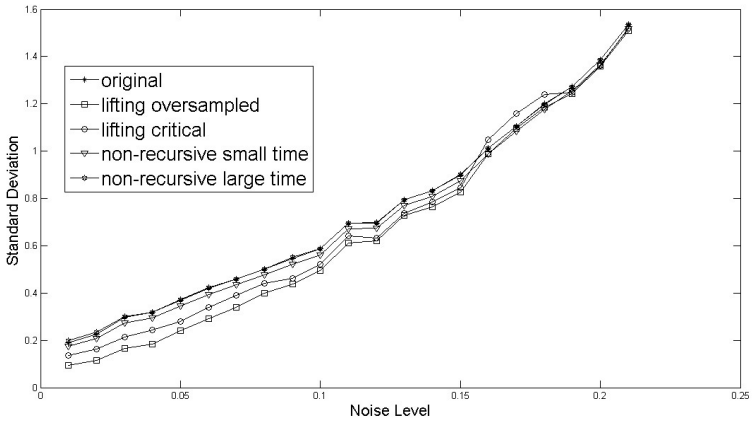


Figure 3.4: STD of the original and denoised samples

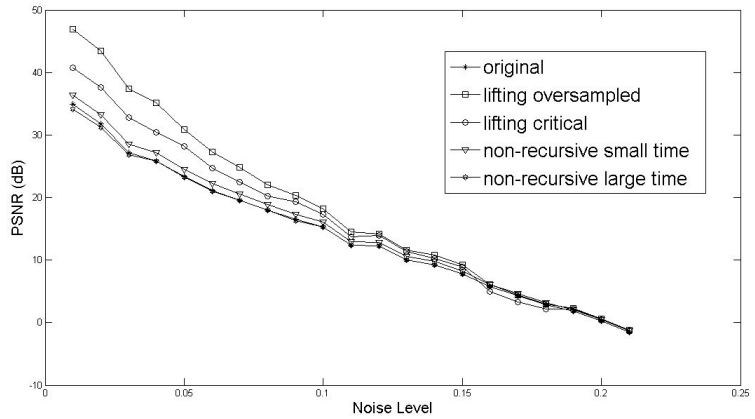


Figure 3.5: PSNR of the original and denoised samples

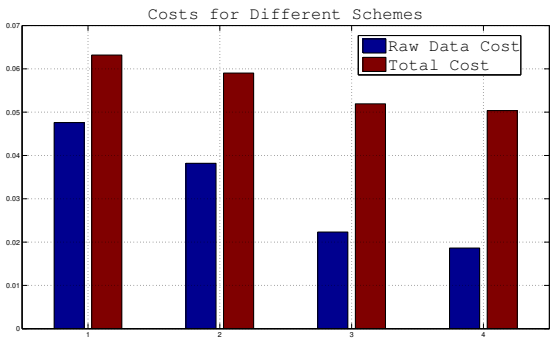


Figure 3.6: Cost Comparison of Different Lifting Schemes (1: Haar-like lifting transform with first level of even/odd split on trees 2: With 3 levels of even/odd split on trees 3: Proposed unweighted set cover based even/odd split on graph 4: Proposed weighted set cover based even/odd split on graph).

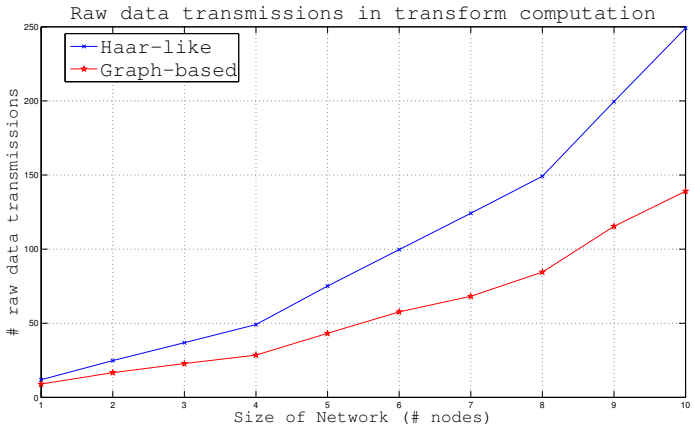
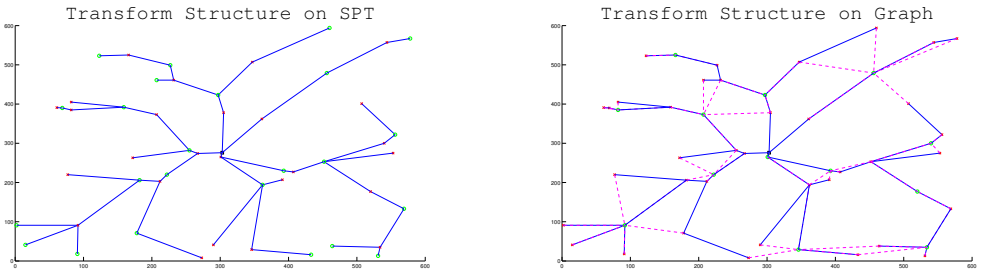


Figure 3.7: Number of raw data transmissions taking place in transform computations of different lifting schemes. The numbers are averages over $Ns = 10$ realizations of each size graphs.



(a) Transform on SPT

(b) Transform on Graph

Figure 3.8: Transform definition on SPT and on graph. Circles denote even nodes and x's denote odd nodes. The sink is shown in the center as a square. Solid lines represent forwarding links. Dashed lines denote broadcast links.

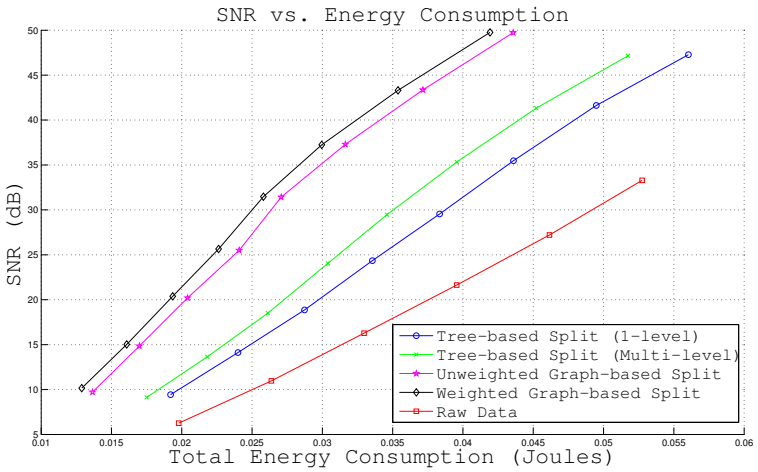


Figure 3.9: Performance comparisons.

Chapter 4

Downsampling in Graphs using Spectral Theory

In traditional signal processing applications, downsampling and upsampling operations are an integral part of multi-rate wavelet filterbanks. An example of downsampling and upsampling in one-dimensional signal $x[n]$ is by a factor of 2, so that in the resulting signal every other sample is zero. The resulting signal $x_{du}[n]$ can be expressed as:

$$x_{du}[n] = \frac{1}{2}(x[n] + (-1)^n x[n]) \tag{4.1}$$

The discrete time Fourier transform of the resulting signal, $x_{du}[n]$, contains the spectrum of the original signal as well as a frequency shifted copy of the original signal, i.e.,

$$X_{du}(e^{j\omega}) = \frac{1}{2}(X(e^{j\omega}) + X(e^{j(\pi-\omega)})) \tag{4.2}$$

which results in *aliasing* if the signal $X(e^{j\omega})$ and shifted copy $X(e^{j(\pi-\omega)})$ have overlapping regions of support. This phenomenon is also termed as *frequency folding*, since the frequency components of the original signal appear to fold across center frequency $\omega = \pi/2$. As described in Chapter 2, the data on graphs can be defined as *graph-signals*, which have a spectral interpretation given by eigenvalues (and eigenvectors) of the graph Laplacian matrix, similar to Fourier transform for regular signals. The focus of this chapter is to propose methods for downsampling graph-signals by extending downsampling results for regular signals to graphs. In particular, we show that for bipartite graphs, the effect of downsampling followed by upsampling can be seen as being analogous to well known aliasing: the spectral representation of the resulting signal is the sum of the spectrum of the original signal and that of a signal obtained by folding the frequency

information around the middle frequency (where we work with a spectral representation based on the graph Laplacian). Although general graphs do not have this form, having a formal approach to analyze downsampling in the bipartite graph case provides a tool to address general graphs, by decomposing them into a series of bipartite graphs. This leads to a "multi-dimensional" decomposition of graphs, in which each "dimension" refers to filtering/downsampling operations restricted to only one bipartite subgraph. The bipartite subgraph decomposition is discussed in Chapter 6. We choose 2-D images as one such example which can be represented as bipartite graphs and for which downsampling/upsampling has a known interpretation. The images can be represented as 4-regular graphs with either rectangular or diamond connectivity. We formulate the problem of downsampling images as bipartizing the underlying graph, and show that common downsampling methods such as rectangular and quincunx sampling can also be understood in terms of spectral properties of these graphs. Then, we design new downsampling methods for images by way of different representations of images into bipartite graphs.

This chapter is organized as follows, in Section 4.1, we formulate the problem of downsampling graphs. In Section 4.2, we discuss downsampling results for *k-regular bipartite graphs* (k-RBG), and in Section 4.3 we extend them to non-regular bipartite graphs. In Section 4.4 we approximate 2-D images as bipartite graphs, and compare the spectral properties of *DU* operations on these bipartite graphs with the standard *DU* operations. Finally, we summarize the findings in this chapter in Section 4.5.

4.1 Problem Formulation

The general formulation of downsampling is described in Section 2.3. In this formulation, we define a downsampling function β_H on the graph $G = (\mathcal{V}, E)$ as choosing a subset $H \subset \mathcal{V}$ such that all samples of the graph signal $\mathbf{f}$ corresponding to indices not in H, are discarded. A subsequent upsampling operation with β_H projects the downsampled signal back to original $\mathbb{R}^N$ space by inserting zeros in place of discarded samples in $L = H^c$. A block diagram of the *DU* operation is given in Figure 4.1.

Referring again to (2.11), the GFT coefficient $\bar{f}_{du}(l)$ of graph signal, after *DU* operation with β_H, consists of GFT coefficient $\bar{f}(l)$ of original signal, and a modulated spectral coefficient $\bar{\mathbf{f}}^d(l)$,

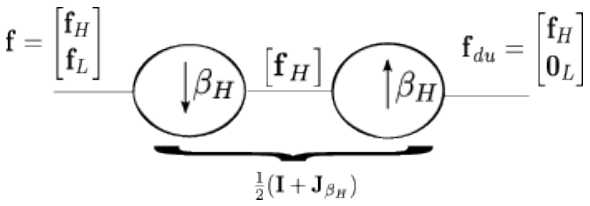


Figure 4.1: Block diagram of *DU* operations in graphs

which can be interpreted as projection of $\mathbf{f}$ onto a modulated basis function $\mathbf{J}_{\beta_H}\mathbf{u}_l$. Similar modulation also occurs during *DU* operations in finite length 1-D signals, which can be represented in terms of their Discrete Fourier Transform (DFT), given as $W_N^k(n) = \exp(-2\pi jkn/N)$. Comparing (2.8) and (4.1), we can write a downsampling function for finite length signals as $\beta_{1D}(n) = 1$ if index n is even and -1 otherwise. The basis functions in this case have the property:

$$\beta_{1D}(n)W_N^k(n) = (-1)^n W_N^k(n) = W_N^{(k-N/2)_N}(n), \tag{4.3}$$

where $W_N^{(k-N/2)_N}$ is the $(k - N/2)$ modulo N basis function. In the matrix form (4.3) can we written as:

$$\mathbf{J}_{\beta_{1D}}\mathbf{W}_N^k = \mathbf{W}_N^{(k-N/2)_N}, \tag{4.4}$$

where $\mathbf{J}_{\beta_{1D}} = diag\{\beta_{1D}(n)\}$ is the downsampling matrix for 1-D signals. Thus, the modulated DFT basis function for finite length signals is also a DFT basis function with a different discrete frequency. This phenomenon is known as *aliasing* in the regular finite length signal domain. We would like to consider conditions for this to be true for graphs as well. This is important as it will allow us to design "anti-aliasing" filters such that the distorted signal $\mathbf{J}_{\beta_H}\mathbf{f}$ will be band-limited and with spectral response disjoint from that of the original signal (after anti-aliasing has been applied). Further, it will also help us design filterbanks which cancel aliasing, and provide *distortion free* reconstruction.

For graph signals, the basis functions are GFT basis, which are eigen-vectors $\{\mathbf{u}_k\}$, $k = 1, 2, ...N$ of the graph Laplacian matrix. Let us first consider regular graphs, which have same degree at each node. For regular graphs, the eigenvectors of Laplacian matrix $\mathbf{L}$ and normalized

Laplacian matrix $\mathcal{L}$ are identical. Similar to finite length signal example we want modulated eigenvector $\mathbf{J}_{\beta_H}\mathbf{u}_k$ to be an eigenvector of the Laplacian $\mathbf{L}$, i.e.

$$\mathbf{L}\mathbf{J}_{\beta_H}\mathbf{u}_k = \hat{\lambda}\mathbf{J}_{\beta_H}\mathbf{u}_k \tag{4.5}$$

This is true if

$$\underbrace{\mathbf{J}_{\beta_H}\mathbf{L}\mathbf{J}_{\beta_H}}_{\hat{\mathbf{L}}}\mathbf{u}_k = \hat{\lambda}\mathbf{J}^2_{\beta_H}\mathbf{u}_k = \hat{\lambda}\mathbf{u}_k \tag{4.6}$$

where matrix $\hat{\mathbf{L}}(i,i) = \mathbf{L}(i,i)\beta_H^2(i) = \mathbf{L}(i,i) = d_i$. Thus

$$\hat{\mathbf{L}} = \mathbf{D} - \underbrace{\mathbf{J}_{\beta_H}\mathbf{A}\mathbf{J}_{\beta_H}}_{\hat{\mathbf{A}}} \tag{4.7}$$

Here $\hat{\mathbf{A}}$ is a modulated adjacency matrix given as:

$$\hat{\mathbf{A}}(i,j) = \begin{cases} \mathbf{A}(i,j) & \text{if } \beta_H(i)\beta_H(j) > 0 \\ -\mathbf{A}(i,j) & otherwise \end{cases} \tag{4.8}$$

Clearly one way in which matrices $\hat{\mathbf{A}}$ and $\mathbf{A}$ will have the same eigen-vectors is if $\hat{\mathbf{A}} = -\mathbf{A}$. This implies that $\mathbf{A}(i,j) = 0$ whenever $\beta_H(i) = \beta_H(j)$, which is only true if the graph is *bipartite* with H and L as its bipartite sets. In the next section, we use this property to describe a spectral folding phenomenon in *k-regular bipartite graphs (k-RBG)* analogous to well known aliasing in the regular signal domain. Subsequently, we extend the results to general bipartite graphs by matrix normalization.

4.2 Downsampling in kRBG graphs

A bipartite graph $\mathcal{B} = (L, H, E)$ is a graph whose vertices can be divided into two disjoint sets L and H, such that every link connects a vertex in L to one in H. Bipartite graphs are also known as *two-colorable graphs* since the vertices can be colored perfectly into two colors so that no two connected vertices are of the same color. A k-regular bipartite graph $\mathcal{B}$ has the same degree k at each of its vertices (i.e. $\mathbf{D} = k\mathbf{I}$). The following known results are useful to understand the spectral properties of *DU* operations in k-RBG:

Lemma 1 ([17, Section 2]). *The following statements are equivalent for any graph G:*

1. *$\mathcal{B}$ is k-RBG with bipartitions H and L.*
2. *$\mathcal{B}$ has an even number of nodes $N = 2n$ with $|L| = |H| = n$ nodes in each partition.*
3. *The spectrum of Laplacian matrix $\mathbf{L}(\mathcal{B})$ is symmetric about k and the minimum and maximum eigenvalues of $\mathbf{L}(\mathcal{B})$ are 0 and $2k$ respectively.*
4. *If $\mathbf{u} = \begin{bmatrix} \mathbf{u}_1^T & \mathbf{u}_2^T \end{bmatrix}^T$ is an eigenvector of $\mathbf{L}(\mathcal{B})$ with eigenvalue λ with $\mathbf{u}_1$ indexed on H and $\mathbf{u}_2$ indexed on L (or vice-versa) then the modulated eigenvector $\hat{\mathbf{u}} = \begin{bmatrix} \mathbf{u}_1^T & -\mathbf{u}_2^T \end{bmatrix}^T$ is also an eigenvector of $\mathbf{L}(\mathcal{B})$ with eigenvalue $2k - \lambda$.*

Thus, given a k-RBG, $\mathcal{B} = (L, H, \mathbf{E})$ a downsampling function can be defined such that $\beta(n) = 1$ if $n \in L$ and $\beta(n) = -1$ if $n \in H$. This downsampling function has the property that $\hat{\mathbf{A}} = \mathbf{J}_\beta \mathbf{A} \mathbf{J}_\beta = -\mathbf{A}$. Thus

$$\begin{aligned} \hat{\mathbf{L}} &= \mathbf{D} + \mathbf{A} = k\mathbf{I} + \mathbf{A} \\ \mathbf{L} &= \mathbf{D} - \mathbf{A} = k\mathbf{I} - \mathbf{A} \end{aligned} \tag{4.9}$$

In this case both matrices $\mathbf{L}$ and $\hat{\mathbf{L}}$ have the same set of eigenvectors. This leads to following proposition:

Proposition 1. *Given a k-regular bipartite graph $\mathcal{B} = (L, H, \mathbf{E})$, let the downsampling function be chosen as $\beta = \beta_H$ or $\beta = \beta_L$, and let $\mathbf{L}$ be the graph Laplacian matrix. If $\mathbf{u}_\lambda$ is an eigenvector of $\mathbf{L}$ of a with eigenvalue λ then modulated eigenvector $\mathbf{J}_\beta \mathbf{u}_\lambda$ is also an eigen-vector of $\mathbf{L}$ with eigenvalue $2k - \lambda$.*

Proof. By definition $\mathbf{J}_\beta \mathbf{L} \mathbf{J}_\beta \mathbf{u}_\lambda = (k\mathbf{I} + \mathbf{A})\mathbf{u}_\lambda = (2k\mathbf{I} - (k\mathbf{I} - \mathbf{A}))\mathbf{u}_\lambda = (2k - \lambda)\mathbf{u}_\lambda$.
$\rightarrow \mathbf{L}\mathbf{J}_\beta \mathbf{u}_\lambda = (2k - \lambda)\mathbf{J}_\beta \mathbf{u}_\lambda$.
Thus $\mathbf{J}_\beta \mathbf{u}_\lambda$ is an eigenvector of $\mathbf{L}$ with eigenvalue $2k - \lambda$. □

Proposition 1 implies that if λ is a unique[1] eigenvalue of $\mathbf{L}$, then

$$\mathbf{J}_\beta \mathbf{u}_\lambda = \pm \mathbf{u}_{2k-\lambda}, \tag{4.10}$$

[1]For eigenvalues with algebraic multiplicity greater than 1, the modulated eigenvector $\mathbf{J}_\beta \mathbf{u}_\lambda$ can be any vector in the V_λ subspace. A general result dealing with non-unique eigenvalues is described in Proposition 2.

where the sign on the right side will change depending upon whether $\beta = \beta_H$ or $\beta = \beta_L$. The result in (4.10) is analogous to aliasing result for finite length signals in (4.3). Substituting this in (2.11) we get for k-regular bipartite graphs:

$$\begin{aligned} \bar{f}_{du}(\lambda) &= \frac{1}{2}(\bar{f}(\lambda) \pm < \mathbf{J}_\beta \mathbf{u}_\lambda \ , \ \mathbf{f} >) \\ &= \frac{1}{2}(\bar{f}(\lambda) \pm < \mathbf{u}_{2k-\lambda} \ , \ \mathbf{f} >) \\ &= \frac{1}{2}(\bar{f}(\lambda) \pm \bar{f}(2k-\lambda)) \end{aligned} \tag{4.11}$$

where $\bar{f}(2k-\lambda)$ is an alias of $\bar{f}(\lambda)$ in the GFT domain. We term this phenomenon *spectral folding* in k-RBG, since the spectrum of any signal after *DU* operations, consists of the original spectrum and the spectrum folded across eigenvalue $\lambda = k$. This phenomenon leads to following Nyquist-like sampling theorem for signals defined on k-RBG:

Theorem 1. *A graph-signal* $\mathbf{f}$ *on a* k*-RBG,* $\mathcal{B} = (L, H, \mathbf{E})$ *can be completely described by only half of its samples in the set* L *or* H *if the spectrum of* $\mathbf{f}$ *is band-limited by* $\lambda = k$*.*

Proof. The spectrum of $\mathbf{f}$, band-limited by $\lambda = k$, implies $\bar{\mathbf{f}}(\lambda) = 0$ for $\lambda \geq k$. Since we keep only samples in subset H, and discard everything else, $\mathbf{f}_{du}$ can be written as $[\mathbf{f}_H^t \ 0^t]^t$, and using (4.11), can be expanded as:

$$\begin{aligned} \mathbf{f}_{du} &= \sum_\lambda \bar{\mathbf{f}}_{du}(\lambda)\mathbf{u}_\lambda \\ &= \frac{1}{2}\sum_\lambda (\bar{\mathbf{f}}(\lambda) + \bar{\mathbf{f}}(2k-\lambda))\mathbf{u}_\lambda \\ &= \frac{1}{2}\sum_\lambda \bar{\mathbf{f}}(\lambda)\mathbf{u}_\lambda + \frac{1}{2}\sum_\lambda \bar{\mathbf{f}}(2k-\lambda)\mathbf{u}_\lambda, \end{aligned} \tag{4.12}$$

where the summation is over all eigenvalues λ. Discarding the terms with $\bar{\mathbf{f}}(\lambda) = 0$ in (4.12), we get:

$$\mathbf{f}_{du} = \frac{1}{2}\sum_{\lambda<k} \bar{\mathbf{f}}(\lambda)\mathbf{u}_\lambda + \frac{1}{2}\sum_{\lambda>k} \bar{\mathbf{f}}(2k-\lambda)\mathbf{u}_\lambda, \tag{4.13}$$

Thus, $\bar{\mathbf{f}}_{du}(\lambda) = 0.5\bar{\mathbf{f}}(\lambda)$ for $\lambda < k$, and $\bar{\mathbf{f}}_{du}(\lambda) = 0.5\bar{\mathbf{f}}(2k-\lambda)$ for $\lambda > k$. In order to recover, $\bar{\mathbf{f}}(\lambda)$ from $\bar{\mathbf{f}}_{du}(\lambda)$, we define an *anti-aliasing* filter as:

$$\mathbf{H}_0^{ideal} = \sum_{\lambda} h_0^{ideal}(\lambda)\mathbf{u}_\lambda \mathbf{u}_\lambda^t \tag{4.14}$$

where spectral kernel h_0^{ideal} is given as:

$$h_0^{ideal}(\lambda) = \begin{cases} 2 & \text{if } \lambda < k \\ 0 & \text{if } \lambda \geq k \end{cases} \tag{4.15}$$

Using (4.15) and (4.13), we get:

$$\begin{aligned} \mathbf{H}_0^{ideal}\mathbf{f}_{du} &= \sum_{\lambda,\gamma} h_0^{ideal}(\lambda).\bar{\mathbf{f}}_{du}(\gamma)\mathbf{u}_\lambda \mathbf{u}_\lambda^t \mathbf{u}_\gamma \\ &= \sum_{\lambda} h_0^{ideal}(\lambda).\bar{\mathbf{f}}_{du}(\lambda)\mathbf{u}_\lambda \\ &= \frac{1}{2}\sum_{\lambda<k} 2.\bar{\mathbf{f}}(\lambda)\mathbf{u}_\lambda + \frac{1}{2}\sum_{\lambda>k} 0.\bar{\mathbf{f}}(2k-\lambda)\mathbf{u}_\lambda \\ &= \sum_{\lambda<k} \bar{\mathbf{f}}(\lambda)\mathbf{u}_\lambda = \mathbf{f} \end{aligned} \tag{4.16}$$

Thus, $\mathbf{f}$ can be recovered from $\mathbf{f}_{du}$ by applying anti-aliasing filter $\mathbf{H}_0^{ideal}$.

□

4.3 Extension to non-regular bipartite graphs

The results obtained in the case of k-RBG cannot be extended to other non-regular bipartite graphs, if using combinatorial Laplacian matrix $\mathbf{L}$. The reason is that for non-regular graphs the degree of nodes is not a constant, and therefore adjacency matrix $\mathbf{A}$ and the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$ do not share the same set of eigenvectors. Therefore, for general bipartite graphs, instead of operating on the unnormalized adjacency matrix $\mathbf{A}$, we operate on a *symmetric normalized adjacency matrix* $\mathcal{A} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. The normalization reweighs the edges of graph G so that the degree of each node is equal to 1. Further, we choose eigenvectors of the normalized

Laplacian matrix $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$, as the GFT basis functions[2]. To understand the spectral interpretation of *DU* operations in bipartite graphs, the following properties of bipartite graphs are useful:

Lemma 2 ([3, Lemma 1.8])**.** *The following statements are equivalent for any graph $\mathcal{B}$:*

1. *$\mathcal{B}$ is bipartite with bipartitions H and L.*
2. *The spectrum of $\mathcal{L}(\mathcal{B})$ is symmetric about 1 and the minimum and maximum eigenvalues of $\mathcal{L}(\mathcal{B})$ are 0 and 2 respectively.*
3. *If $\mathbf{u} = \begin{bmatrix}\mathbf{u}_1^T & \mathbf{u}_2^T\end{bmatrix}^T$ is an eigenvector of $\mathcal{L}$ with eigenvalue λ with $\mathbf{u}_1$ indexed on H and $\mathbf{u}_2$ indexed on L (or vice-versa) then the modulated eigenvector $\hat{\mathbf{u}} = \begin{bmatrix}\mathbf{u}_1^T & -\mathbf{u}_2^T\end{bmatrix}^T$ is also an eigenvector of $\mathcal{L}$ with eigenvalue $2-\lambda$.*

Note that the results in Lemma 2 are similar to the results for k-RBG in Lemma 1. The primary difference between these results is that the Laplacian matrix used for k-RBG is the combinatorial Laplacian matrix $\mathbf{L}$, while for bipartite graphs we use symmetric normalized Laplacian matrix $\mathcal{L}$. However, since k-RBG graphs are also bipartite graphs, the results in Lemma 2 are also applicable to them. Therefore, from now on we only make use of symmetric normalized Laplacian matrix, unless otherwise stated. This enables us to define the following spectral folding result, similar to k-RBG graphs:

Proposition 2. *Given a bipartite graph $\mathcal{B} = (L, H, E)$ with Laplacian matrix $\mathcal{L}$, if we choose downsampling function β as β_H or β_L as defined in (2.7), and if $\mathbf{P}_\lambda$ is the projection matrix corresponding to the eigenspace V_λ, then*

$$\mathbf{J}_\beta \mathbf{P}_\lambda = \mathbf{P}_{2-\lambda}\mathbf{J}_\beta. \tag{4.17}$$

Alternatively, if $\mathbf{u}_\lambda$ is an eigen-vector of $\mathcal{L}$ with eigenvalue λ then $\mathbf{J}_\beta\mathbf{u}_\lambda$ is also an eigen-vector of $\mathcal{L}$ with eigen-value $2-\lambda$.

Proof. Let λ be an eigenvalue of $\mathcal{B}$ with multiplicity k. This implies that there exists an orthogonal set of k eigenvectors $\{\mathbf{u}_i\}_{\lambda_i=\lambda}$ of Laplacian matrix $\mathcal{L}$ with eigenvalue λ. The projection matrix $\mathbf{P}_\lambda$

[2]Note that the eigenvectors of Laplacian matrix $\mathbf{L}$ and normalized Laplacian matrix $\mathcal{L}$ are identical for regular graphs. The normalized Laplacian matrix popularized by Fan K. Chung [3] is found to be a more natural tool to deal with non-regular graphs.

corresponding to λ is given by $\mathbf{P}_\lambda = \sum_{\lambda_i=\lambda} \mathbf{u}_i.\mathbf{u}_i^t$. If the downsampling function β is chosen as β_H or β_L, then the modulated eigenvector $\hat{\mathbf{u}}$ in Lemma 2 is equal to $\mathbf{J}_\beta\mathbf{u}$, which is an eigen-vector of $\mathcal{L}$ with eigen-value $2-\lambda$. It can also be seen that if eigenvectors $\{\mathbf{u}_i\}_{\lambda_i=\lambda}$ are orthogonal to each other then so are the modulated set of eigenvectors $\{\mathbf{J}_\beta\mathbf{u}_i\}_{\lambda_i=\lambda}$ and form basis of eigenspace $\mathbf{P}_{2-\lambda}$. Therefore, $\mathcal{L}\mathbf{J}_\beta\mathbf{P}_\lambda\mathbf{J}_\beta = \sum_{\lambda_i=\lambda} \mathcal{L}.\mathbf{J}_\beta\mathbf{u}_i.(\mathbf{J}_\beta\mathbf{u}_i)^t = \sum_{\lambda_i=\lambda}(2-\lambda).\mathbf{J}_\beta\mathbf{u}_i.(\mathbf{J}_\beta\mathbf{u}_i)^t = (2-\lambda)\mathbf{P}_{2-\lambda}$, therefore $\mathbf{J}_\beta\mathbf{P}_\lambda\mathbf{J}_\beta = \mathbf{P}_{2-\lambda}$ which implies that $\mathbf{J}_\beta\mathbf{P}_\lambda = \mathbf{P}_{2-\lambda}\mathbf{J}_\beta$. □

This phenomenon is termed as *spectrum folding* in bipartite graphs, as the modulated eigenvector (or eigenspace) for any $\lambda \in \sigma(\mathcal{B})$ appears as another eigenvector (or eigenspace) at a mirror eigenvalue around $\lambda = 1$. **As a result, for any graph-signal $\mathbf{f}$ on $\mathcal{B}$, the modulated signal $\mathbf{J}_\beta\mathbf{f}$ is an aliased (although frequency reversed) version of $\mathbf{f}$.** To understand it, let $\mathbf{f}$ be an N-D graph-signal on bipartite graph $\mathcal{B} = (L, H, E)$ with eigenspace decomposition

$$\mathbf{f} = \sum_{\lambda\in\sigma(\mathcal{B})} \mathbf{P}_\lambda\mathbf{f} = \sum_{\lambda\in\sigma(\mathcal{B})} \mathbf{f}^\lambda, \tag{4.18}$$

where $\mathbf{f}^\lambda = \mathbf{P}_\lambda\mathbf{f}$ is the projection of $\mathbf{f}$ onto the eigenspace V_λ. Similarly, we define $(\mathbf{J}_\beta\mathbf{f})^\lambda$ as the projection of $\mathbf{J}_\beta\mathbf{f}$ onto the eigenspace V_λ. Using (4.17), we can write $(\mathbf{J}_\beta\mathbf{f})^\lambda$ as:

$$(\mathbf{J}_\beta\mathbf{f})^\lambda = \mathbf{P}_\lambda\mathbf{J}_\beta\mathbf{f} = \mathbf{J}_\beta\mathbf{P}_{2-\lambda}\mathbf{f} = \mathbf{J}_\beta\mathbf{f}^{2-\lambda}. \tag{4.19}$$

Thus for bipartite graphs, $(\mathbf{J}_\beta\mathbf{f})^\lambda$, is same as modulated $\mathbf{f}^{2-\lambda}$. Further, using (4.19) and the fact that $\mathbf{J}_\beta^2 = \mathbf{I}$, we can show that:

$$||(\mathbf{J}_\beta\mathbf{f})^\lambda||_2^2 = ||\mathbf{J}_\beta\mathbf{f}^{2-\lambda}||_2^2 = (\mathbf{f}^{2-\lambda})^t.\mathbf{J}_\beta^2.\mathbf{f}^{2-\lambda} = ||\mathbf{f}^{2-\lambda}||_2^2, \tag{4.20}$$

which implies that the energy of modulated signal in eigenspace V_λ is equal to the energy of original signal in $V_{2-\lambda}$ eigenspace. Both (4.19) and (4.20) show that $\mathbf{J}_\beta\mathbf{f}$ is an aliased version of $\mathbf{f}$. Using (4.19), the eigenspace decomposition of output signal $\mathbf{f}_{du}$ after *DU* operation can be written as:

$$\mathbf{f}_{du} = \frac{1}{2}(\mathbf{f} + \mathbf{J}_\beta\mathbf{f}) = \frac{1}{2}\sum_{\lambda\in\sigma(\mathcal{B})} (\mathbf{f}^\lambda + \mathbf{J}_\beta\mathbf{f}^{2-\lambda}) = \frac{1}{2}(\mathbf{f} + \mathbf{f}^{alias}) \tag{4.21}$$

In other words, the output signal is the average of the original signal and a shifted and aliased

version of the original signal.

4.4 Example: Images as kRBG

In this section, we show some examples of how downsampling applied on the graph representation of images leads to familiar results that could be derived from the standard frequency formulation. Digital images are 2-D regular signals, but they can also be represented as graphs by connecting every pixel (node) in an image with its neighboring pixels (nodes) and by interpreting pixel values as the values of the graph-signal at each node. to apply downsampling in the image Figure 4.2 shows some of the ways in which pixels in an image can be connected with other pixels to formulate a bipartite graph representation of any image. The bipartite sets L and H on the graphs are shown as nodes of different colors. Thus, the image after downsampling in each graph case, would consist of samples of only one color set. The decision to choose a particular graph representation can be compared with choosing various lattice-subsampling patterns [27] in standard image processing. For example, we can choose Figure 4.2(a) to represent the image which is equivalent to the *quincunx* downsampling scheme. Similarly, Figures 4.2(c) and 4.2(d) are equivalent to rectangular downsampling schemes ($\mathbf{V} = [2\ 0; 0\ 1]$) and ($\mathbf{V} = [1\ 0; 0\ 2]$) respectively. The advantage of using a graph representation of the images is that it provides flexibility of linking pixels in arbitrary ways, leading to different filtering/downsampling patterns. For example, we can represent images as bipartite graphs by connecting each pixel with its 4 diagonal neighbors. This bipartization scheme is shown in Figure 4.2(b). In this scheme since the pixels are connected along the diagonal axis, which can be useful for images with high frequency response in diagonal directions

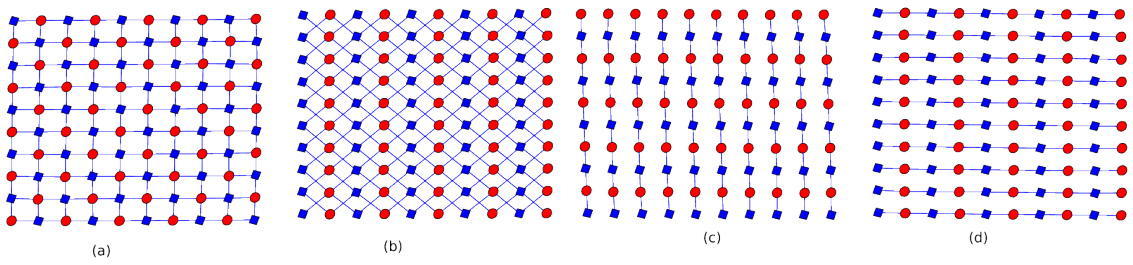


Figure 4.2: Bipartite graph representation of 2D images (a) a 4 connected rectangular image-graph G^r (b) a 4-connected diagonal image graph G^d, (c) a 2-connected image-graph G_v with vertical links only, and (d) a 2-connected image-graph G_h with horizontal links only.

Once we choose a graph representation of image, we can design filtering operations on graphs, based on the theory developed in this chapter. In order to compare the spectral interpretation of filtering operations based on graph, we implement ideal low-pass graph filters for different graph representations of the images. Since, the image-graphs are bipartite graphs, the ideal spectral lowpass filter $\mathbf{H}_0^{ideal}$ on these graph can be computed as in (4.14). In Figure 4.3, we plot the DFT magnitude response of ideal lowpass spectral transforms on these bipartite image-graphs. Because of the regularity and symmetry of the links, the resulting filters at each node, are translated versions of each other (except at the boundary nodes), and so we can compute the 2-D DFT magnitude response of a spectral transform, by computing the DFT response of the filtering operations at a single node. In particular, we choose a row of the ideal spectral lowpass filter, corresponding to a pixel away from the boundary, reshape it into two dimensions as the original image, and perform a 2-dimensional DFT.

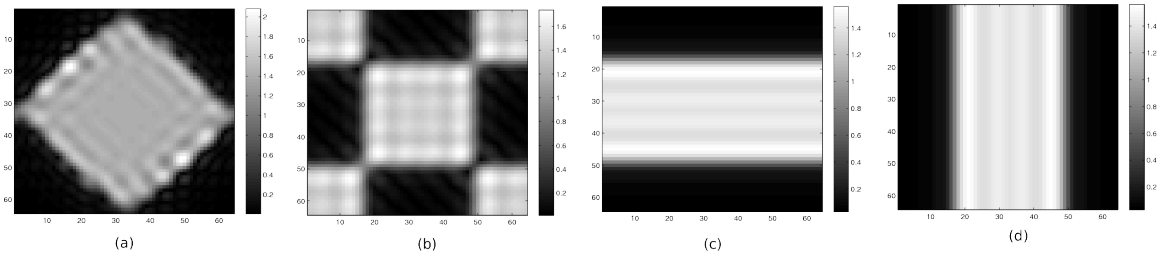


Figure 4.3: DFT magnitude responses of the ideal spectral filters $\mathbf{H}_0^{ideal}$ on (a) G_r, (b) G_d and (c) G_v.

We observe in Figure 4.2(a) that, the downsampling pattern (red/blue nodes) on the rectangular subgraph G^r is identical to the the quincunx downsampling pattern, and in Figure 4.3(a), it can be observed that the DFT magnitude response of $\mathbf{H}_0^{ideal}$ filter on G^r is same as the DFT magnitude response of the standard anti-aliasing filter for quincunx downsampling. Similarly, we observe that the spectral low-pass filter for G^v (or G^h) in Figure 4.2(c) (or 4.2(d)), have the same DFT magnitude responses (Figure 4.3(c) (or Figure 4.3(d))) as the anti-aliasing filters for vertical (or horizontal) factor-of-2 downsampling case. The graph formulation of images also allows us to explore new downsampling patters, for example, the image pixels can be connected to their diagonally opposite neighbors as shown in Figure 4.2(b). The DFT magnitude response of the ideal spectral low-pass filter in this case, is shown in Figure 4.3(b) and has a wider passband in the diagonal directions.

4.5 Summary

In this chapter, we have proposed a method for downsampling datasets defined on graphs. Especially, we have described a *spectral folding* phenomenon in bipartite graphs, which leads to aliasing in the spectral domain of the graph. Further, we have shown that images can be represented as bipartite graphs and shown some of the bipartization examples. The anti-aliasing filters on some of these image-graphs have identical frequency response, as those designed using standard signal framework. The results show that our proposed graph based method provide an alternative way of interpreting downsampling filtering in images. In the next chapter, we use this property to design critically sampled two-channel filterbanks on graphs.

Chapter 5

Two-channel Wavelet Filterbanks on Bipartite Graphs

In Chapter 4, we described downsampling/upsampling (DU) operations in bipartite graphs, which produce a *spectral folding* phenomenon in graph signals. In this chapter, we utilize this property of bipartite graphs to propose designs of critically sampled two-channel wavelet filterbanks for analyzing functions defined on the vertices of any arbitrary finite weighted undirected graph. A general framework for the design of two channel critically sampled filterbanks on graphs was introduced in Section 2.4. In this chapter, we expand upon this framework for the special case of bipartite graphs. We specifically design wavelet filters based on the spectral decomposition of the graph, which helps us translate the aliasing-cancellation, and perfect reconstruction conditions given in (2.16), in very simple terms. In addition, we state necessary and sufficient conditions for orthogonality in these filterbanks. As a practical solution, we propose quadrature mirror filterbanks (referred to as graph-QMF) for bipartite graph which have all the above mentioned properties. While the exact *graph-QMF* designs satisfy all the above conditions, they are not compactly supported[1] on the graph . In order to design compactly supported graph-QMF transforms, we perform a Chebychev polynomial approximation of the exact filters in the spectral domain. This leads to some error in the reconstruction of the signal, and loss of orthogonality. As an alternative, we relax the conditions of orthogonality and design filters with compact support, which satisfy the perfect reconstruction conditions. These biorthogonal designs are of two types: the first type of filterbanks have 1-hop localized analysis filters. These filterbanks are invertible, and we choose synthesis filters to guarantee perfect reconstruction. The second type of designs, termed as *graph-Bior*, have both analysis and synthesis filters with compact support.

[1] see Section 2.1 for definition of compact support for functions defined on graphs.

This design is analogous to the standard Cohen-Daubechies-Feauveau's (CDF) [5] construction to obtain maximally half-band filters. Even though these filterbanks are not orthogonal, we show that they can be designed to nearly preserve energy. In particular, we compute expressions for Riesz bounds of the filterbanks, and choose graph-wavelets with the minimum difference between upper and lower Riesz bounds. The rest of the chapter is organized as follows: in Section 5.1, we formulate the problem of designing two-channel wavelet filterbank on graphs. In Section 5.2, we describe the filterbank design specific to bipartite graphs and state necessary and sufficient conditions for these filterbanks to provide perfect-reconstruction, aliasing cancellation and orthogonal basis. In Section 5.3, we describe the proposed graph-QMF solution, which satisfies all the above conditions. In Section 5.4, we describe the one-hop localized biorthogonal transforms on graphs. In Section 5.5, we revisit the necessary and sufficient conditions for perfect reconstruction and orthogonality, and pose them as solving a system of biorthogonal spectral kernels. This leads to the proposed *graphBior* solution. Finally, we summarize the chapter in Section 5.7.

5.1 Problem Formulation

A general formulation of two-channel wavelet filterbanks is described in Section 2.4. A block diagram of the two-channel critically sampled graph wavelet filterbanks is shown in Figure 2.1. For designing wavelet filters on graphs, we exploit similar concepts of spectral decomposition as in [14]. Because of this, it is useful to define analysis wavelet filters $\mathbf{H}_0$ and $\mathbf{H}_1$ in terms of spectral kernels $h_0(\lambda)$ and $h_1(\lambda)$, respectively. In our analysis, we use the normalized form of the Laplacian matrix $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$, which in the case of regular graphs has the same set of eigenvectors as $\mathbf{L}$. The normalization reweighs the edges of graph G so that the degree of each node is equal to 1. Thus, given the eigen-space decomposition of Laplacian matrix $\mathcal{L}$ as in (2.5), the analysis filters can be represented as:

$$\begin{aligned} \mathbf{H}_0 = h_0(\mathcal{L}) &= \sum_{\lambda \in \sigma(G)} h_0(\lambda)\mathbf{P}_\lambda \\ \mathbf{H}_1 = h_1(\mathcal{L}) &= \sum_{\lambda \in \sigma(G)} h_1(\lambda)\mathbf{P}_\lambda \end{aligned} \tag{5.1}$$

Since the Laplacian matrix $\mathcal{L}$ is real and symmetric, the filters designed in (5.1) are also real and symmetric. These filters have the following interpretation: the output of a spectral filter

with kernel $h(\lambda)$ can be expanded as: $\mathbf{f}_H = \mathbf{H}\mathbf{f} = \sum_{\lambda\in\sigma(G)} h_i(\lambda)\ \mathbf{P}_\lambda\mathbf{f}$, where $\mathbf{f}^\lambda = \mathbf{P}_\lambda\mathbf{f}$ is the component of input signal $\mathbf{f}$ in the λ-eigenspace. Thus, filter $\mathbf{H}$ either attenuates or enhances different harmonic components of input signals depending upon the magnitude of $h(\lambda)$. For a general kernel function, the filtering operations with $\mathbf{H}_i$ and $\mathbf{G}_i$ require spectral decomposition of the Laplacian matrix, which is non-scalable and computationally expensive. However it has been shown in [14] that when the spectral kernels are approximated as polynomial kernels of degree k, the filters can be computed iteratively with k one-hop operations at each node. Thus, the computational complexity of the filtering operations reduces to $\mathcal{O}(k|E|)$, which increases linearly with the number of edges $|E|$ of the graph, and the degree k of polynomial approximations. Further, any spectral transform corresponding to a k degree polynomial kernel is exactly *k-hop spatially localized* , and can be efficiently computed without diagonalizing the Laplacian matrix. The degree of the polynomial kernels can be interpreted as the length of the corresponding spectral filters, and one of the desirable feature of graph wavelets is to have shorter filter lengths. Referring again to Figure 2.1, for graph $G = (L, H, E)$, let $\beta_H = \beta$ be the downsampling function for $\mathbf{H}_1$ filter channel and let $\beta_L = -\beta$ be the downsampling function for $\mathbf{H}_0$ channel. Thus the nodes in H only retain the output of highpass channel and nodes in L retain the output of the lowpass channel. In our proposed design, we also choose the synthesis filters $\mathbf{G}_0$ and $\mathbf{G}_1$ to be spectral filters with kernels $g_0(\lambda)$ and $g_1(\lambda)$ respectively [3]. Let $\mathbf{f}$ be the graph signal on G, and $\mathbf{f}^\lambda = \mathbf{P}_\lambda\mathbf{f}$ is the projection of $\mathbf{f}$ onto the eigenspace V_λ. Then by using (5.1), the perfect reconstruction conditions in (2.16) can be rewritten as:

$$\begin{aligned}
\mathbf{T}_{eq}\mathbf{f} &= (\mathbf{G}_0\mathbf{H}_0 + \mathbf{G}_1\mathbf{H}_1)\mathbf{f} \\
&= \sum_{\lambda,\gamma\in\sigma(G)} \left(g_0(\lambda)h_0(\gamma) + g_1(\lambda)h_1(\gamma)\right)\mathbf{P}_\lambda\mathbf{P}_\gamma\mathbf{f} \\
&= \sum_{\lambda\in\sigma(G)} \left(g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)\right)\mathbf{P}_\lambda\mathbf{f} \\
&= \sum_{\lambda\in\sigma(G)} \left(g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)\right)\mathbf{f}^\lambda
\end{aligned} \tag{5.2}$$

[3] In general, synthesis filters do not have to be based on the spectral design. A case is presented in our previous work [29] with linear kernel spectral analysis filters and non-spectral synthesis filters.

$$\begin{aligned}
\mathbf{T}_{alias}\mathbf{f} &= (\mathbf{G}_1\mathbf{J}_\beta\mathbf{H}_1 - \mathbf{G}_0\mathbf{J}_\beta\mathbf{H}_0)\mathbf{f} \\
&= \sum_{\lambda,\gamma\in\sigma(G)} \left(g_1(\lambda)h_1(\gamma) - g_0(\lambda)h_0(\gamma)\right)\mathbf{P}_\lambda\mathbf{J}_\beta\mathbf{P}_\gamma\mathbf{f} \\
&= \sum_{\lambda,\gamma\in\sigma(G)} \left(g_1(\lambda)h_1(\gamma) - g_0(\lambda)h_0(\gamma)\right)\mathbf{P}_\lambda\mathbf{J}_\beta\mathbf{f}^\gamma \qquad (5.3)
\end{aligned}$$

In (5.2), we use the orthogonality property of eigenspaces, given in (2.6), which reduces the double summation over λ and γ in the expansion of $\mathbf{T}_{eq}$, into a single summation over λ. However, the same cannot be applied in (5.6), where projection $\mathbf{f}^\gamma$ of signal $\mathbf{f}$ onto V_γ eigenspace, is modulated with downsampling matrix $\mathbf{J}_\beta$ before multiplication with $\mathbf{P}_\lambda$. For an arbitrary graph, modulated signal $\mathbf{J}_\beta\mathbf{f}^\gamma$ may not entirely belong to a single eigenspace (in fact it can have components in all eigenspaces). This means that for arbitrary graphs alias-free perfect reconstruction can be guaranteed for any graph-signal, if

$$\begin{aligned}
g_1(\lambda)h_1(\gamma) - g_0(\lambda)h_0(\gamma) &= 0 \\
g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)) &= c^2, \qquad (5.4)
\end{aligned}$$

for some constant c and for all $\lambda, \gamma \in \sigma(G)$. Note that the system of equations in (5.4) may not have a solution for every graph G, since there are more constraints ($\mathcal{O}(N^2)$) than the number of variables ($\mathcal{O}(N)$). However for bipartite graphs, (5.4) can be simplified to $\mathcal{O}(N)$ constraints, because of the spectral folding property. This is discussed in the next section.

5.2 Two-Channel Filterbank Conditions for Bipartite Graphs

We showed in Chapter 4 that, if the underlying graph is a bipartite graph $\mathcal{B} = (L, H, E)$, and if we choose the downsampling function β to be either β_H or β_L, then $\mathbf{J}_\beta\mathbf{f}$ is an alias of signal $\mathbf{f}$. Using the spectral folding property of bipartite graphs in (4.19), $\mathbf{J}_\beta\mathbf{f}^\gamma$ can be expressed as:

$$\mathbf{J}_\beta\mathbf{f}^\gamma = (\mathbf{J}_\beta\mathbf{f})^{2-\gamma}, \qquad (5.5)$$

where $(\mathbf{J}_\beta \mathbf{f})^{2-\gamma}$, is the projection of modulated graph-signal $\mathbf{J}_\beta \mathbf{f}$, onto eigenspace $V_{2-\gamma}$. This implies that $\mathbf{J}_\beta \mathbf{f}^\gamma$ belongs to eigenspace $V_{2-\gamma}$ for bipartite graphs. This property, when combined with (5.6) and orthogonality property of eigenspaces in (2.6), leads to:

$$\begin{aligned}
\mathbf{T}_{alias}\mathbf{f} &= \sum_{\lambda,\gamma\in\sigma(\mathcal{B})} \left(g_1(\lambda)h_1(\gamma) - g_0(\lambda)h_0(\gamma)\right) \mathbf{P}_\lambda \mathbf{J}_\beta \mathbf{f}^\gamma \\
&= \sum_{\lambda,\gamma\in\sigma(\mathcal{B})} \left(g_1(\lambda)h_1(\gamma) - g_0(\lambda)h_0(\gamma)\right) \mathbf{P}_\lambda (\mathbf{J}_\beta \mathbf{f})^{2-\gamma} \\
&= \sum_{\lambda\in\sigma(\mathcal{B})} \left(g_1(\lambda)h_1(2-\lambda) - g_0(\lambda)h_0(2-\lambda)\right) (\mathbf{J}_\beta \mathbf{f})^{\lambda} \qquad (5.6)
\end{aligned}$$

Thus, in case of bipartite graphs, the double summation in the expansion of $\mathbf{T}_{alias}$ over λ and γ, reduces to a single summation over λ, as derived in (5.6), and perfect reconstruction in the filterbanks can be guaranteed if

$$\begin{aligned}
g_1(\lambda)h_1(2-\lambda) - g_0(\lambda)h_0(2-\lambda) &= 0 \\
g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)) &= c^2, \qquad (5.7)
\end{aligned}$$

for some constant c and for all $\lambda \in \sigma(\mathcal{B})$. The system of equations in (5.7) can also be represented in matrix form as:

$$\underbrace{\begin{bmatrix} h_0(\lambda) & h_1(\lambda) \\ -h_0(2-\lambda) & h_1(2-\lambda) \end{bmatrix}}_{\mathbf{H}_m(\lambda)} \begin{bmatrix} g_0(\lambda) \\ g_1(\lambda) \end{bmatrix} = \begin{bmatrix} c \\ 0 \end{bmatrix}, \qquad (5.8)$$

and will have atleast one solution for any bipartite graph, assuming full rank of $\mathbf{H}_m(\lambda)$ for all $\lambda \in \sigma(\mathcal{B})$ (i.e., $det(\mathbf{H}_m(\lambda)) \neq 0$, where $det(.)$ is the determinant of a matrix). Before proposing a solution of (5.7), we state necessary and sufficient conditions for a two-channel spectral filterbank to be aliasing-cancellation, perfect reconstruction and orthogonal on any bipartite graph.

5.2.1 Aliasing cancellation

Combining (5.2) and (5.6) we can write the overall transfer function of the two-channel spectral filterbank on any bipartite graph as:

$$\begin{aligned}\mathbf{y} = \mathbf{T}\mathbf{f} &= \sum_{\lambda\in\sigma(\mathcal{B})} \left(g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)\right)\mathbf{f}^{\lambda} \\ &+ \sum_{\lambda\in\sigma(\mathcal{B})} \left(g_1(\lambda)h_1(2-\lambda) - g_0(\lambda)h_0(2-\lambda)\right)(\mathbf{J}_\beta\mathbf{f})^{\lambda} \end{aligned} \tag{5.9}$$

Taking projections of both LHS and RHS in (5.9) onto V_γ space, we get:

$$\begin{aligned}\mathbf{P}_\gamma\mathbf{y} = \mathbf{y}^{\lambda} &= \underbrace{\left(g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda)\right)}_{T_{eq}(\lambda)}\mathbf{f}^{\lambda} \\ &+ \underbrace{\left(g_1(\lambda)h_1(2-\lambda) - g_0(\lambda)h_0(2-\lambda)\right)}_{T_{alias}(\lambda)}(\mathbf{J}_\beta\mathbf{f})^{\lambda}. \end{aligned} \tag{5.10}$$

Thus, projection $\mathbf{y}^{\lambda}$ of the output signal for all λ is a weighted linear sum of projections of input signal $\mathbf{f}$ and alias signal $\mathbf{J}_\beta\mathbf{f}$ onto the V_λ eigenspace. Therefore, an *alias-free reconstruction using spectral filters is possible if and only if* the weight of alias signal in (5.10) is zero for all $\lambda \in \sigma(\mathcal{B})$, i.e.,

$$T_{alias}(\lambda) = g_0(\lambda)h_0(2-\lambda) - g_1(\lambda)h_1(2-\lambda) = 0. \tag{5.11}$$

5.2.2 Perfect reconstruction

Referring again to (5.5), $(\mathbf{J}_\beta\mathbf{f})^{\lambda} = \mathbf{J}_\beta\mathbf{f}^{2-\lambda}$. Perfect reconstruction means that the reconstructed signal $\hat{\mathbf{f}}$ is the same as (or possibly a scaled version of) the input signal $\mathbf{f}$. This implies $\mathbf{y} = c^2\mathbf{f}$ in (5.9), or equivalently $\mathbf{y}^{\lambda} = c^2\mathbf{f}^{\lambda}$ in (5.10) for some constant c and for all $\lambda \in \sigma(\mathcal{B})$. Therefore, in case of perfect reconstruction (5.10) can be written as:

$$\mathbf{y}^{\lambda} = c^2\mathbf{f}^{\lambda} = T_{eq}(\lambda)\mathbf{f}^{\lambda} + T_{alias}(\lambda)\mathbf{J}_\beta\mathbf{f}^{2-\lambda}, \tag{5.12}$$

or

$$(c^2 - T_{eq}(\lambda))\mathbf{f}^{\lambda} = T_{alias}(\lambda)\mathbf{J}_\beta\mathbf{f}^{2-\lambda}, \tag{5.13}$$

Since $\mathbf{f}^{\lambda}$ and $\mathbf{f}^{2-\lambda}$ are mutually orthogonal components of input signal $\mathbf{f}$ and hence are independent of each other, the only way (5.13) holds for all signals is, if

$$\begin{aligned} \mathbf{T}_{eq}(\lambda) &= c^2, \\ \mathbf{T}_{alias}(\lambda) &= 0. \end{aligned} \tag{5.14}$$

Thus, a necessary and sufficient condition for *perfect reconstruction*, using spectral filters, in bipartite graphs filterbanks is that for all λ in $\sigma(\mathcal{B})$,

$$\begin{aligned} g_0(\lambda)h_0(\lambda) + g_1(\lambda)h_1(\lambda) &= c^2, \\ g_0(\lambda)h_0(2-\lambda) - g_1(\lambda)h_1(2-\lambda) &= 0. \end{aligned} \tag{5.15}$$

5.2.3 Orthogonality

The wavelet coefficient vector $\mathbf{w}$ produced in the filterbank shown in Figure 2.1 is given as:

$$\begin{aligned} \mathbf{w} = \mathbf{T}_a\mathbf{f} &= \frac{1}{2}\left((\mathbf{I}-\mathbf{J}_\beta)\mathbf{H}_0\mathbf{f} + (\mathbf{I}+\mathbf{J}_\beta)\mathbf{H}_1\mathbf{f}\right) \\ &= \frac{1}{2}(\mathbf{H}_1+\mathbf{H}_0)\mathbf{f} + \frac{1}{2}\mathbf{J}_\beta(\mathbf{H}_1-\mathbf{H}_0)\mathbf{f} \end{aligned} \tag{5.16}$$

Applying (5.1) in (5.16), we get:

$$\mathbf{w} = \frac{1}{2}\sum_{\lambda\in\sigma(\mathcal{B})}(h_1(\lambda)+h_0(\lambda))\mathbf{P}_\lambda\mathbf{f} + \frac{1}{2}\sum_{\lambda\in\sigma(\mathcal{B})}(h_1(\lambda)-h_0(\lambda))\mathbf{J}_\beta\mathbf{P}_\lambda\mathbf{f}. \tag{5.17}$$

Using (4.17), and changing the variable λ to $2-\lambda$ in the second summation term in (5.17), we get:

$$\mathbf{w} = \frac{1}{2}\sum_{\lambda\in\sigma(\mathcal{B})}\underbrace{(h_1(\lambda)+h_0(\lambda))}_{C_\lambda}\mathbf{f}^{\lambda} + \underbrace{(h_1(2-\lambda)-h_0(2-\lambda))}_{D_{2-\lambda}}(\mathbf{J}_\beta\mathbf{f})^{\lambda}. \tag{5.18}$$

Taking projections of both LHS and RHS in (5.18) onto V_λ space, we get:

$$\mathbf{w}^{\lambda} = \mathbf{P}_\lambda\mathbf{w} = \frac{1}{2}(C_\lambda\mathbf{f}^{\lambda} + D_{2-\lambda}(\mathbf{J}_\beta\mathbf{f})^{\lambda}). \tag{5.19}$$

The filterbank provides an *orthogonal decomposition* for any graph signal if and only if $||\mathbf{w}||_2^2 = ||\mathbf{T}_a\mathbf{f}||_2^2 = ||\mathbf{f}||_2^2$ for all $\mathbf{f} \in \mathbb{R}^N$. For brevity we simply denote 2-norm of $\mathbf{f}$ as $||\mathbf{f}||$. Since the eigenspaces of $\mathcal{L}$ are orthogonal, the projections $\mathbf{w}^\lambda$ are orthogonal and the energy $||\mathbf{w}^\lambda||^2$ can be computed as sum of energy of $\mathbf{w}$ in each eigenspace, i.e.,

$$||\mathbf{w}||^2 = \sum_{\lambda\in\sigma(\mathcal{B})} ||\mathbf{w}^\lambda||^2 = \sum_{\lambda\in\sigma(\mathcal{B})} ||\frac{1}{2}(C_\lambda\mathbf{f}^\lambda + D_{2-\lambda}(\mathbf{J}_\beta\mathbf{f})^\lambda)||^2. \tag{5.20}$$

With some algebraic manipulation $||\mathbf{w}^\lambda||^2$ in (5.20) can be written as:

$$||\mathbf{w}^\lambda||^2 = \frac{1}{4}\left(C_\lambda^2||\mathbf{f}^\lambda||^2 + D_{2-\lambda}^2||(\mathbf{J}_\beta\mathbf{f})^\lambda)||^2 + 2C_\lambda D_{2-\lambda} < \mathbf{f}^\lambda\ (\mathbf{J}_\beta\mathbf{f})^\lambda >\right). \tag{5.21}$$

Using (4.19) and (4.20) in (5.21), we get:

$$||\mathbf{w}^\lambda||^2 = \frac{1}{4}\left(C_\lambda^2||\mathbf{f}^\lambda||^2 + D_{2-\lambda}^2||\mathbf{f}^{2-\lambda}||^2 + 2C_\lambda D_{2-\lambda} < \mathbf{f}^\lambda\ \mathbf{J}_\beta\mathbf{f}^{2-\lambda} >\right). \tag{5.22}$$

It can be seen from (5.22), that $||\mathbf{w}^\lambda||^2$ only depends on the $\mathbf{f}^\lambda$ and $\mathbf{f}^{2-\lambda}$ components of signal $\mathbf{f}$. Similarly, $||\mathbf{w}^{2-\lambda}||^2$ also depends on only the $\mathbf{f}^\lambda$ and $\mathbf{f}^{2-\lambda}$ components of signal $\mathbf{f}$. Further, $\mathbf{f}^\lambda$ and $\mathbf{f}^{2-\lambda}$ are only used to compute $\mathbf{w}^\lambda$ and $\mathbf{w}^{2-\lambda}$. Therefore, for all $\lambda \in \sigma(\mathcal{B})$, if $\mathbf{f} = \mathbf{f}^\lambda + \mathbf{f}^{2-\lambda}$, then $\mathbf{w} = \mathbf{w}^\lambda + \mathbf{w}^{2-\lambda}$. Thus, orthogonality of filterbank is guaranteed if for all $\lambda \in \sigma(\mathcal{B})$:

$$||\mathbf{w}^\lambda||^2 + ||\mathbf{w}^{2-\lambda}||^2 = ||\mathbf{f}^\lambda||^2 + ||\mathbf{f}^{2-\lambda}||^2 \tag{5.23}$$

Using (5.22), and with some algebraic manipulation we can write:

$$\begin{aligned} ||\mathbf{w}^\lambda||^2 + ||\mathbf{w}^{2-\lambda}||^2 &= \frac{1}{4}(C_\lambda^2 + D_\lambda^2)||\mathbf{f}^\lambda||^2 \\ &+ \frac{1}{4}(C_{2-\lambda}^2 + D_{2-\lambda}^2)||\mathbf{f}^{2-\lambda}||^2 \\ &+ \frac{1}{2}(C_\lambda D_{2-\lambda} + C_{2-\lambda}D_\lambda) < \mathbf{f}^\lambda\ \mathbf{J}_\beta\mathbf{f}^{2-\lambda} > . \end{aligned} \tag{5.24}$$

In (5.24), $\mathbf{f}^{\lambda}$ and $\mathbf{f}^{2-\lambda}$ are orthogonal to each other, and hence are independent of each other. Therefore, for (5.23) to hold true for all $\lambda \in \sigma(\mathcal{B})$ and for all signals $\mathbf{f} \in \mathbb{R}^N$:

$$\begin{aligned} C_{\lambda}^2 + D_{\lambda}^2 &= 4 \\ C_{\lambda} D_{2-\lambda} + C_{2-\lambda} D_{\lambda} &= 0. \end{aligned} \tag{5.25}$$

Expanding C_{λ} and D_{λ} in terms of $h_0(\lambda)$ and $h_1(\lambda)$, we get:

$$\begin{aligned} C_{\lambda}^2 + D_{\lambda}^2 &= 2(h_0^2(\lambda) + h_1^2(\lambda)) = 4 \\ C_{\lambda} D_{2-\lambda} + C_{2-\lambda} D_{\lambda} &= h_0(\lambda) h_0(2-\lambda) - h_1(\lambda) h_1(2-\lambda) = 0. \end{aligned} \tag{5.26}$$

all λ. Thus, a necessary and sufficient condition for orthogonality in bipartite graph filterbanks using spectral filters is :

$$\begin{aligned} h_0(\lambda) h_0(2-\lambda) - h_1(\lambda) h_1(2-\lambda) &= 0 \\ h_0^2(\lambda) + h_1^2(\lambda) &= 2. \end{aligned} \tag{5.27}$$

Note that comparing (5.15) and (5.27), the orthogonality conditions can be obtained from the perfect reconstruction conditions by selecting $g_0(\lambda) = h_0(\lambda)$ and $g_1(\lambda) = h_1(\lambda)$. This is analogous to the case of standard filterbanks and leads to our proposed graph-QMF design as explained in Section 5.3.

5.3 Graph-QMF Filterbanks

In this section, we extend the well-known quadrature mirror filter (QMF) solution to the case of bipartite graphs. Our proposed solution, termed as graph-QMF, requires the design of a single spectral kernel $h_0(\lambda)$, while the other spectral kernels are chosen as a function of $h_0(\lambda)$ as:

$$\begin{aligned} h_1(\lambda) &= h_0(2-\lambda) \\ g_0(\lambda) &= h_0(\lambda) \\ g_1(\lambda) &= h_1(\lambda) = h_0(2-\lambda) \end{aligned} \tag{5.28}$$

Proposition 3 (QMF Filters on Graph)**.** *For a bipartite graph $G = (L, H, E)$, let a two-channel filterbank be as shown in Figure 2.1 with the downsampling function $\beta = \beta_H$ and with spectral filters $\{\mathbf{H}_0, \mathbf{H}_1, \mathbf{G}_0, \mathbf{G}_1\}$ corresponding to spectral kernels $\{h_0(\lambda), h_1(\lambda), g_0(\lambda), g_1(\lambda)\}$ respectively. Then for any arbitrary choice of kernel $h_0(\lambda)$, the proposed graph-QMF solution cancels aliasing in the filterbank. In addition for solution $h_0(\lambda)$ such that $h_0(\lambda)^2 + h_0(2-\lambda)^2 = c^2$ for all $\lambda \in \sigma(\mathcal{B})$ and $c \neq 0$ the filterbank provides perfect reconstruction and an orthogonal decomposition of graph-signals.*

Proof. Substituting (5.28) into (5.11) leads to $g_0(\lambda)h_0(2-\lambda) - g_1(\lambda)h_1(2-\lambda) = 0$ and aliasing is indeed canceled. The reconstructed signal $\hat{\mathbf{x}}$ in this case is simply equal to $(1/2)\mathbf{T}_{eq}\mathbf{x}$ and can be written as:

$$\hat{\mathbf{x}} = \frac{1}{2} \sum_{\lambda \in \sigma(\mathcal{B})} (h_0^2(\lambda) + h_0^2(2-\lambda))\mathbf{x}^\lambda \tag{5.29}$$

Thus for $(h_0^2(\lambda) + h_0^2(2-\lambda)) = c^2$ and $c \neq 0$, the reconstructed signal $\hat{\mathbf{x}} = \frac{c^2}{2}\mathbf{x}$ is a scaled version of the original signal. Similarly applying the mirror design $h_1(\lambda) = h_0(2-\lambda)$ in the conditions (5.27) we get $h_0(\lambda)h_0(2-\lambda) - h_1(\lambda)h_1(2-\lambda) = 0$ and $h_0^2(\lambda) + h_1^2(\lambda) = c^2$ and hence the corresponding analysis side transform $\mathbf{T}_a$ is orthogonal. □

5.3.1 Chebychev polynomial approximation

We now consider the design of kernels $h_0(\lambda)$ satisfying the design constraint of Proposition 3, i.e., for which $h_0^2(\lambda) + h_0^2(2-\lambda) = c^2$ for all $\lambda \in \sigma(\mathcal{B})$. For maximum spectrum splitting in the two channels of the filterbank, the ideal choice of kernel $h_0(\lambda)$ would be a lowpass rectangular function on λ given as:

$$h_0^{ideal}(\lambda) = \begin{cases} c & \text{if } \lambda < 1 \\ c/\sqrt{(2)} & \text{if } \lambda = 1 \\ 0 & \text{if } \lambda > 1 \end{cases} \tag{5.30}$$

The corresponding ideal filter is given by

$$\mathbf{H}_0^{ideal} = \sum_{\lambda<1} c\mathbf{P}_\lambda + \frac{c}{\sqrt{2}}\mathbf{P}_{\lambda=1} \tag{5.31}$$

Note that the ideal transform has a non-analytic spectral kernel response with sharp peaks and is therefore a global transform (i.e., the filters operations are not localized). Even analytic solutions

of the constraint equation $h_0^2(\lambda) + h_0^2(2-\lambda) = c^2$, such as $h_0(\lambda) = c\sqrt{1-\lambda/2}$ or $h_0(\lambda) = c\ cos(\pi\lambda/4)$, are not compactly supported in the spatial domain. By relaxing the constraints one can obtain compact support solutions at the cost of some small reconstruction error and near-perfect orthogonality. One such solution is the approximation of the desired kernel with a polynomial kernel. We choose polynomial approximations of the desired kernel due to the following localization property for corresponding transforms:

Lemma 3 ([14]). *Let $h_0(\lambda)$ be a polynomial of degree k and let $\mathcal{L}$ be the normalized Laplacian matrix for any weighted graph G, then the matrix polynomial $\mathbf{H}_0 = h_0(\mathcal{L})$ is exactly k-hop localized at each node of G. In other words for any two nodes n and m if $m \notin \mathcal{N}_k(n)$ then $\mathbf{H}_0(n,m) = 0$.*

Further, we choose a minimax polynomial approximation which minimizes the Chebychev norm (worst-case norm) of the reconstruction error since it has been shown in [14] that it also minimizes the upper-bound on the error $||H^{ideal} - H^{poly}||$ between ideal and approximated filters. Thus, in order to localize the filters on the graph, we approximate h_0^{ideal} with the truncated Chebychev polynomials (which are a good approximation of minimax polynomials) of different orders. However since h_0^{ideal} is a rectangular function it projects a lot of its energy in the truncated part of the polynomial expansions and as a result the polynomial approximation errors for h_0^{ideal} are high. A possible solution of this problem is to soften the ideal case, by finding a smooth function that is low-pass and satisfies the constraint. An analogous construction in regular signal processing is *Meyer's wavelet* design which replaces the brick-wall type ideal frequency-response with a smooth scaling function that satisfies the orthogonality and scaling requirements. By a change in variable from $\omega \in [-1,1]$ to $\lambda \in [0,2]$ we can extend Meyer's wavelet construction in the case of bipartite graph. The construction involves choosing a function $\nu(x)$ such that $\nu(\lambda) = 0$ for $\lambda \leq 0$, $\nu(\lambda) = 1$ for $\lambda \geq 1$ and $\nu(\lambda) + \nu(1-\lambda) = 1$ everywhere. One such function is given as:

$$\nu(\lambda) = \begin{cases} 0 & \text{if } \lambda \leq 0 \\ 3\lambda^2 - 2\lambda^3 & \text{if } 0 \leq \lambda \leq 1 \\ 1 & \text{if } \lambda \geq 1 \end{cases}, \tag{5.32}$$

which leads to a smooth kernel given as:

$$h_0^{Meyer}(\lambda) = \sqrt{\nu(2 - \frac{3}{2}\lambda)} \tag{5.33}$$

In Figure 5.1(a), we plot the ideal and Meyer wavelet kernels and in Figures 5.1(b)-(f) we plot the reconstruction errors between desired kernels and their polynomial approximations of different orders. It can be seen that Meyer's wavelet approximations yield small reconstruction errors

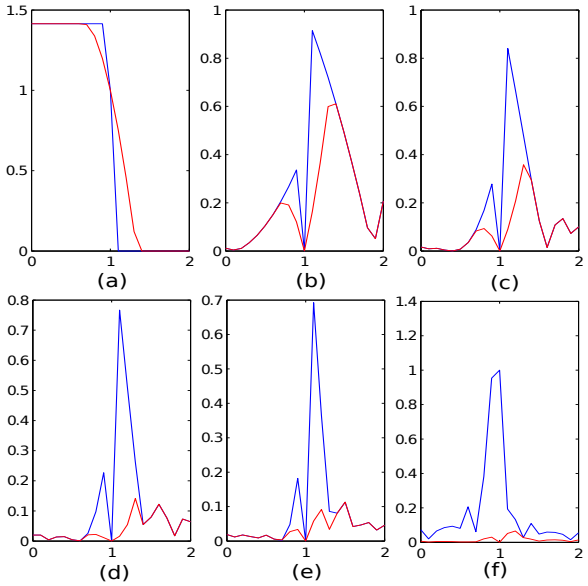


Figure 5.1: (a) Ideal kernel (blue) vs. Meyer's wavelet kernel (red). It can be seen that Meyer's wavelet has smoother transition at $\lambda = 1$ than the ideal kernel, (b)-(f) the reconstruction error magnitudes between original kernels and their polynomial approximations of order $2, 4, 6, 8$ and 10 respectively: ideal kernel (blue curves) and Meyers kernel(red curve).

as compared to ideal-filter approximations. Thus by choosing $h(\lambda)$ as the low-order polynomial approximations of smooth low-pass functions (such as Meyer's wavelets), we obtain near perfect reconstruction QMF wavelet filters on any bipartite graph which are very well localized in spatial domain. In order to obtain a bound on the reconstruction error due to polynomial approximation, we apply graph-QMF kernels given in (5.28) to the overall transform in (2.15). We observe that the graph-QMF filterbanks cancel aliasing term (i.e., term $\mathbf{T}_{alias}$) and the reconstructed signal can be written as:

$$\hat{\mathbf{f}} = \sum_{\lambda \in \sigma(\mathcal{B})} \frac{h_0^2(\lambda) + h_1^2(\lambda)}{2} \bar{f}(\lambda) \mathbf{u}_\lambda \tag{5.34}$$

Hence, the reconstruction MSE is given as:

$$MSE = \frac{1}{N}||\mathbf{f} - \hat{\mathbf{f}}||_2^2 = \frac{1}{N} \sum_{\lambda \in \sigma(\mathcal{B})} \left(1 - \frac{h_0^2(\lambda) + h_1^2(\lambda)}{2}\right)^2 \bar{f}^2(\lambda) \leq \frac{1}{N} \underbrace{\arg\max_{\lambda} \left(1 - \frac{h_0^2(\lambda) + h_1^2(\lambda)}{2}\right)^2}_{MSE_{max}} ||\mathbf{f}||_2^2 \tag{5.35}$$

where MSE_{max} is the square of maximum deviation of equivalent kernel response $\frac{h_0^2(\lambda)+h_1^2(\lambda)}{2}$ from 1. The SNR of reconstruction can thus be bounded from below using MSE_{max} as:

$$SNR = 10log_{10}\frac{\frac{1}{N}||\mathbf{f}||_2^2}{\frac{1}{N}||\mathbf{f}-\hat{\mathbf{f}}||_2^2} \geq -20log_{10}\sqrt{(MSE_{max})} \tag{5.36}$$

Table 5.2 shows the average SNR obtained using different polynomial approximations of graph-QMF filterbanks. The average is computed by randomly generating 20 graph-signals on 10 random instances of bipartite graphs, each with 100 nodes. We observe that increasing the degree of polynomial approximations leads to higher reconstruction SNR, at the cost of increasing computational complexity (i.e., longer filters). In the next section, we relax the condition of orthogonality, and turn to compactly supported designs, which are perfect reconstruction and can be designed with shorter polynomial spectral kernels.

5.4 One-hop Localized Spectral Filterbanks

We are interested in filterbanks which can be computed locally at each node (i.e., have compact support), critically sampled and invertible. To achieve compact support, instead of finding polynomial approximations of arbitrary kernel functions as described in Section 5.3, we propose designing low-degree polynomial kernels which have localized support, and are invertible. In this approach, we use analysis filters to be based on *spectral kernels*, similar to the ones defined in (5.1). We specifically choose degree-1 polynomial kernels:

$$\begin{aligned} h_0(\lambda) &= a_0\lambda + b_0 \\ h_1(\lambda) &= a_1\lambda + b_1, \end{aligned} \tag{5.37}$$

in which case the corresponding filters are given as:

$$\begin{aligned} \mathbf{H}_0 &= a_0\mathcal{L} + b_0\mathbf{I} \\ \mathbf{H}_1 &= a_1\mathcal{L} + b_1\mathbf{I}. \end{aligned} \tag{5.38}$$

Referring again to (5.16) the analysis wavelet transform $\mathbf{T}_a$ is given as:

$$\mathbf{T}_a = \frac{1}{2}(\mathbf{H}_1 + \mathbf{H}_0) + \frac{1}{2}\mathbf{J}_\beta(\mathbf{H}_1 - \mathbf{H}_0). \tag{5.39}$$

Thus, $\mathbf{T}_a$ corresponds to choosing m rows of lowpass transform matrix $\mathbf{H}_0$ for nodes in L and $N - m$ rows of highpass transform $\mathbf{H}_1$ for nodes in H.

5.4.1 One-hop localized designs for arbitrary graphs

Proposition 4 below states that transform $\mathbf{T}_a$ is invertible (non-singular) for any partition $L \cup H$ of vertex set for our choice of degree-1 kernel functions $h_0(\lambda)$ and $h_1(\lambda)$.

Proposition 4. *Let transforms $\mathbf{H}_0$ and $\mathbf{H}_1$ be two transforms on a graph G based on linear spectral kernels h_0 and h_1 as defined above, and $\mathbf{T}_a$ is the overall analysis transform. Let $\mathcal{V} = L \cup H$ be any partition of vertex set of G. Then for $a_0, a_1 \neq 0$ and $b_1/a_1, b_0/a_0 \geq 0$ with strict inequality for at least one b, the matrix $\mathbf{T}_a$ is invertible.*

Proof. Without loss of generality, let us assume $\mathbf{T}_a$ consists of first m rows of matrix $\mathbf{H}_1$ and remaining $l = N - m$ rows of matrix $\mathbf{H}_0$. Represent the Laplacian matrix $\mathbf{L}$ in block form as

$$\mathcal{L} = \begin{bmatrix} (\mathbf{L}_1)_{m\times m} & (\mathbf{L}_3)_{m\times l} \\ (\mathbf{L}_3')_{l\times m} & (\mathbf{L}_2)_{l\times l} \end{bmatrix}. \tag{5.40}$$

Using (5.39), $\mathbf{T}_a$ can be written as:

$$\begin{aligned} \mathbf{T}_a &= \begin{bmatrix} a_1\mathbf{L}_1 + b_1\mathbf{I}_1 & a_1\mathbf{L}_3 \\ a_0\mathbf{L}_3' & a_0\mathbf{L}_2 + b_0\mathbf{I}_2 \end{bmatrix} \\ &= \underbrace{\begin{bmatrix} a_1\mathbf{I}_1 & 0 \\ 0 & a_0\mathbf{I}_2 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{L}_1 + \frac{b_1}{a_1}\mathbf{I}_1 & \mathbf{L}_3 \\ \mathbf{L}_3' & \mathbf{L}_2 + \frac{b_0}{a_0}\mathbf{I}_2 \end{bmatrix}}_{\mathbf{B}} \end{aligned} \tag{5.41}$$

Since matrix $\mathbf{A}$ is full-rank for $a_1, a_0 \neq 0$. Therefore $rank(\mathbf{T}_a) = rank(\mathbf{B})$. However, matrix $\mathbf{B}$ is the sum of Laplacian matrix $\mathbf{L}$ and matrix $\mathbf{E}$ given as:

$$\mathbf{B} = \mathcal{L} + \underbrace{\begin{bmatrix} (b_1/a_1)\mathbf{I}_1 & 0 \\ 0 & (b_0/a_0)\mathbf{I}_2 \end{bmatrix}}_{\mathbf{E}} \tag{5.42}$$

which is positive semi-definite (for $b_1/a_1, b_0/a_0 \geq 0$). Therefore $\mathbf{B}$ is positive semi-definite. Further for ($b_1 = 0$, $b_0 > 0$) the null-space $\mathcal{N}(L) = \{\mathbf{x} = c\mathbf{D}^{1/2}1 : c \in \mathbb{R}\}$ of matrix L and null-space $\mathcal{N}(E) = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{x}(k) = 0 \ \forall k \in \{m+1...N\}\}$ of matrix $\mathbf{E}$ are disjoint ($\forall \mathbf{x} \neq \mathbf{0}$). Hence $\mathbf{B}$ is positive definite and is therefore full rank. Same is true for the case when ($b_0 = 0$, $b_1 > 0$). Hence matrix $\mathbf{T}_a$ is full-rank and invertible. □

The advantage of the 1-hop localized design, described above, is that it can be applied to any arbitrary graph and for any partition sets L and H. However, the design is valid only for linear spectral kernels h_0 and h_1 with the constraints mentioned in Proposition 4, and does not hold for larger filters.

5.4.2 One-hop localized designs for bipartite graphs

The constraints required to satisfy invertibility in 1-hop localized filterbanks can be more relaxed, and easy to satisfy if the underlying graph is a bipartite graph. Let us consider the same 1-hop localized spectral filters on bipartite graphs. Referring again to (5.8), the filterbanks on a bipartite graph $\mathcal{B}$ with a pair of kernels $h_0(\lambda)$ and $h_1(\lambda)$ are invertible (i.e., have a solution of $g_0(\lambda)$ and $g_1(\lambda)$), if $det(\mathbf{H}_\lambda) \neq 0$ for all $\lambda \in \sigma(\mathcal{B})$. Using linear spectral kernels given in (5.37) in (5.8), we get:

$$det(\mathbf{H}_\lambda) = det\left(\begin{bmatrix} a_0\lambda + b_0 & a_1\lambda + b_1 \\ -a_0(2-\lambda) - b_0 & a_1(2-\lambda) + b_1 \end{bmatrix}\right) \tag{5.43}$$

This can be written as:

$$\begin{aligned} det(\mathbf{H}_\lambda) &= (a_0\lambda + b_0)(-a_1\lambda + 2a_1 + b_1) - (a_1\lambda + b_1)(a_0\lambda - 2a_0 - b_0) \\ &= -2(a_0a_1\lambda^2 - 2a_0a_1\lambda - a_1b_0 - a_0b_1 - b_1b_0) \\ &= -2a_0a_1\left(\lambda - (1-\delta_a)\right)\left(\lambda - (1+\delta_a)\right) \end{aligned} \tag{5.44}$$

where

$$\delta_a = \sqrt{1 + \frac{a_1 b_0 + a_0 b_1 + b_0 b_1}{a_0 a_1}} \tag{5.45}$$

Thus, $det(\mathbf{H}_\lambda)$ is a quadratic polynomial in terms of λ with roots at $1-\delta_a$ and $1+\delta_a$. Therefore, the filterbanks are invertible, as long as $1-\delta_a$ and $1+\delta_a$ are not the eigenvalues of $\mathcal{B}^2$. This is a much more relaxed condition than the conditions in Proposition 4, but works only in case of bipartite graphs. Let us take a specific design choice in which $a_1 = 1/2, a_0 = -1/2, b_1 = 0$ and $b_0 = 1$. The kernels $h_0(\lambda)$ and $h_1(\lambda)$ in this case, are plotted in Figure 5.2(a). Here $h_0(\lambda)$ can be considered a linear approximation of ideal lowpass kernel h_0^{ideal} in (5.30), and $h_1(\lambda) = h_0(2-\lambda)$, is the highpass kernel. The design does not satisfy the conditions in Proposition 4, since $(b_0/a_0) < 0$. However for bipartite graphs, applying these values in (5.45), we get:

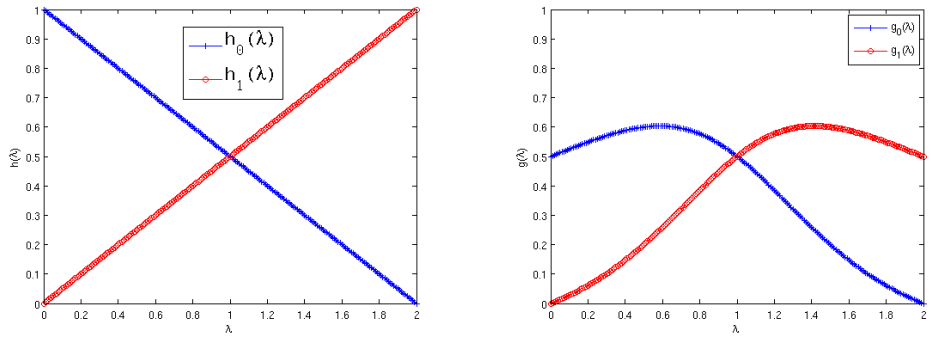
$$\delta_a = \sqrt{1 + \frac{a_1 b_0 + a_0 b_1 + b_0 b_1}{a_0 a_1}} = \sqrt{-1}, \tag{5.46}$$

which is imaginary. This implies that $det(\mathbf{H}_\lambda)$ does not have any real roots, i.e. $det(\mathbf{H}_\lambda) \neq 0$ for all $\lambda \in [0\ 2]$. Therefore, the filterbank is always invertible on any bipartite graph. Solving (5.8) by applying the specific parameters we get:

$$\begin{aligned} h_0(\lambda) &= 1 - \frac{\lambda}{2} \\ h_1(\lambda) &= \frac{\lambda}{2} \\ g_0(\lambda) &= \frac{1 - \frac{1}{2}\lambda}{\lambda^2 - 2\lambda + 2} \\ g_1(\lambda) &= \frac{1}{2}\frac{\lambda}{\lambda^2 - 2\lambda + 2} \end{aligned} \tag{5.47}$$

To summarize this section, we have proposed 1-hop localized filterbanks on graphs which are invertible and critically sampled. The conditions for invertibility on any arbitrary graph are given in Proposition 4. These conditions are quite constrained. On the other hand, we presented analysis of 1-hop localized filterbanks for bipartite graphs and showed that the invertibility conditions in bipartite graphs are much more relaxed, and easy to find. One example of 1-hop localized filterbanks is proposed in (5.47), which is perfect reconstruction on any bipartite graphs and for all graph-signals. However, the synthesis kernels in the proposed design are not polynomials,

[2] As stated in Lemma 2, the eigenvalues of a bipartite graph occur symmetrically around $\lambda = 1$, i.e., if $1+\delta_a$ is an eigenvalue of $\mathcal{B}$ then $1-\delta_a$ is also an eigenvalue of $\mathcal{B}$.



(a) analysis kernels (b) synthesis kernels

Figure 5.2: Proposed 1-hop spectral kernels for bipartite graphs.

which implies that they do not have compact support. In the next section, we consider designs where both analysis and synthesis kernels have compact support.

5.5 Graph-Bior Filterbanks

We now revisit the perfect reconstruction conditions on bipartite graphs, described in Section 5.2 to design critically sampled perfect reconstruction filterbanks, in which both analysis and synthesis filters are compactly supported. In our proposed designs, both analysis filters $\mathbf{H}_i$ and synthesis filters $\mathbf{G}_i$ for $i = 0, 1$, of the two channels are graph transforms characterized by spectral kernels $h_i(\lambda)$ and $g_i(\lambda)$ for $i = 0, 1$ respectively. The minimum constraints for designing biorthogonal filters are to satisfy the perfect reconstruction conditions given in (5.15), which involves designing four spectral kernels, namely lowpass analysis kernel $h_0(\lambda)$, highpass analysis kernel $h_1(\lambda)$, lowpass synthesis kernel $g_0(\lambda)$, and highpass synthesis kernel $g_1(\lambda)$. If we choose analysis and synthesis high-pass kernels to be:

$$\begin{aligned} h_1(\lambda) &= g_0(2-\lambda) \\ g_1(\lambda) &= h_0(2-\lambda), \end{aligned} \tag{5.48}$$

then, (5.15) reduces to a single constraint for all eigenvalues, given as:

$$h_0(\lambda)g_0(\lambda) + h_0(2-\lambda)g_0(2-\lambda) = 2. \tag{5.49}$$

Further, define $p(\lambda) = h_0(\lambda)g_0(\lambda)$, then (5.49) can be written as:

$$p(\lambda) + p(2 - \lambda) = 2. \tag{5.50}$$

In our approach, we first design $h_0(\lambda)$ and $g_0(\lambda)$, and then $h_1(\lambda)$ and $g_1(\lambda)$ can be obtained using (5.48). Further, since $p(\lambda)$ is the product of two low-pass kernels, it is also a low-pass kernel. Therefore, the objective is to design $p(\lambda)$ as a polynomial *half-band kernel*[3], which satisfies (5.49), and then obtain kernels $h_0(\lambda)$ and $h_1(\lambda)$ via spectral factorization. The following result is useful in our analysis:

Proposition 5. *If $h_0(\lambda)$ and $g_0(\lambda)$ are polynomial kernels, then any $p(\lambda) = h_0(\lambda)g_0(\lambda)$, which satisfies (5.48) for all $\lambda \in [0\ 2]$, is an odd degree polynomial.*

Proof. By changing the variable λ to $1 - \lambda$, we can write (5.48) as:

$$p(1 + \lambda) + p(1 - \lambda) = 2, \tag{5.51}$$

where $p(1+\lambda) = h_0(1+\lambda)g_0(1+\lambda)$. If $h_0(\lambda)$ and $g_0(\lambda)$ are polynomial kernels, then the functions $p(1 + \lambda)$ and $p(1 - \lambda)$ are also polynomials and can be expressed as:

$$\begin{aligned} p(1 + \lambda) &= \sum_{k=0}^{K} c_k(\lambda)^k. \\ p(1 - \lambda) &= \sum_{k=0}^{K} c_k(-\lambda)^k. \end{aligned} \tag{5.52}$$

Using (5.52) in (5.51), we get:

$$p(1 + \lambda) + p(1 - \lambda) = \sum_{k=0}^{K} c_k((\lambda)^k + (-\lambda)^k) = 2c_0 + \sum_{k=1}^{K/2} c_{2k}\lambda^{2k}. \tag{5.53}$$

Thus $p(1 + \lambda) + p(1 - \lambda)$ is an even polynomial function of λ. However, we need to design $p(\lambda)$ such that $p(1 + \lambda) + p(1 - \lambda) = 2$ for all $\lambda \in [-1\ 1]$, which is true *if and only if*, $c_0 = 1$ and all

[3] An half band kernel $h(\lambda)$ in the spectral domain of a graph can be defined as a kernel with $h(\lambda) \approx 1$ for $\lambda \leq 1$ (i.e., less than Nyquist frequency.), and $h(\lambda) \approx 0$ otherwise. Examples of half-band kernels are ideal spectral kernel in (5.30), and Meyer's kernel in (5.33).

other even power coefficients c_n in the polynomial expansion of $p(1+\lambda)$ are 0. Therefore, the solution $p(1+\lambda)$, expressed as:

$$p(1+\lambda) = 1 + \sum_{n=0}^{K} c_{2n+1}\lambda^{2n+1}, \tag{5.54}$$

is an odd degree polynomial. Thus, ignoring the trivial case $p(1+\lambda) = 1$, the highest degree of $p(1+\lambda)$ (and hence $p(\lambda)$) is always odd. □

5.5.1 Designing half-band kernel $p(\lambda)$

While the graph-QMF filters cannot be exact polynomials, there exist non-orthogonal filters that satisfy (5.48), and (5.49). The following known results help us prove the existence of a polynomial $p(\lambda)$ that satisfy (5.49), and its spectral factorization:

Lemma 4 (Bezout's identity [44, prop. 3.13]). *Given any two polynomials $a(\lambda)$ and $b(\lambda)$,*

$$a(\lambda)x(\lambda) + b(\lambda)y(\lambda) = c(\lambda), \tag{5.55}$$

has a solution $[x(\lambda),\ y(\lambda)]$, if and only if $gcd(a(\lambda), b(\lambda))$ divides $c(\lambda)$, where $gcd(a(\lambda), b(\lambda))$ refers to the greatest common divisor of polynomials $a(\lambda)$ and $b(\lambda)$.

Theorem 2 (Complementary Filters [44, prop. 3.13]). *Given a polynomial kernel $h_0(\lambda)$, there exists a complementary polynomial kernel $g_0(\lambda)$ which satisfies the perfect reconstruction relation in (5.49), if and only if $h_0(1+\lambda)$ and $h_0(1-\lambda)$ are coprime.*

Proof. Let us denote $a(\lambda) = h_0(1+\lambda)$, $b(\lambda) = h_0(1-\lambda)$, $x(\lambda) = g_0(1+\lambda)$, $y(\lambda) = g_0(1-\lambda)$ and $c(\lambda) = 2$. Then, (5.49) can be written in the same form as (5.55). Following the result of Lemma 4, given polynomial kernel $h_0(\lambda)$, a polynomial solution of $g_0(\lambda)$ exists if and only if $gcd(h_0(1+\lambda), h_0(1-\lambda))$ divides $c(\lambda) = 2$, which is a prime number. This implies $gcd(h_0(1+\lambda), h_0(1-\lambda))$ is either 1 or 2 for all $\lambda \in [-1\ 1]$, which is true iff $h_0(1+\lambda)$ and $h_0(1-\lambda)$ do not have any common roots. This implies that $h_0(1+\lambda)$ and $h_0(1-\lambda)$ are coprime. □

Corollary 1 ([44, exercise. 3.12]). *There is always a complementary filter for the polynomial kernel $(1+\lambda)^k$, i.e.,*

$$(1+\lambda)^k R(\lambda) + (1-\lambda)^k R(-\lambda) = 2 \tag{5.56}$$

always has a real polynomial solution $R(\lambda)$ for $k \geq 0$.

Proof. Let us denote $a(\lambda) = (1+\lambda)^k$, $b(\lambda) = (1-\lambda)^k$, $x(\lambda) = R(\lambda)$, $y(\lambda) = R(-\lambda)$ and $c(\lambda) = 2$. Then, (5.56) can be written in the same form as (5.55). Since $a(\lambda)$ and $b(\lambda)$, in this case are coprime, therefore $gcd(a(\lambda), b(\lambda)) = 1$ divides $c(\lambda) = 2$. Hence, a polynomial $R(\lambda)$, which satisfies (5.56) always exists. □

For a perfect reconstruction biorthogonal filterbank, we need to design a polynomial half-band kernel $p(\lambda)$ that satisfies (5.49), or equivalently (5.51). Following Daubechies' approach [5], we propose a *maximally-flat* design, in which we assign K roots to $p(\lambda)$ at the lowest eigenvalue (i.e., at $\lambda = 0$). Subsequently, we select $p(\lambda)$ to be the shortest length polynomial, which has K roots at $\lambda = 0$ and satisfies (5.51). This implies that $p(1+\lambda)$ has K roots at $\lambda = -1$, and can be expanded as:

$$p(1+\lambda) = (1+\lambda)^K \underbrace{\sum_{m=0}^{k} r_m \lambda^m}_{R(\lambda)}. \tag{5.57}$$

where $R(\lambda)$ is the residual k degree polynomial. By Corollary 1, there always exist such a polynomial $R(\lambda)$. On the other hand, Proposition 5 says that any $p(1+\lambda)$ that satisfies (5.51) has to be an odd-degree polynomial. Hence, $p(1+\lambda)$ can also be expanded as:

$$p(1+\lambda) = 1 + \sum_{n=0}^{M} c_{2n+1} \lambda^{2n+1}. \tag{5.58}$$

for a given M. Comparing (5.57) and (5.58), we get:

$$(1+\lambda)^K \sum_{m=0}^{k} r_m \lambda^m = 1 + \sum_{n=0}^{M} c_{2n+1} \lambda^{2n+1}. \tag{5.59}$$

Comparing the constant terms in the left and right side of (5.59), we get $r_0 = 1$. Further, comparing the highest powers on both sides of (5.59) we get:

$$M = \frac{K+k-1}{2} \tag{5.60}$$

Further, the right side in (5.59) has M constraints $c_{2n} = 0$ for $n = \{1, 2, ...K\}$, and the left side

in (5.59) has k unknowns r_m for $m = \{1, 2, ...k\}$. In order to get a unique $p(1+\lambda)$ that satisfies (5.51), we must have equal number of unknowns and constraints, i.e,

$$M = k = \frac{K + k - 1}{2} \quad \Rightarrow \quad M = K - 1. \tag{5.61}$$

Thus, (5.59) can be written as:

$$(1+\lambda)^K (1 + \sum_{m=1}^{K-1} r_m \lambda^m) = 1 + \sum_{n=0}^{K-1} c_{2n+1} \lambda^{2n+1}, \tag{5.62}$$

and $K-1$ unknowns can be found uniquely, by solving a linear system of $K-1$ equations. Note that given K, the length of $p(\lambda)$ (i.e, highest degree) is $K + M = 2K - 1$. As an example, we design $p(\lambda)$ with $K = 2$ zeros at $\lambda = 0$. In this case $p(1+\lambda)$ can be written as:

$$p(1+\lambda) = (1+\lambda)^2 (1 + r_1 \lambda) = 1 + (r_1 + 2)\lambda + (1 + 2r_1)\lambda^2 + r_1 \lambda^3$$

Since $p(1+\lambda)$ is an odd polynomial, the term corresponding to λ^2 is zero, i.e., $1 + 2r_1 = 0$ or $r_1 = -1/2$. Therefore, $p(1+\lambda)$ is given as:

$$p(1+\lambda) \quad = \quad (1+\lambda)^2 (1 - \frac{1}{2}\lambda). \tag{5.63}$$

which implies that:

$$p(\lambda) \quad = \quad \frac{1}{2}\lambda^2 (3 - \lambda). \tag{5.64}$$

We plot in Figure 5.3 $p(\lambda)$ for various values of K, and it can be seen that by increasing K, we get better ideal half band filter approximation of $p(\lambda)$.

Note that for graph-QMF designs $h_1(\lambda) = h_0(2 - \lambda)$, hence $p(\lambda) = h_0^2(\lambda)$ is a perfect square of a polynomial. While graph-QMF designs satisfy constraints given in both (5.48) and (5.49), they cannot be designed as exact polynomials. As proven in Proposition 5, any $p(\lambda)$ which satisfies (5.49) is an odd-degree polynomial. Using this result, we can prove the following:

Proposition 6. *The kernels in the graph-QMF design cannot be exact polynomial functions.*

Proof. For the graph-QMF solution in (5.28), the design was based on selecting $g_0(\lambda) = h_0(\lambda)$

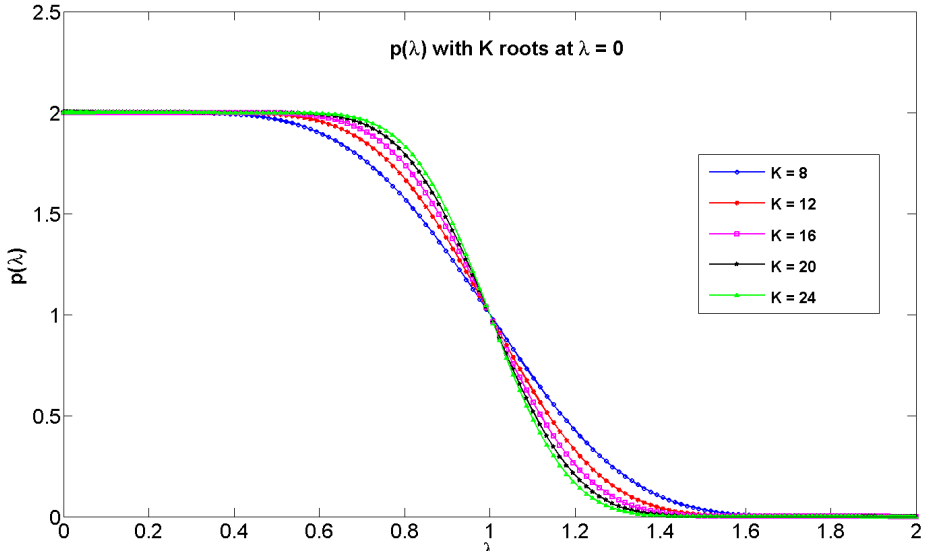


Figure 5.3: The spectral distribution of $p(\lambda)$ with K zeros at $\lambda = 0$

and we can write $p_{QMF}(\lambda) = h_0^2(\lambda)$. Thus, if $h_0(\lambda)$ is a polynomial kernel then $p_{QMF}(\lambda)$ is the square of a polynomial, and therefore should have an even degree. However as proved in Proposition 5 above, $p_{QMF}(\lambda)$ is an odd degree polynomial and cannot be factored into the square of a polynomial. Therefore, $h_0(\lambda)$ in the graph-QMF designs, cannot be an exact polynomial. □

5.5.2 Spectral factorization of half-band kernel $p(\lambda)$

Once we obtain $p(\lambda)$ by using above mentioned design, we need to factorize it into filter kernels $h_0(\lambda)$ and $g_0(\lambda)$. Since $p(\lambda)$ is a real polynomial of odd degree, it has at least one real root and all the complex roots occur in conjugate pairs. Since we want the two kernels to be polynomials with real coefficients, each complex conjugate root pair of $p(\lambda)$ should be assigned together to either $h_0(\lambda)$ or $g_0(\lambda)$. While any such factorization would lead to perfect reconstruction biorthogonal filterbanks, of particular interest is the design of filterbanks that are as close to orthogonal as possible. For this, we define a criteria based on energy preservation. In particular, we compute the Riesz bounds of analysis wavelet transform $\mathbf{T}_a$, which are the tightest lower and upper bounds, $A > 0$ and $B < \infty$, of $||\mathbf{T}_a\mathbf{f}||_2$, for any graph-signal $\mathbf{f}$ with $||\mathbf{f}||_2 = 1$. For near-orthogonality, we require $A \approx B \approx 1$. The analysis transform $\mathbf{T}_a$ can be expressed in terms of transforms $\mathbf{H}_0$

and $\mathbf{H}_1$ as in (5.39), and the Riesz bounds can be computed as the square-roots of the extreme eigenvalues of $\mathbf{T}_a^t\mathbf{T}_a$. By expanding $\mathbf{T}_a^t\mathbf{T}_a$, using (5.39) and (5.1) we obtain:

$$\begin{aligned}\mathbf{T}_a^t\mathbf{T}_a &= 1/2\sum_{\lambda\in\sigma(\mathcal{B})}\underbrace{(h_0^2(\lambda)+h_1^2(\lambda))}_{C_\lambda}\mathbf{P}_\lambda \\ &+ 1/2\sum_{\lambda\in\sigma(\mathcal{B})}\underbrace{(h_1(\lambda)h_1(2-\lambda)-h_0(\lambda)h_0(2-\lambda))}_{D_\lambda}\mathbf{J}_\beta\mathbf{P}_\lambda \end{aligned} \tag{5.65}$$

In (5.65), the term $D(\lambda)$ consists of product terms $h_0(\lambda)h_0(2-\lambda)$ and $h_1(\lambda)h_1(2-\lambda)$, which are small for λ away from 1 (since it is the product of a low pass and a high pass kernel), and approximately cancel out each other for λ close to 1 (see Figure 5.4). Therefore, we can ignore $D(\lambda)$ in comparison to $C(\lambda)$, and (5.65) can be approximately reduced to (5.66).

$$\mathbf{T}_a^t\mathbf{T}_a \approx 1/2\sum_{\lambda\in\sigma(\mathcal{B})}\underbrace{(h_0^2(\lambda)+h_1^2(\lambda))}_{C_\lambda}\mathbf{P}_\lambda \tag{5.66}$$

Thus, $\mathbf{T}_a^t\mathbf{T}_a$ is a spectral transform with eigenvalues $1/2(h_0^2(\lambda)+h_1^2(\lambda))$ for $\lambda\in\sigma(\mathcal{B})$, and the Riesz Bounds can be given as:

$$\begin{aligned}A &= \inf_\lambda \frac{1}{2}(h_0^2(\lambda)+h_1^2(\lambda)) \\ B &= \sup_\lambda \frac{1}{2}(h_0^2(\lambda)+h_1^2(\lambda))\end{aligned} \tag{5.67}$$

We choose the $\Delta = A/B$, as the measure of orthogonality (for orthogonal filterbanks $\Delta = 1$). The exact computation of Δ requires all eigenvalues of the graph. In general, the eigenvalues can be computed from the eigen-decomposition of the graph, if the graph is known. However, this incurs additional computational complexity, something we have avoided so far in our designs. Therefore, we compute an approximte Δ as the difference of lowest and highest values of $1/2(h_0^2(\lambda)+h_1^2(\lambda))$ at 100 uniformly sampled points from the continuous region [0 2]. We choose filters with least dissimilar lengths, and compute Δ for all such possible factorizations (which are $\binom{2K-1}{K}$ in number). Finally, we choose the factorization with the maximum magnitude Δ.

5.5.3 Nomenclature and design of graph-Bior filterbanks

The proposed biorthogonal filterbanks are specified by four parameters ($k_0,\ k_1,\ l_0,\ l_1$), where k_0 is the number of roots of low pass analysis kernel $h_0(\lambda)$ at $\lambda = 0$, k_1 is the number of roots of low pass synthesis kernel $g_0(\lambda)$ at $\lambda = 0$, l_0 is the highest degree of low pass analysis kernel

$h_0(\lambda)$, and l_1 is the highest degree of low pass synthesis kernel $g_0(\lambda)$, respectively. The other two filters, namely $h_1(\lambda)$ and $g_1(\lambda)$ can be computed as in (5.48). Given these specifications, we design $p(\lambda) = h_0(\lambda)g_0(\lambda)$ as a maximally flat half band polynomial kernel with $K = k_0 + k_1$ number of roots at $\lambda = 0$. As a result, $p(\lambda)$ turns out to be a $2K - 1$ degree polynomial, and we factorize it into $h_0(\lambda)$ and $g_0(\lambda)$, with least dissimilar lengths (i.e., we choose $l_0 = K$ and $l_1 = K - 1$). We use Δ to be the criteria to compare various possible factorizations, and choose the one with the maximum value of Δ. This leads to a unique design of biothogonal filterbanks. We term our proposed filterbanks as *graphBior*(k_0, k_1). We designed graphBior filterbanks for various values of (k_0, k_1), and we observed that designs with $k_0 = k_1$ stand out, as they are close to orthogonal and have near-flat pass-band responses. The low-pass and high-pass analysis kernels are plotted in Figure 5.4, and their coefficients are shown in Table 5.2. A comparison between proposed graph-Bior filterbanks and proposed graph-QMF filterbanks, in terms of perfect reconstruction error (SNR) and orthogonality (Δ) is shown in Table 5.1. The reconstruction SNR and orthogonality Δ are computed as an average over 20 instances of randomly generated graph-signals on 10 random bipartite graphs with 100 nodes each. In this table, the filter length of graph-Bior designs is chosen to be the maximum of the two filter lengths (i.e, K). It can be seen from Table 5.1 that all graph-Bior designs provide perfect reconstruction ($SNR > 100dB$). The graph-QMF filters in comparison are more orthogonal (i.e., Δ closer to 1), but have considerably lower reconstruction SNR. In Table 5.1, the value of Δ for graph-Bior filterbanks seem far off from 1. However, this is primarily because the lowpass and highpass kernels are not symmetric or even equal length. As a result, the basis in the biorthogonal case (i.e., rows of analysis transform $\mathbf{T}_a$) corresponding to different kernels have different norms, and therefore the signal projected on to different basis experiences different gains, leading to $\Delta << 1$. In order to see the orthogonality of the basis in a practical case, we empirically compute the *mutual coherence* M of the transform matrix $\mathbf{T}_a$, which is defined as the maximum absolute value of the cross-correlation between the rows of $\mathbf{T}_a$, i.e.,

$$M = \max_{1 \leq i \neq j \leq N} | < \mathbf{t}_{ai} \ \mathbf{t}_{aj} > |, \tag{5.68}$$

and for orthogonal basis, $M = 0$. We observe in Table 5.1 that the value of mutual coherence M is very close to 0 for biorthogonal filters, which we compute as the average mutual coherence over 10 instances of random bipartite graphs. This implies that the basis in $\mathbf{T}_a$ are nearly orthogonal.

In order to get a uniform gain over all basis, we need to adjust the gains of lowpass and highpass basis in $\mathbf{T}_a$, as done in the standard biorthogonal filterbanks. This is a part of our future work.

Filterlength	**Graph-QMF**		**Graph-Bior**		
	SNR (in dB)	Δ	SNR (in dB)	Δ	M
4	32.2107	0.9657	300.3649	0.5042	0.0946
10	42.1521	0.9856	245.691	0.5777	0.0634
14	48.0403	0.9902	173.5528	0.6528	0.0518
16	44.6852	0.9854	154.7572	0.6802	0.0445
18	45.1262	0.9876	123.9701	0.6864	0.0423
20	54.6041	0.9963	115.0719	0.7112	0.0378

Table 5.1: Comparison between graph-QMF filterbanks and graph-Bior filterbanks

graphBior(k_0,k_1)	**filter coefficients**
$k_0 = 6$, $k_1 = 6$	h_1 = [-0.3864 4.0351 -17.0630 36.5763 -39.8098 17.6477 0 0 0 0 0 0] h_0 = [0.4352 -4.9802 23.2396 -55.4662 67.2657 -29.0402 -13.0400 7.5253 9.5267 -4.8746 -2.0616 1.2633 1.2071]
$k_0 = 7$, $k_1 = 7$	h_1 = [0.3115 -3.9523 21.0540 -60.3094 98.0605 -85.9222 31.7578 0 0 0 0 0 0 0] h_0 = [-0.4975 6.8084 -39.6151 126.2423 -234.3683 241.5031 -97.6557 -46.2635 62.1232 -19.3648 -2.0766 6.5886 -4.5632 0.5775 1.5614]
$k_0 = 8$, $k_1 = 8$	h_1 = [-0.3232 4.7284 -29.7443 104.3985 -221.0705 282.7915 -202.6283 62.8477 0 0 0 0 0 0 0 0] h_0 = [0.4470 -6.9872 47.5460 -183.6940 440.0924 -670.0905 643.3979 -396.0713 209.9824 -154.0976 92.8617 -30.8228 16.6112 -12.7664 3.2403 -0.0284 1.3793]

Table 5.2: Polynomial expansion coefficients (highest degree first) of graphBior (k_0, k_1) filters (approximated to 4 decimal places) on a bipartite graph.

5.6 Filterbank designs using asymmetric Laplacian matrix

A DC signal on a graph corresponds to a scalar multiple of the eigenvector of graph Laplacian matrix corresponding to the lowest eigenvalue (i.e., $\lambda = 0$). So far in this chapter, we have used the symmetric normalized Laplacian matrix $\mathcal{L}$ to design spectral filters, in which case, a DC vector on a bipartite graph is of the form $c\mathbf{D}^{1/2}\mathbf{f} = c\mathbf{1}$, where $\mathbf{1}$ is a vector with all elements 1, and $\mathbf{D}$ is the degree matrix. As described in Section 4.3, the normalization is necessary in order to extend the downsampling results for k-RBG to other non-regular bipartite graphs. Further, the matrix $I - \mathcal{L}$ has the same eigenvalues as the probability transition matrix $\mathbf{D}^{-1}\mathbf{A}$ of a random

walk defined on the graph, and thus is consistent with the stochastic properties of the graph. However, in some applications, such as image-processing, a DC signal is defined as an all constant signal $\mathbf{f} = c\mathbf{1}$, and a desired property of wavelet filters in this case is to have zero response (i.e., the wavelet coefficients are all zero) corresponding to $\mathbf{f} = c\mathbf{1}$. In order to make our filterbanks compatible with these applications, we propose designing spectral filters using the asymmetric Laplacian matrix $\mathcal{L}_a$, which is defined as:

$$\mathcal{L}_a = \mathbf{D}^{-1}\mathbf{L} = \mathbf{D}^{-1/2}\mathcal{L}\mathbf{D}^{1/2} \tag{5.69}$$

Since $\mathbf{1}$ is an eigenvector of $\mathcal{L}$ with eigenvalue $\lambda = 0$, it is also an eigenvector of $\mathcal{L}_a$ with eigenvalue $\lambda = 0$. Further, matrix $\mathcal{L}_a$ is *similar* to $\mathcal{L}$, and therefore has the same set of eigenvalues as $\mathcal{L}$. The eigenvector $\mathbf{u}_{\lambda,a}$ of $\mathcal{L}_a$ is related to eigenvector $\mathbf{u}_\lambda$ of $\mathcal{L}$ as:

$$\mathbf{u}_{\lambda,a} \quad = \quad \mathbf{D}^{-1/2}\mathbf{u}_\lambda \tag{5.70}$$

Note that for non-regular graphs $\mathcal{L}_a$ is an asymmetric matrix, therefore the eigenvectors of $\mathcal{L}_a$ are not orthogonal. The eigenvector decomposition of $\mathcal{L}_a$ is given as :

$$\mathcal{L}_a = (\mathbf{D}^{-1/2}\mathbf{U})\mathbf{\Lambda}(\mathbf{D}^{-1/2}\mathbf{U})^{-1}. \tag{5.71}$$

Therefore, similar to (5.1), a spectral filters using $\mathcal{L}_a$, corresponding to spectral kernel $h(\lambda)$ can be defined as:

$$\begin{aligned} \mathbf{H}_a = h(\mathcal{L}_a) &= (\mathbf{D}^{-1/2}\mathbf{U})h(\mathbf{\Lambda})(\mathbf{D}^{-1/2}\mathbf{U})^{-1} \\ &= \mathbf{D}^{-1/2}\mathbf{U}h(\mathbf{\Lambda})\mathbf{U}^t\mathbf{D}^{1/2} \\ &= \mathbf{D}^{-1/2}\mathbf{H}\mathbf{D}^{1/2}, \end{aligned} \tag{5.72}$$

where $\mathbf{H}$ is the spectral filter with the same spectral kernel $h(\lambda)$, defined using normalized Laplacian matrix $\mathcal{L}$. To avoid confusion, we will refer to filters designed using asymmetric Laplacian matrix as simply *asymmetric filters*, and filters designed using symmetric Laplacian matrix as *symmetric filters*. According to (5.72) , any *asymmetric filters* $\mathbf{H}_a$ is similar to a *symmetric filters* $\mathbf{H}$ with same spectral kernel. However, the advantage of using $\mathbf{H}_a$ instead of $\mathbf{H}$ in a filterbank

is that $\mathbf{1}$ is an eigenvector of $\mathbf{H}_a$ with eigenvalue $h(0)$. Referring again to Figure 2.1, the overall transfer function of an asymmetric filterbank can be written as:

$$\begin{aligned}\hat{\mathbf{f}} &= \frac{1}{2}\mathbf{G}_{0a}(\mathbf{I}+\mathbf{J}_\beta)\mathbf{H}_{0a}\mathbf{f} + \frac{1}{2}\mathbf{G}_{1a}(\mathbf{I}-\mathbf{J}_\beta)\mathbf{H}_{1a}\mathbf{f} \\ &= \frac{1}{2}(\mathbf{G}_{0a}\mathbf{H}_{0a}+\mathbf{G}_{1a}\mathbf{H}_{1a})\mathbf{f} + \frac{1}{2}(\mathbf{G}_{0a}\mathbf{J}_\beta\mathbf{H}_{0a}-\mathbf{G}_{1a}\mathbf{J}_\beta\mathbf{H}_{1a})\mathbf{f}. \end{aligned} \tag{5.73}$$

Using the similarity relation given in (5.72), we can simplify (5.73) as:

$$\begin{aligned}\hat{\mathbf{f}} &= \frac{1}{2}(\mathbf{D}^{-1/2}\mathbf{G}_0\mathbf{D}^{1/2}\mathbf{D}^{-1/2}\mathbf{H}_0\mathbf{D}^{1/2} + \mathbf{D}^{-1/2}\mathbf{G}_1\mathbf{D}^{1/2}\mathbf{D}^{-1/2}\mathbf{H}_1\mathbf{D}^{1/2})\mathbf{f} \\ &+ \frac{1}{2}(\mathbf{D}^{-1/2}\mathbf{G}_0\mathbf{D}^{1/2}\mathbf{J}_\beta\mathbf{D}^{-1/2}\mathbf{H}_0\mathbf{D}^{1/2} - \mathbf{D}^{-1/2}\mathbf{G}_1\mathbf{D}^{1/2}\mathbf{J}_\beta\mathbf{D}^{-1/2}\mathbf{H}_1\mathbf{D}^{1/2})\mathbf{f}. \end{aligned} \tag{5.74}$$

In (5.74), the matrices $\mathbf{D}^{1/2}, \mathbf{J}_\beta$, and $\mathbf{D}^{-1/2}$ are diagonal matrices and hence commute with each other. Therefore,

$$\mathbf{D}^{1/2}\mathbf{J}_\beta\mathbf{D}^{-1/2} = \mathbf{J}_\beta\mathbf{D}^{1/2}\mathbf{D}^{-1/2} = \mathbf{J}_\beta \tag{5.75}$$

Thus, (5.74), can be simplified as:

$$\begin{aligned}\hat{\mathbf{f}} &= \frac{1}{2}(\mathbf{D}^{-1/2}\mathbf{G}_0\mathbf{H}_0\mathbf{D}^{1/2} + \mathbf{D}^{-1/2}\mathbf{G}_1\mathbf{H}_1\mathbf{D}^{1/2})\mathbf{f} \\ &+ \frac{1}{2}(\mathbf{D}^{-1/2}\mathbf{G}_0\mathbf{J}_\beta\mathbf{H}_0\mathbf{D}^{1/2} - \mathbf{D}^{-1/2}\mathbf{G}_1\mathbf{J}_\beta\mathbf{H}_1\mathbf{D}^{1/2})\mathbf{f} \\ &= \mathbf{D}^{-1/2}\mathbf{T}_{eq}\mathbf{D}^{1/2}\mathbf{f} + \mathbf{D}^{-1/2}\mathbf{T}_{alias}\mathbf{D}^{1/2}\mathbf{f} \\ &= \mathbf{D}^{-1/2}(\mathbf{T}_{eq}+\mathbf{T}_{alias})\mathbf{D}^{1/2}\mathbf{f}, \end{aligned} \tag{5.76}$$

where $\mathbf{T}_{eq}$ and $\mathbf{T}_{alias}$ correspond to the overall transfer function of symmetric filterbanks, as defined in (5.2) and (5.3), respectively. **Thus, the asymmetric filterbanks are equivalent to symmetric filterbanks designed with same spectral kernels, in which the input is normalized with $\mathbf{D}^{1/2}$ prior to filtering/downsampling operations, and the output is de-normalized with $\mathbf{D}^{-1/2}$ after the filtering/downsampling operations.** We use this result to find out the perfect reconstruction conditions and orthogonality in asymmetric filterbanks.

5.6.1 Perfect Reconstruction

If $\mathbf{T}_{eq} + \mathbf{T}_{alias} = c\mathbf{I}$, then $\hat{\mathbf{f}} = \mathbf{D}^{-1/2}(c\mathbf{I})\mathbf{D}^{1/2}\mathbf{f} = c\mathbf{f}$ in (5.76), therefore the *asymmetric filterbanks* are perfect reconstruction if the *symmetric filterbanks* designed using the same spectral kernels, are perfect reconstruction. As a result, the conditions mentioned in (5.15), are also necessary and sufficient conditions for perfect reconstruction in *asymmetric filterbanks*.

5.6.2 Orthogonality

Since the eigenvectors of $\mathcal{L}_a$ are not orthogonal, the asymmetric filterbanks using graph-QMF kernels are also not orthogonal. The following analysis explains the frame property of asymmetric filterbanks.

Similar to (5.16), the wavelet coefficient vector $\mathbf{w}$ produced in the asymmetric filterbanks can be written as:

$$\begin{aligned} \mathbf{w} = \mathbf{T}_{aa}\mathbf{f} &= \frac{1}{2}(\mathbf{H}_{1a} + \mathbf{H}_{0a})\mathbf{f} + \frac{1}{2}\mathbf{J}_{\beta}(\mathbf{H}_{1a} - \mathbf{H}_{0a})\mathbf{f} \\ &= \frac{1}{2}\mathbf{D}^{-1/2}(\mathbf{H}_1 + \mathbf{H}_0)\mathbf{D}^{1/2}\mathbf{f} + \frac{1}{2}\mathbf{D}^{-1/2}\mathbf{J}_{\beta}(\mathbf{H}_1 - \mathbf{H}_0)\mathbf{D}^{1/2}\mathbf{f} \\ &= \mathbf{D}^{-1/2}\mathbf{T}_a\mathbf{D}^{1/2} \end{aligned} \tag{5.77}$$

In (5.77), if we define $\mathbf{f}_n = \mathbf{D}^{1/2}\mathbf{f}$, and $\mathbf{w}_n = \mathbf{D}^{1/2}\mathbf{w}$, then (5.77) can be written as $\mathbf{w}_n = \mathbf{T}_a\mathbf{f}_n$. Thus, if the corresponding symmetric filterbank is orthogonal, i.e., if the spectral kernels satisfy, (5.27), then $||\mathbf{w}_n|| = ||\mathbf{f}_n||$ (the 2-norm). However,

$$\begin{aligned} d_{min}\sum_{i=1}^{N} w^2(i) \leq ||\mathbf{w}_n||^2 &= \sum_{i=1}^{N} d_i w^2(i) \leq d_{max}\sum_{i=1}^{N} w^2(i) \\ d_{min}\sum_{i=1}^{N} f^2(i) \leq ||\mathbf{f}_n||^2 &= \sum_{i=1}^{N} d_i f^2(i) \leq d_{max}\sum_{i=1}^{N} f^2(i), \end{aligned} \tag{5.78}$$

where d_{min} is the minimum degree in the graph (1 if there is an isolated node), and d_{max} is the maximum degree. Using (5.78), we obtain:

$$\begin{aligned} d_{min}||\mathbf{w}||^2 \leq ||\mathbf{w}_n||^2 &= ||\mathbf{f}_n||^2 \leq d_{max}||\mathbf{f}||^2 \\ d_{min}||\mathbf{f}||^2 \leq ||\mathbf{f}_n||^2 &= ||\mathbf{w}_n||^2 \leq d_{max}||\mathbf{w}||^2, \end{aligned} \tag{5.79}$$

and

$$\frac{d_{min}}{d_{max}}||\mathbf{f}||^2 \leq ||\mathbf{w}||^2 \leq \frac{d_{max}}{d_{min}}||\mathbf{f}||^2 \tag{5.80}$$

Thus, the asymmetric graph-QMF filterbanks define a frame in the graph-signal space, with lower bound $A = \sqrt{d_{min}/d_{max}}$ and upper-bound $B = \sqrt{d_{max}/d_{min}}$. Note that for regular graphs $d_{min} = d_{max}$, hence $A = B = 1$, and the asymmetric graph-QMF filterbanks are orthogonal. Similar analysis can be done for asymmetric graphBior filterbanks.

Thus, we can use both symmetric normalized and asymmetric normalized Laplacian matrices to design spectral filters in any of our proposed filterbank designs. The decision about which filterbank design and which Laplacian matrix to choose depends on the desired properties of the filterbanks. In Table 5.3, we present a comparison of all of our proposed designs. Note that all of these filterbanks are designed for bipartite graphs. The extension of these designs to any arbitrary graph is presented in the Chapter 6.

Method	**Laplacian matrix**	**DC**	**CS**	**PR**	**Comp**	**OE**
Graph-QMF (exact)	symmetric[4]	$\mathbf{f} = c\mathbf{D}^{-1/2}\mathbf{1}$	Yes	Yes	No	Yes
	asymmetric[5]	$\mathbf{f} = c\mathbf{1}$	Yes	Yes	No	No
Graph-QMF (approx.)	symmetric[4]	$\mathbf{f} = c\mathbf{D}^{-1/2}\mathbf{1}$	Yes	No[7]	Yes	No
	asymmetric[5]	$\mathbf{f} = c\mathbf{1}$	Yes	No	Yes	No
One-hop localized	symmetric[4]	$\mathbf{f} = c\mathbf{D}^{-1/2}\mathbf{1}$	Yes	Yes	Yes[6]	No
	asymmetric[5]	$\mathbf{f} = c\mathbf{1}$	Yes	Yes	Yes[6]	No
Graph-Bior	symmetric[4]	$\mathbf{f} = c\mathbf{D}^{-1/2}\mathbf{1}$	Yes	Yes	Yes	No
	asymmetric[5]	$\mathbf{f} = c\mathbf{1}$	Yes	Yes	Yes	No

Table 5.3: Comparison of proposed two-channel filterbank designs on bipartite graphs. DC: subspace corresponding to lowest eigenvalue, CS: Critical Sampling, PR: Perfect Reconstruction, Comp: compact support, OE: Orthogonal Expansion

5.7 Summary

In this chapter, we proposed the construction of critically sampled wavelet filterbanks for analyzing graph-signals defined on any undirected weighted bipartite graph. We designed wavelet filters based on spectral techniques, and provided necessary and sufficient conditions for aliasing cancellation, perfect reconstruction and orthogonality in these filterbanks. As a practical solution,

[4]designed using symmetric Laplacian matrix $\mathcal{L}$
[5]designed using asymmetric Laplacian matrix $\mathcal{L}_a$
[6]for analysis filters only.
[7]This reconstruction error can be reduced to arbitrary small levels by increasing the degree of approximation.

we have proposed a graph-QMF design for bipartite graphs which has all the above mentioned features. The filterbanks are however, realized by Chebychev polynomial approximations at the cost of small reconstruction error and loss of orthogonality. As alternatives to graph-QMF filterbanks on graphs, we described two approaches for constructing two-channel filter-banks on bipartite graphs. One of the approaches is to design spectral 2-channel filterbanks, where analysis filters are designed using linear spectral kernels, and synthesis filters are chosen so as to guarantee invertibility. The other approach is to design spectral 2-channel filterbanks, where both analysis and synthesis filters are based on polynomial spectral kernels. These filters are not orthogonal, but they have compact support, and provide perfect reconstruction. All these filterbanks are designed to operate on a bipartite graph. In the next chapter, we describe a separable multi-dimensional implementation of these designs to any arbitrary graph via bipartite subgraph decomposition.

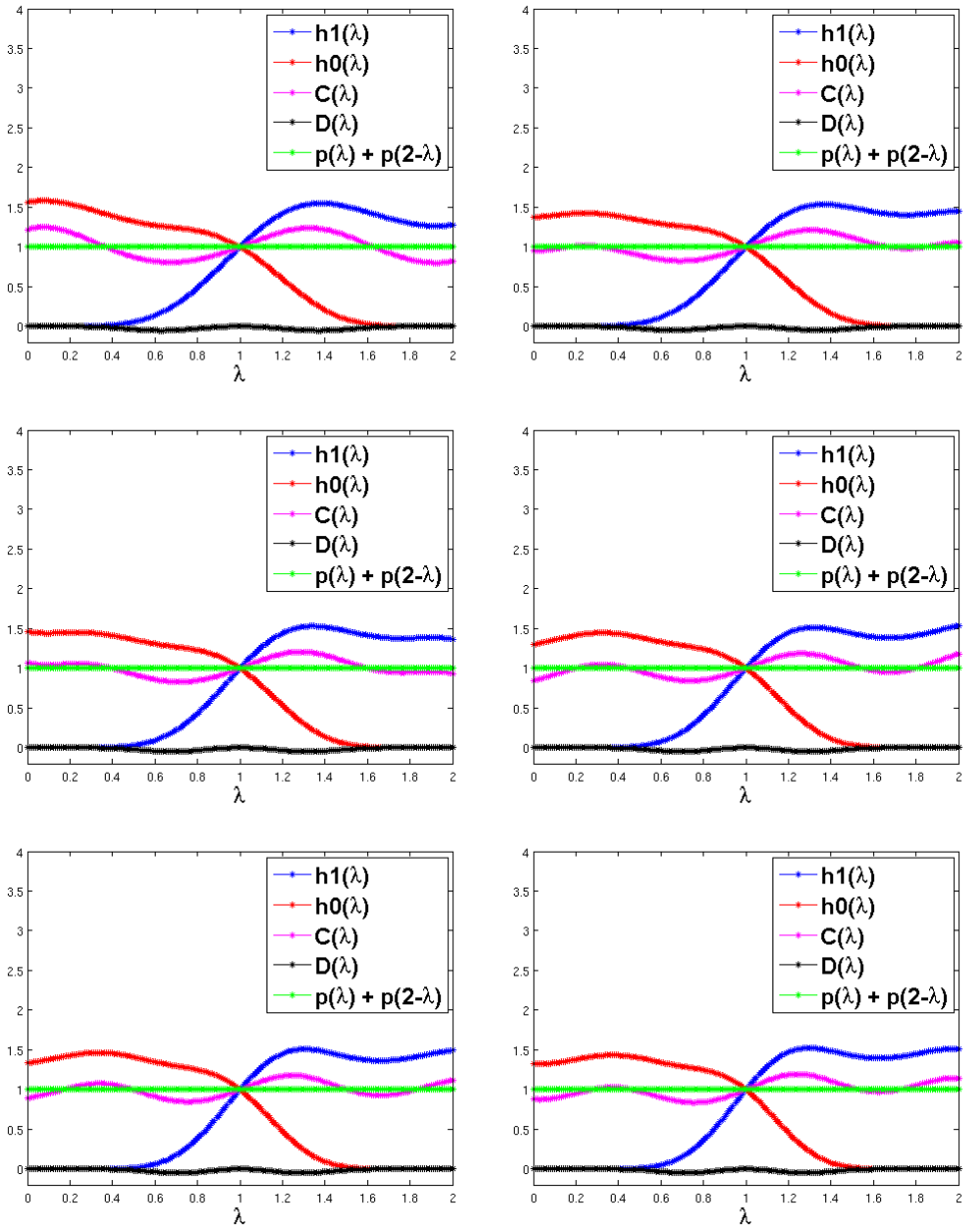


Figure 5.4: Spectral responses of graphBior(k_0, k_1) filters on a bipartite graph. In each plot, $h_0(\lambda)$ and $h_1(\lambda)$ are low-pass and high-pass analysis kernels, $C(\lambda)$ and $D(\lambda)$ constitute the spectral response of the overall analysis filter $\mathbf{T}_a$, as in (5.65). For near-orthogonality $D(\lambda) \approx 0$ and $C(\lambda) \approx 1$. Finally, $(p(\lambda) + p(2-\lambda))/2$ represents perfect reconstruction property as in (5.51), and should be constant equal to 1, for perfect reconstruction.

Chapter 6

Separable Multi-dimensional Wavelet Filterbanks on Graphs

Both, the lifting wavelet filterbanks in Chapter 3 and the spectral wavelet filterbanks in Chapter 5, are designed for bipartite graphs. This is because, bipartite graphs are a natural choice for implementing lifting wavelet filterbanks, and provide easy-to-interpret perfect reconstruction conditions for spectral wavelet filterbanks, in terms of simple functions of spectral kernels. However, not all graphs are bipartite. For arbitrary graphs, the results applicable to bipartite graphs can be extended in a variety of ways. One way is to approximate G with a bipartite subgraph $\hat{G}$, and implement designs proposed in Chapter 3 and Chapter 5 on the approximate graph. This approach results in *edge-losses*, since the edges between nodes belonging to the same partition are discarded while computing the transform coefficients. We refer to this approach as *"one-dimensional" high loss* implementation. As an alternative, we decompose the graph G into K edge-disjoint bipartite subgraphs whose union is G and implement filtering/downsampling operation in K stages, restricting the filtering/downsampling operations in each stage to one bipartite graph. This way, all the edges in the graphs participate in computing the wavelet transform. We refer to this approach as *"multi-dimensional" no loss* implementation. However, the requirement of using all edges in the graph may sometimes lead to very high-dimensional representation of graph-signals, where most of the edges are contained in a few bipartite subgraphs, and the remaining subgraphs are nearly empty (in terms of edges). Therefore, another alternative is to use a hybrid approach, where we compute K edge-disjoint bipartite subgraphs whose union is G, but discard bipartite subgraphs with very few edges. This way, some edges in G are not used in computing the wavelet transform, but these edges constitute a very small fraction of the total number of edges. We refer

to this approach as *"multi-dimensional" low-loss* implementation.

In this chapter, we describe the properties of "multi-dimensional" implementations of proposed two-channel filterbanks. In high-dimensional regular signals, filtering and downsampling is done along the geometrical directions (horizontal, vertical etc) of the underlying regular lattice. The subgraph decomposition in graphs can be interpreted in the same spirit as defining "graph dimensions" for filtering and downsampling. Thus, a graph "dimension" can be interpreted as a subset of links $\hat{E} \subset E$ for traversing the graph, starting at any node, and can also be represented as a subgraph $(\mathcal{V}, \hat{E})$ of graph $(\mathcal{V}, E)$. Further, two graph-dimensions may be considered "orthogonal", if the graph-filters implemented in these dimensions (i.e, on the corresponding subgraphs), measure non-redundant information. This can be achieved if the sets of nodes discovered, while traversing the graph starting at any node, in two different dimensions are mutually disjoint. *Therefore, the "dimensionality" of a graph G can be defined as the minimum K, for which the graph can be decomposed into K subgraphs $(\mathcal{V}, \hat{E}_p)$, $p = 1, 2, ...K$, such that the k-hop neighborhood sets $\{\mathcal{N}_k^p(n)\}$, centered at a node n, corresponding to all subgraphs $(\mathcal{V}, \hat{E}_p)$, are pairwise disjoint for all k, and for all nodes n.* Further, since our proposed filterbanks operate only on bipartite graphs, we define dimensionality in terms of decomposing the graph into K bipartite subgraphs. So, the question that frames the rest of our discussion in this chapter is that of how to find these orthogonal subgraph decompositions. A more relevant question, especially for the *multi-dimensional low-loss* case, is to find "good" K-dimensional bipartite subgraph decompositions of a graph for a fixed K.

In this chapter, we propose a *separable downsampling and filtering* approach to apply our filterbank design to an arbitrary graph, $G = (\mathcal{V}, E)$, where our previously designed two-channel filterbanks are applied in a "cascaded" manner, by filtering along a series of bipartite subgraphs of the original graph. This is illustrated in Figure 6.1. We call this a "separable" approach in analogy to separable transforms for regular multidimensional signals. For example in the case of separable transforms for 2D signals, filtering in one dimension (e.g., row-wise) is followed by filtering of the outputs along the second dimension (column-wise). In our proposed approach, a stage of filtering along one "dimension" corresponds to filtering using *only* those edges that belong to the corresponding bipartite subgraph. As shown in Figure 6.1, after filtering along one subgraph the results are stored in the vertices, and a new transform is applied to the resulting graph signals following the edges of the next level bipartite subgraph. We study the desired properties

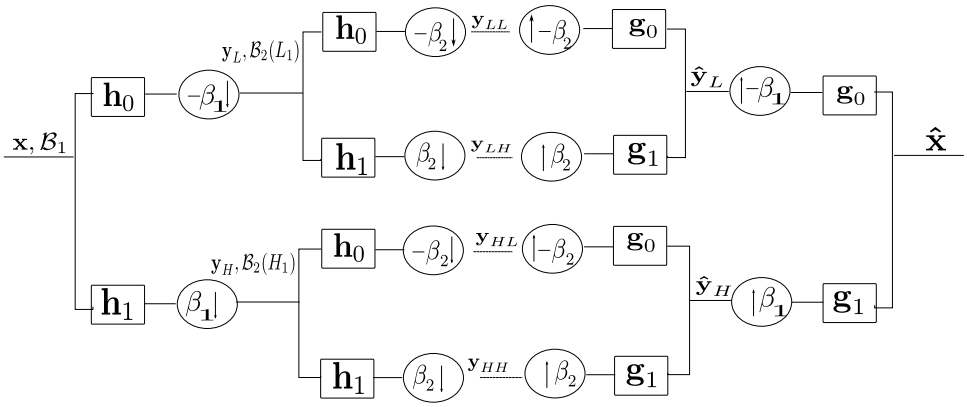


Figure 6.1: Block diagram of a 2D Separable two-channel Filter Bank: the graph G is first decomposed into two bipartite subgraphs $\mathcal{B}_1$ and $\mathcal{B}_2$, using the proposed decomposition scheme. By construction $\mathcal{B}_2$ is composed of two disjoint graphs $\mathcal{B}_2(L)$ and $\mathcal{B}_2(H)$, each of which is processed independently, by one of the two filterbanks at the second stage. The 4 sets of output transform coefficients, denoted as $\mathbf{y}_{HH}, \mathbf{y}_{HL}, \mathbf{y}_{LH}$ and,$\mathbf{y}_{LL}$, are stored at disjoint sets of nodes.

of these bipartite subgraph decompositions, and propose metrics to quantitatively measure the separations. Subsequently, we propose greedy heuristic to optimize these metrics and compare the resulting decompositions with other non-optimized schemes.

The rest of the chapter is organized as follows: in Section 6.1, we describe our proposed approach for implementing wavelet filterbanks on arbitrary graphs via bipartite subgraph decomposition. In Section 6.2, we discuss the desired properties of bipartite subgraph decomposition in the filterbank design, and define some metrics to compare various bipartite decompositions based on these properties. In the same section we propose two algorithms, namely *Harary's decomposition*, and min-cut weighted max-cut *(MCWMC) decomposition*, to compute bipartite subgraphs. In Section 6.3, we compare proposed algorithms in terms of desired properties. Finally we conclude the chapter in Section 6.4.

6.1 Proposed Design

In what follows we will assume that G has been decomposed into a series of K bipartite subgraphs $\mathcal{B}_i = (L_i, H_i, E_i)$, $i = 1 \dots K$; how such a decomposition may be obtained will be discussed later. The bipartite subgraphs cover the same vertex set: $L_i \cup H_i = \mathcal{V}$, $i = 1, 2, ...K$. Each edge in G belongs to exactly one E_i, i.e., $E_i \cap E_j = \emptyset$, $i \neq j$, $\bigcup_i E_i = E$. Note that for each bipartition we need to decide both a 2-coloring (H_i, L_i) *and* an assignment of edges (E_i). In order to guarantee

invertibility for structures such as those of Figure 6.1, given the chosen 2-colorings (H_i, L_i), the edge assignment has to be performed iteratively based on the order of the subgraphs. That is, edges for subgraph 1 are chosen first, then those for subgraph 2 are selected, and so on. The basic idea is that at each stage i *all* edges between vertices of different colors *that have not been assigned yet* will be included in E_i. More formally, at stage i with sets H_i and L_i, E_i contains *all* the links in $E - \bigcup_{k=1}^{i-1} E_k$ that connect vertices in L_i to vertices in H_i. Thus E_1 will contain all edges between H_1 and L_1. Then, we will assign to E_2 all the links between nodes in H_2 and L_2 that were not already in E_1. This is also illustrated in Figure 6.2. Note that by construction $G_1 = G - \mathcal{B}_1 = (\mathcal{V}, E - E_1)$ contains now two disjoint graphs, since all edges between L_1 and H_1 were assigned to E_1. Thus, at the second stage in Figure 6.1, $\mathcal{B}_2$ is composed of two disjoint graphs $\mathcal{B}_2(L_1)$ and $\mathcal{B}_2(H_1)$, which each will be processed independently by one of the two filterbanks at this second stage. Clearly, this guarantees invertibility of the decomposition of Figure 6.1, since it will be possible to recover the signals in $\mathcal{B}_2(L_1)$ and $\mathcal{B}_2(H_1)$ from the outputs of the 2nd stage of the decomposition. The same argument can be applied to the decompositions with more than two stages. That is, the output of a two-channel filterbank at level i leads to two subgraphs, one per channel, that are disconnected when considering the remaining edges $(E - \bigcup_{k=1}^{i} E_k)$. The output of a K-level decomposition leads to 2^K disconnected subgraphs.

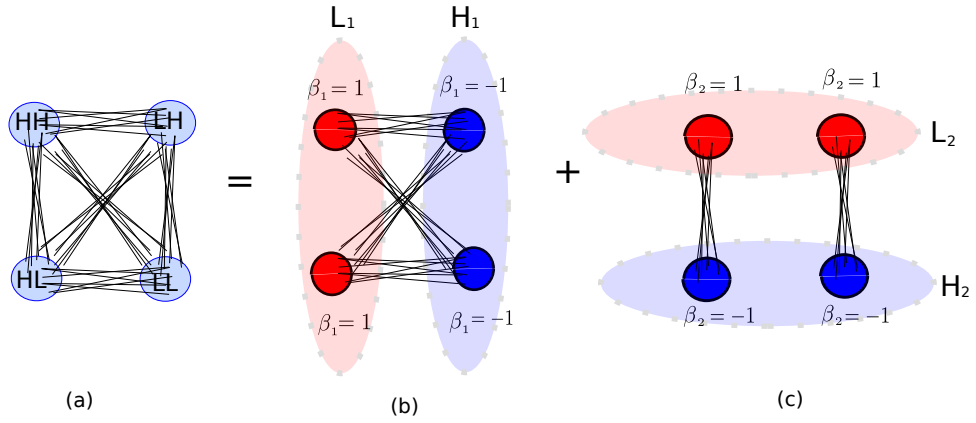


Figure 6.2: Example of 2-dimensional separable downsampling on a graph: (a) original graph G, (b) the first bipartite graph $\mathcal{B}_1 = (L_1, H_1, E_1)$, containing *all* the links in G between sets L_1 and H_1. (c) the second bipartite graph $\mathcal{B}_2 = (L_2, H_2, E_2)$, containing *all* the links in $G - \mathcal{B}_1$, between sets L_2 and H_2

We now derive expressions for the proposed cascaded transform along bipartite subgraphs. Using the $K = 2$ case as an example, assuming that the original graph can be approximated exactly

with two bipartite subgraphs as shown in Figure 6.2, we choose $\beta_i = \beta_{H_i}$ as the downsampling function for bipartite graph $\mathcal{B}_i$, for $i = 1, 2$. Further, let us denote $\mathbf{J}_{\beta_i}$, as the downsampling matrices, and $\mathbf{H}_{i0}$ and $\mathbf{H}_{i1}$ as the low-pass and high-pass filters respectively, for the bipartite graph $\mathcal{B}_i$, for $i = 1, 2$. Since, the vertex sets L_1 and H_1 in bipartite graph $\mathcal{B}_2$ are disconnected, the filtering and downsampling operations on graphs $\mathcal{B}_2(L_1)$ and $\mathcal{B}_2(H_1)$ do not interact with each other. Therefore, graph-filters $\mathbf{H}_{2j}$, for $j = 0, 1$ on the second bipartite graph $\mathcal{B}_2$, can be represented as *block-diagonal matrices* with diagonal entries $\mathbf{H}_{2j}(H_1, H_1)$ and $\mathbf{H}_{2j}(L_1, L_1)$. As a result, $\mathbf{H}_{20}$ and $\mathbf{H}_{21}$ commute with downsampling matrix $\mathbf{J}_{\beta_1}$ of the first bipartite subgraph, i.e.,

$$\mathbf{H}_{2j}\mathbf{J}_{\beta_1} = \mathbf{J}_{\beta_1}\mathbf{H}_{2j}, \tag{6.1}$$

for $j = 1, 2$. [1] Further, let $\mathbf{T}_{ai}$ be the equivalent analysis transform for $\mathcal{B}_i$, for $i = 1, 2$. The combined analysis transform $\mathbf{T}_a$ in the 2-dimensions can be written as the product of analysis transform in each dimension. Using (5.16), we obtain:

$$\mathbf{T}_a = \mathbf{T}_{a2}.\mathbf{T}_{a1} = \prod_{i=1}^{2} \frac{1}{2} \left((\mathbf{H}_{i1} + \mathbf{H}_{i0}) + \mathbf{J}_{\beta_i}(\mathbf{H}_{i1} - \mathbf{H}_{i0}) \right), \tag{6.2}$$

Note that for graph-Bior filter designs, lifting wavelet filter designs, and for exact graph-QMF filter design such as with the Meyer kernel in (5.33), $\mathbf{T}_{ai}$ is invertible with $\mathbf{T}_{ai}^{-1} = \mathbf{T}_{ai}^{t}$, for $i = 0, 1$. As a result, $\mathbf{T}_a$ is invertible with $\mathbf{T}_a^{-1} = \mathbf{T}_{a1}^{t}.\mathbf{T}_{a2}^{t}$ [2]. The transform function $\mathbf{T}_a$ can be further decomposed into the transform functions $\mathbf{T}_{HH}, \mathbf{T}_{Hl}, \mathbf{T}_{LH}$ and $\mathbf{T}_{LL}$ corresponding to the four channels in Figure 6.1. For example, the transform $\mathbf{T}_{HH}$, consists of all the terms in the expansion of $\mathbf{T}_a$ in (6.2), containing filters H_{11} and H_{21}. Thus,

$$\mathbf{T}_{HH} = \frac{1}{4}(\mathbf{H}_{21}\mathbf{H}_{11} + \mathbf{H}_{21}\mathbf{J}_{\beta_1}\mathbf{H}_{11} + \mathbf{J}_{\beta_2}\mathbf{H}_{21}\mathbf{H}_{11} + \mathbf{J}_{\beta_2}\mathbf{H}_{21}\mathbf{J}_{\beta_1}\mathbf{H}_{11}), \tag{6.3}$$

where $(1/4)\mathbf{H}_{21}\mathbf{H}_{11}$ is the transform without downsampling, and the remaining terms arise primarily due to the downsampling in the HH channel. Using (6.1), which is a property of our proposed decomposition scheme in (6.3), we obtain:

[1] In general, this result can be applied to any general K-dimensional decomposition using proposed recursive method, as the downsampling matrix $\mathbf{J}_{\beta_i}$ commutes with all filter matrices $\mathbf{H}_{k1}$ and $\mathbf{H}_{k2}$ corresponding to bipartite subgraph $\mathcal{B}_k$, where $k > i$.

[2] For polynomial approximations, of Meyer kernels, we incur some reconstruction errors in each dimension.

$$\begin{aligned}\mathbf{T}_{HH} &= \frac{1}{4}(\mathbf{H}_{21}\mathbf{H}_{11} + \mathbf{J}_{\beta_1}\mathbf{H}_{21}\mathbf{H}_{11} \\ &+ \mathbf{J}_{\beta_2}\mathbf{H}_{21}\mathbf{H}_{11} + \mathbf{J}_{\beta_2}\mathbf{J}_{\beta_1}\mathbf{H}_{21}\mathbf{H}_{11}) \\ &= \frac{1}{4}(\mathbf{I} + \mathbf{J}_{\beta_2})(\mathbf{I} + \mathbf{J}_{\beta_1})\mathbf{H}_{21}\mathbf{H}_{11}.\end{aligned} \tag{6.4}$$

Thus, the equivalent transform in each channel of the proposed 2-dimensional separable filterbanks can be interpreted as filtering with a 2-dimensional filter, such as $\mathbf{H}_{21}\mathbf{H}_{11}$ for the HH channel, followed by DU operations with two downsampling functions $\beta_2(n)$ and $\beta_1(n)$ in cascade. It also follows from (6.5), that the output of $\mathbf{H}_{21}\mathbf{H}_{11}$ in the HH channel is stored only at the nodes corresponding to $H_1 \cap H_2$. Thus, the output of each channel is stored at mutually disjoint sets of nodes, and each node stores the output of exactly one of the channel. Therefore, the overall filterbank is *critically sampled*. Further, if the spectral decompositions of $\mathcal{B}_1$ and $\mathcal{B}_2$ are given as $\{\lambda, \mathbf{P}^1_\lambda\}$ and $\{\gamma, \mathbf{P}^2_\gamma\}$, then $\mathbf{H}_{21}\mathbf{H}_{11}$ consists of a two dimensional spectral kernel $h_{21}(\gamma)h_{11}(\lambda)$ and corresponding eigenspace $\mathbf{P}^2_\gamma\mathbf{P}^1_\lambda$. The analysis extends to any dimension $K > 2$ with K-dimensional graph-frequencies $(\lambda_1, \lambda_2, ..., \lambda_K)$, corresponding eigenspace $\mathbf{P}^1_{\lambda_1}, \mathbf{P}^2_{\lambda_2}, ...\mathbf{P}^K_{\lambda_K}$ and transforms with spectral response $\prod_{i=1}^K g_i(\lambda_i)$.

Note that invertible cascaded transforms can also be constructed even when the conditions for edge selection described are not followed, e.g., if an edge e_1 between nodes in H_1 and L_1 is *not* included in E_1. In such a situation, it is possible to perform an invertible cascaded decomposition if e_1 is no longer used in further stages of decomposition. Thus, we would have an invertible decomposition but on a graph that approximates the original one (i.e., without considering e_1). Alternatively it can be shown that it is possible to design invertible transforms with arbitrary E_i selections (i.e., not following the rules set out in this chapter), but these transforms are not necessarily critically sampled. A more detailed study of this case falls outside of the scope of this thesis.

6.1.1 Graph after downsampling

The DU operations in graphs, only define the node-sets H or L to be retained after downsampling. For bipartite graphs, unlike the case of regular lattices, the resulting downsampled graphs G_L and G_H may neither be identical nor bipartite. Therefore, for the next level of decomposition, we can

either operate on a single bipartite graph approximation of G_L which leads to a one-dimensional two-channel filterbank, or a multiple bipartite graph approximation, which leads to a multi-dimensional two-channel filterbank implementation on the downsampled graph, which shall be explained in the Chapter 5. Further, this multiresolution decomposition of graph-signals can be extended to the case of general K-dimensional two-channel filterbanks for any arbitrary graph G, which decomposes the signal into 2^K lower-resolution versions, as described in Section 6.1. In this case, the downsampled graphs in each channel, can be computed by reconnecting two nodes in the downsampled vertex set, if they are 2^K-hops away in the original graph.

6.2 Bipartite Subgraph Decomposition

So far we have described how to implement separable multi-dimensional two-channel filterbanks on a graph G, given a decomposition of G into K bipartite subgraphs. In particular, we defined a "separable" method of graph decomposition, which leads to a cascaded tree-structured implementation of the multi-dimensional filterbanks. While these multi-dimensional filterbanks can be implemented for *any* separable bipartite subgraph decomposition of G, the definition of a "good" bipartite decomposition is not clear. In this section, we study the desired properties of these bipartite subgraph decompositions. As described in the beginning of this chapter, the bipartite subgraph decomposition of a graph can be interpreted as decomposing the graph into different "graph-dimensions", where a "graph dimension" refers to a subset of edges in the graph. Further, two bipartite subgraphs are "orthogonal" if the neighborhood sets defined on the two subgraphs at each node are mutually disjoint. However, it can be problematic to strictly impose orthogonality in the decomposition of some graphs (specially dense graphs), where this can lead to the generation of too many bipartite subgraphs, with very few edges in most of these subgraphs. Therefore, the question is, given a fixed K (such as $K = 2$ in this case), what is a "good" K-dimensional bipartite subgraph decomposition of a graph. The answer we propose is a bipartite subgraph decomposition $G = \bigcup_p^K \mathcal{B}_p$ in which the k-hop neighborhoods $\mathcal{N}_k^p(n)$ defined on each subgraph $\mathcal{B}_p$ are *maximally* disjoint for all k and for all n. For simplicity, we restrict our discussion to the 2-dimensional case. The extension to higher dimensional decomposition is straightforward.

Before finding the solution, we define some metrics which measure the neighborhood separation in bipartite subgraphs. For this, we define k-hop adjacency matrix as $\mathbf{A}_{i,k}$ so that $\mathbf{A}_{i,k}(n, m)$

represents the number of paths from node n to node m of length up to k, in the bipartite subgraph $\mathcal{B}_i$. The diagonal entries of $\mathbf{A}_{i,k}$ are set to zero. Using matrix $\mathbf{A}_{i,k}$, we measure separability in the k-hop neighborhoods of bipartite subgraphs $\mathcal{B}_i$ by computing the correlation between n^{th} rows of adjacency matrices $\mathbf{A}_{i,k}$ at each node n. The *k-hop neighborhood set correlation* NSC(k) between two bipartite subgraphs is the average correlation between the k-hop neighborhoods defined as:

$$NSC(k) = \frac{1}{N}\sum_{n=1}^{N} \frac{\sum_m \mathbf{A}_{1,k}(n,m)\mathbf{A}_{2,k}(n,m)}{\sqrt{\sum_m \mathbf{A}_{1,k}(n,m)^2 \sum_m \mathbf{A}_{2,k}(n,m)^2}} \tag{6.5}$$

A low value of NSC would imply mutually disjoint neighborhoods in the decomposed bipartite subgraphs. At a global scale, the eigen-vectors of bipartite subgraph Laplacian matrices which form the graph-Fourier basis should also be decorrelated with each other. The correlation between the l^{th} eigen-vectors $\mathbf{u}_{1,l}$ and $\mathbf{u}_{2,l}$ on two bipartite subgraphs can be measured by their inner-product. Therefore, we define *spectral basis correlation* SBC between bipartite subgraphs $\mathcal{B}_1$ and $\mathcal{B}_2$ to be the Euclidean norm of inner-products between the corresponding eigenvectors:

$$SBC = \sqrt{\sum_{l=1}^{N} (\mathbf{u}_{1,l}^t \mathbf{u}_{2,l})^2} \tag{6.6}$$

To measure the loss because of the approximation, we define an *edge-loss fraction* (ELF) which is the ratio between total number of edges in remaining graph $G_2 = G - \mathcal{B}_1 - \mathcal{B}_2$ and the total number of edges in G. Thus, ELF measures the fraction of edges not used in computing the transform. We next present two heuristic algorithms to find good subgraph decompositions in arbitrary graphs.

6.2.1 Harary's decomposition algorithm

In this section, we propose a bipartite subgraph decomposition method, referred to as *Harary's decomposition*, which provides a $\lceil log_2 k \rceil$ bipartite decomposition of a graph G given a k-coloring defined on it[3]. The method is derived from [15] and we describe it in Algorithm 3.[4] Although the

[3] A graph is perfectly k-colorable if its vertices can be assigned k-colors in such a way that no two adjacent vertices share the same color. The term *chromatic number* $\chi(G)$ of a graph refers to smallest such k.

[4] Note that the bipartite decomposition is not unique and depends on the ordering in which the k-colors are divided.

problem of determining the chromatic number $\chi(G)$ is NP-complete, there exist several approximate minimum coloring algorithms with various orders of accuracies, a comparison of which can be found in [23]. The complexity of these algorithms range from $\mathcal{O}(N^2)$ for greedy algorithms to $\mathcal{O}(N^4)$ for a backtracking sequential coloring (BSC) algorithm presented in [23]. Based on this result, we propose a $\lceil log_2 k \rceil$- bipartite decomposition of the graph G, given a perfect k-coloring defined on it. We refer to this method as *Harary's decomposition*

Algorithm 3 Harary's Decomposition

Require: $\mathbf{F}$, s.t. $F(v)$ is the color assigned to node v, $\min(F)=1$, $\max(F)=k$.

1: Set L_1 = set of nodes with $F(v) \leq \lfloor k/2 \rfloor$ colors.
2: Set H_1 = set of nodes with $F(v) > \lceil k/2 \rceil$ colors.
3: Set $E_1 \subset E$ containing all the edges between sets H_1 and L_1.
4: Compute bipartite subgraph $\mathcal{B}_1 = (L_1, H_1, E_1)$,
5: Set $G = G - \mathcal{B}_1$.
6: G is now a union of two disconnected subgraphs $G(H_1)$ and $G(L_1)$.
7: Graph $G(L_1)$ is $\lceil k/2 \rceil$-colorable.
8: Compute coloring $\mathbf{F}_L$ on $G(L_1)$ s.t. $\min(F_L)=1$, $\max(F_L)=\lceil k/2 \rceil$.
9: Graph $G(H_1)$ is $\lfloor k/2 \rfloor$-colorable.
10: Compute coloring $\mathbf{F}_H$ on $G(H_1)$ s.t. $\min(F_H)=1$, $\max(F_L)=\lfloor k/2 \rfloor$.
11: Repeat $1-4$ on $G(L_1)$ and $G(H_1)$ to obtain bipartite subgraphs $\mathcal{B}_2(L_1)$ and $\mathcal{B}_2(H_1)$.
12: Compute bipartite subgraph $\mathcal{B}_2 = \mathcal{B}_2(L_1) \cup \mathcal{B}_2(H_1)$.
13: Set $G = G - \mathcal{B}_2$.
14: repeat $1-13$ exactly $\lceil log_2 k \rceil$ times after which graph G will become an empty graph.

6.2.2 Min-cut weighted max-cut (MCWMC) algorithm

The nature and complexity of finding minimum bipartite subgraph decomposition with maximally disjoint neighborhoods is not known. We therefore, propose the following greedy heuristic to find bipartite subgraphs with disjoint neighborhoods: given a graph G, let β be chosen as the first downsampling function inducing a partition $(\mathcal{S}_1, \mathcal{S}_2)$ on graph G with sizes $|\mathcal{S}_1| = N_1$ and $|\mathcal{S}_2| = N_2$. Let us define $p(\mathcal{S})$ to be the probability of randomly choosing a node $v \in \mathcal{S}$ in graph G, which is equal to $|\mathcal{S}|/|V|$. Further let $\mathbf{e} = E_{\mathcal{S}_1,\mathcal{S}_2}$ denote the cut-set and $\mathcal{B} = (\mathcal{S}_1, \mathcal{S}_2, \mathbf{e})$ denote the bipartite subgraph corresponding to β. This decomposition can be graphically represented as in Figure 6.3. The exclusion of $\mathcal{B}$ from G changes the neighborhood structure of the resulting graph G_1. Thus in remaining graph G_1, nodes in set $\mathcal{S}_1$ cannot reach nodes in set $\mathcal{S}_2$ and vice-versa. We define the expected change in the neighborhood size at each node given the cut $\mathbf{e}$ as:

$$E[\partial\mathcal{N} \mid \mathbf{e}] = p(\mathcal{S}_1)|\mathcal{S}_2| + p(\mathcal{S}_2)|\mathcal{S}_1| = \frac{2N_1N_2}{N} \tag{6.7}$$

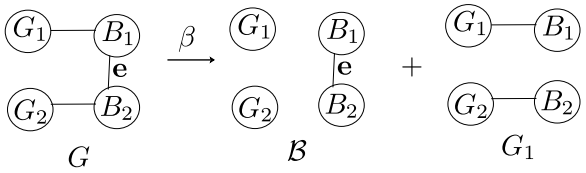


Figure 6.3: Example of a bipartite graph-cut

Clearly $E[\partial\mathcal{N}]$ is maximized if $N_1 \approx N_2$ at each iteration. This problem is widely studied in graph-literature as the *balanced-cut problem* in graphs. However finding balanced-cut iteratively becomes problematic as it leads to roughly $log_2(N)$ bipartite subgraphs for graph-size N. Further, in each bipartite graph the nodes that do not have edges in the cut-set $\mathbf{e}$ are disjoint and do not take part in the transform. Therefore, we would like to maximally pack these edges into larger and fewer bipartite subgraphs, packing edge-sets $\mathbf{e}$ in the order of their importance $E[\partial\mathcal{N} \mid \mathbf{e}]$. In order to do this we assign a weight $w_e = E[\partial\mathcal{N}]/|\mathbf{e}|$ to each edge in the cut-set $e \in \mathbf{e}$ in each iteration of balanced cut decomposition. The weight signifies the importance of the edge in changing the neighborhood structure of resulting decompositions. We perform an iterative max-cut algorithm on the resulting min-cut weighted graph which provides bipartite subgraphs with maximum packing of the weighted edges. The algorithm is thus termed as the *min-cut weighted max-cut* (MCWMC) algorithm, and described in Algorithm 4.

6.3 Experiments

In order to evaluate the different schemes for bipartite subgraph decomposition, we simulate random graphs by uniformly distributing $N = 100$ nodes in a 2-D field and connecting nodes which are within a fixed radius of each other.[5] For MCWMC algorithm, we use the balanced-cut algorithm proposed in [16] and the max-cut algorithm in [1]. For each graph G, we decompose the graph iteratively into bipartite subgraphs up to two steps to obtain bipartite subgraphs $\mathcal{B}_1$ and $\mathcal{B}_2$ respectively. We then evaluate the metrics NSC and SBC for the two bipartite subgraphs obtained by using a) MCWMC algorithm b) Harary's algorithm proposed in [30] and c) a random decomposition (in which we randomly assign downsampling functions to nodes). Table 6.1 summarizes

[5]Note that the 2-D embedding of graph is for illustration only. The MCWMC algorithm only depends on the link-structure of the graph nodes.

Algorithm 4 MCWMC Decomposition

Require: $G = (\mathcal{V}, E)$
Set $(\mathcal{V}, E_d) = min\_cut\_weighing(\mathcal{V}, E)$
Set $k \rightarrow 1$.
while $|E| \neq 0$ **do**
  Compute normalized Laplacian matrix $\mathcal{L}(G_d)$.
  Compute eigenvector $\mathbf{u}_\lambda$, of maximum magnitude eigenvalue λ of $\mathcal{L}(G_d)$.
  Set $L_k = \{v : \mathbf{u}_\lambda(v) \geq 0\}$.
  Set $H_k = \{v : \mathbf{u}_\lambda(v) < 0\}$.
  Set $E_k = \{(u, v) : (u, v) \in E,\ u \in L_k, v \in H_k\}$.
  Set $E_{dk} = \{(u, v) : (u, v) \in E_d,\ u \in L_k, v \in H_k\}$.
  Set $\mathcal{B}_k = (L_k, H_k, E_k)$.
  Set $E = E \setminus E_k$; $G = (\mathcal{V}, E)$
  Set $E_d = E_d \setminus E_{dk}$; $G_d = (\mathcal{V}, E_d)$
  Set $k \rightarrow k + 1$.
end while
function $G_d = min\_cut\_weighing(\mathcal{V}, E)$
if $|E| = 0$ **then**
  Set $E_d = E$.
else
  Compute normalized Laplacian matrix $\mathcal{L}(G)$, where $G = (\mathcal{V}, E)$.
  Compute eigenvector $\mathbf{u}_\lambda$, of minimum non-zero eigenvalue λ of $\mathcal{L}(G)$.
  Set $\mathcal{S}_1 = \{v : \mathbf{u}_\lambda(v) \geq 0\}$.
  Set $\mathcal{S}_2 = \{v : \mathbf{u}_\lambda(v) < 0\}$.
  Set $E_1 = \{(u, v) : (u, v) \in E,\ u \in \mathcal{S}_1, v \in \mathcal{S}_1\}$.
  Set $E_2 = \{(u, v) : (u, v) \in E,\ u \in \mathcal{S}_2, v \in \mathcal{S}_2\}$.
  Set $e = \{(u, v) : (u, v) \in E,\ u \in \mathcal{S}_1, v \in \mathcal{S}_2\}$.
  Compute $w_e = \frac{2|\mathcal{S}_1||\mathcal{S}_2|}{|e|(|\mathcal{S}_1|+|\mathcal{S}_2|)}$
  Set $e_d = w_e.e$. {multiply all edges in e with w_e}
  Set $(\mathcal{S}_1, E_{d1}) = min\_cut\_weighing(\mathcal{S}_1, E_1)$
  Set $(\mathcal{S}_2, E_{d2}) = min\_cut\_weighing(\mathcal{S}_2, E_2)$
  Set $E_d = E_{d1} \cup E_{d2} \cup e_d$.
end if
return $G_d = (\mathcal{V}, E_d)$
end function

the comparison results for 100 instances of such random graphs. A low value of ELF suggests that MCWMC algorithm packs more edges in subgraphs $\mathcal{B}_1$ and $\mathcal{B}_2$ than other algorithms. Further, we observe that NSC(k), in general decreases for large k-hop neighborhoods which makes sense since at each step of iterative decomposition, the removal of a bipartite subgraph bisects the remaining graph and thus reduces the long-hop connections between nodes. However, we observe that the NSC(k) drops sharply with MCWMC algorithm which implies that the neighborhood are better separated than by using other schemes. At global scale, SBC is lowest for MCWMC, which means that the eigenvectors of the resulting bipartite subgraph are also better decorrelated in case of

proposed algorithm. To see it more clearly, we measure similarity (i.e inverse of shortest hoping distance) between all node pairs in different subgraphs. With maximal neighborhood separation, we expect any pair of nodes in the graph to have different similarities on different subgraphs. Figure 6.4 plots the histogram of absolute difference in the similarities of node-pairs on different bipartite subgraphs.

Method	Random	Harary	MCWMC
ELF	0.249	0.225	0.14
NSC(2)	0.48	0.53	0.51
NSC(4)	0.50	0.54	0.51
NSC(6)	0.49	0.53	0.48
NSC(8)	0.47	0.51	0.45
NSC(10)	0.45	0.49	0.42
NSC(12)	0.43	0.48	0.39
SBC	0.60	0.61	0.55

Table 6.1: Comparison of bipartite subgraph decomposition schemes

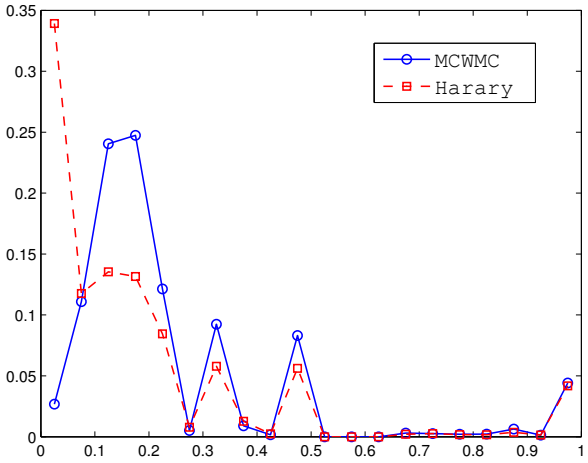


Figure 6.4: Histogram of absolute difference in similarity between node-pairs in two bipartite subgraphs

In case of Harary's decomposition the histogram is concentrated near zero which means that most node-pairs are near similar on the two subgraphs, whereas in the case of proposed MCWMC algorithm the histogram is shifted to the right implying most node-pairs are not equally similar on both subgraphs, i.e., if they are close to each other in one subgraph, then they are away from each other in other subgraph. This further corroborates our claim that neighborhood are better separated using the proposed decomposition scheme.

6.4 Summary

In this chapter, we have proposed a separable, multi-dimensional wavelet filterbank design for any arbitrary undirected graph. The design is an extension of two-channel filterbanks described in Chapter 5. According to our proposed formulation, a graph is iteratively decomposed into a set of bipartite subgraphs, and then filtering and downsampling operations are carried out in cascade on each bipartite subgraph. We have proposed bipartite subgraph decompositions, which provide dimensionality to the graph similar to the case of regularly sampled signals in higher dimensions. We explained that dimensionality in graphs can be understood as neighborhood separability, and we defined some metrics to evaluate various bipartite decompositions based on this understanding. Further, we proposed two algorithms which compute bipartite subgraph decompositions, and compared them based on these metrics. In the next chapter, we describe some applications of our proposed filterbanks.

Chapter 7

Examples and Applications of Graph Wavelet Filterbanks

In this chapter, we consider some applications where the graph filterbanks proposed in Chapter 5, can be applied. We first consider an example of *multi-resolution decomposition of graphs*, where we analyze data defined on the vertices of an arbitrarily linked graph. This is the most general illustration of our proposed designs, where we only use topological information to compute down-sampling and filtering operations. Next, we consider an *edge-aware image-processing* application, where image-pixels can be connected with their neighbors to form undirected graphs.[1] Here, we propose various graph-formulations of images, which capture both directionality and intrinsic edge information. The proposed graph-wavelet filterbanks provide a sparse, edge-aware representation of image-signals. The outline for the rest of the chapter is as follows: in Section 7.1, we discuss an example of proposed graph wavelet filterbanks in multi-resolution decomposition of graphs. In Section 7.2, we discuss application of proposed filterbanks on a graph representation of images. We present some experimental results for non-linear approximation of images in Section 7.3, and finally we conclude the chapter in Section 7.4.

7.1 Multi-resolution Decomposition of Graphs

Our proposed filterbanks can be used as a useful tool in analyzing/compressing arbitrarily linked irregular graphs. To study its feasibility, we take the example of *Minnesota traffic graph* from [14]. The graph is shown in Figure 7.1(a), where the spatial coordinates are only used to display the graph and the wavelet transform, and do not affect the edge-weights. Further, we consider

[1] This research was conducted jointly with Yung-Hsuan Chao. See [28] for details.

a graph-signal on this graph with sharp irregular discontinuity, as shown in Figure 7.1(b), where the color of a node represents the signal value at that node.

7.1.1 Bipartite subgraph decomposition

The Minnesota graph is not bipartite (two-colorable). Therefore, in order to implement wavelet filterbanks, we decompose the graph into bipartite subgraphs, using Harary's decomposition algorithm. The graph is perfectly 3-colorable and hence, it can be decomposed into $\lceil log_2(3) \rceil = 2$ bipartite subgraphs, which are shown in Figure 7.1(c-d).

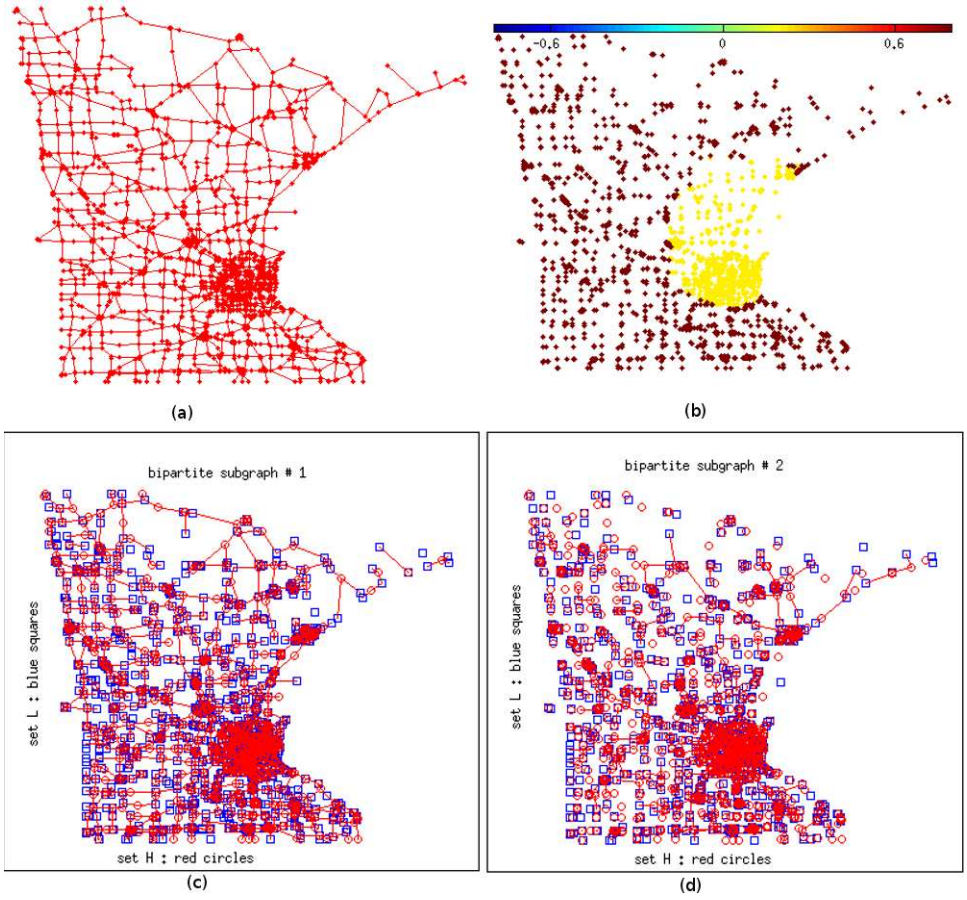


Figure 7.1: (a) The Minnesota traffic graph G, and (b) the graph-signal to be analyzed. The colors of the nodes represent the sample values. (c)(d) bipartite decomposition of G into two bipartite subgraphs using Harary's decomposition.

7.1.2 Spectral wavelet filterbank implementation

Given the bipartite subgraph decomposition of G, we compute the normalized Laplacian matrices $\mathcal{L}_i$ and the downsampling functions $\beta_i = \beta_{H_i}$, for each bipartite subgraph $\mathcal{B}_i$. Further, we compute the low-pass analysis kernel $h_{i,0}(\lambda)$ on $\mathcal{B}_i$, as the m_i^{th} order Chebychev approximation of the Meyer kernel $h_0^{Meyer}(\lambda)$, for $m_i = 24$. The remaining spectral kernels $h_{i,1}(\lambda), g_{i,0}(\lambda), g_{i,1}(\lambda)$ are computed from $h_{i,0}(\lambda)$ according to graph-QMF relations mentioned in (5.28). The corresponding analysis and synthesis transforms are then computed as $\mathbf{H}_{i,j} = h_j(\mathcal{L}_i)$ and $\mathbf{G}_{i,j} = g_j(\mathcal{L}_i)$, respectively, for $j = 0, 1$. Note that since the kernels are polynomials, the transforms are also matrix polynomials of Laplacian matrices and do not require explicit eigenspace decompositions. In our experiments, we use $m_i = m$, and hence $h_{i,j}(\lambda) = h_j(\lambda),\ j \in \{0, 1\}$ for all i, in which case the resulting transforms are exactly m-hop localized on each bipartite subgraph. The order m is a parameter of our design and should be chosen based on the required level of spatial localization and how much reconstruction error can be tolerated. The overall filterbank is designed by concatenating filterbanks of each bipartite subgraph in the form of a tree, analogous to Figure 6.1 in the 2-dimensional decomposition case. Since, the proper coloring of the Minnesota graph is 3, the output of the HL channel (i.e. nodes for which $(\beta_1(n), \beta_2(n)) = (-1, 1)$) is empty after downsampling. The output coefficients of the other three non-empty channels (LL, LH, HH) are shown in Figure 7.2. Note that after downsampling, the total number of output coefficients in the four channels is equal to the number of input samples, thus making the transform *critically sampled*. We observe in Figure 7.2 that the output coefficients in the LH and HH channels, have significantly high magnitude along the discontinuity, hence reflecting the high-pass nature of these channels. Further, in order to see how much energy of the original signal is captured in each channel, we upsample then filter the coefficient of each channel by the synthesis part of proposed filterbank. This is shown in Figure 7.3. In this figure, we see that the reconstructed signal from LL channel coefficients, provides a low-pass approximation of the original signal (sharp boundaries blurred), whereas the signals reconstructed from the LH and HL channels provide a high-pass approximation of the input signal (highlighting the boundaries). Thus, the proposed graph-based filterbanks, provide a meaningful decomposition of input signals, analogous to the standard wavelet-filterbanks.

Similarly, we design graph-Bior filterbanks. For this we choose parameters $(k_0, k_1) = (7, 7)$, as

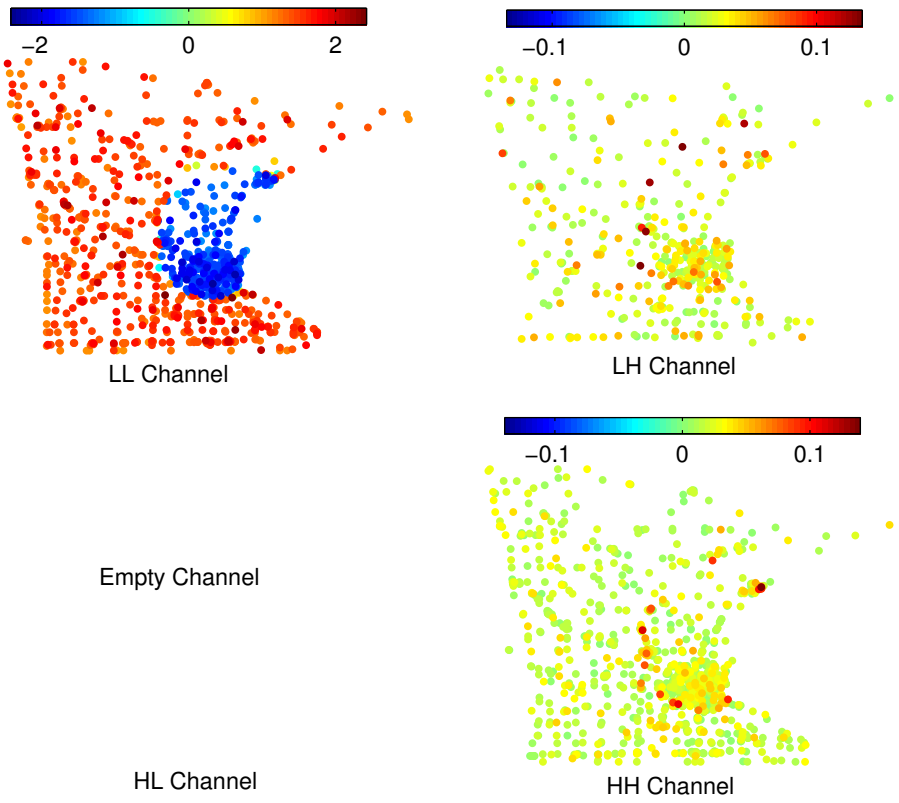


Figure 7.2: Output coefficients of the proposed graph-QMF filterbanks with parameter $m = 24$. The node-color reflects the value of the coefficients at that point. Top-left: LL channel wavelet coefficients, top-right: absolute value of LH channel wavelet coefficients, and bottom-right: absolute value of HH channel wavelet coefficients

described in Section 5.5.3. Given these specifications, we design $p(\lambda) = h_0(\lambda)g_0(\lambda)$ as a maximally flat half band polynomial kernel with $K = k_0 + k_1 = 14$ number of roots at $\lambda = 0$. As a result, $p(\lambda)$ turns out to be a $2K - 1 = 27$ degree polynomial, and we factorize it into $h_0(\lambda)$ and $g_0(\lambda)$, with least dissimilar lengths (i.e., we choose $l_0 = K = 14$ and $l_1 = K - 1 = 13$). The other two filters, namely $h_1(\lambda)$ and $g_1(\lambda)$ are computed as in (5.48). Once again, we choose the same filter kernels for both bipartite subgraphs, in which case the resulting transforms $\mathbf{H}_0$ and $\mathbf{G}_0$ are exactly K-hop and $(K-1)$-hop localized on each bipartite subgraph, respectively. The wavelet coefficients of resulting filterbanks are given in Figure 7.4, and the reconstructed signals from individual channels are given in Figure 7.4.

The proposed decomposition of graphs can be useful in many applications. From the analysis point of view, the high-magnitude samples in the signals, reconstructed from high-pass channels,

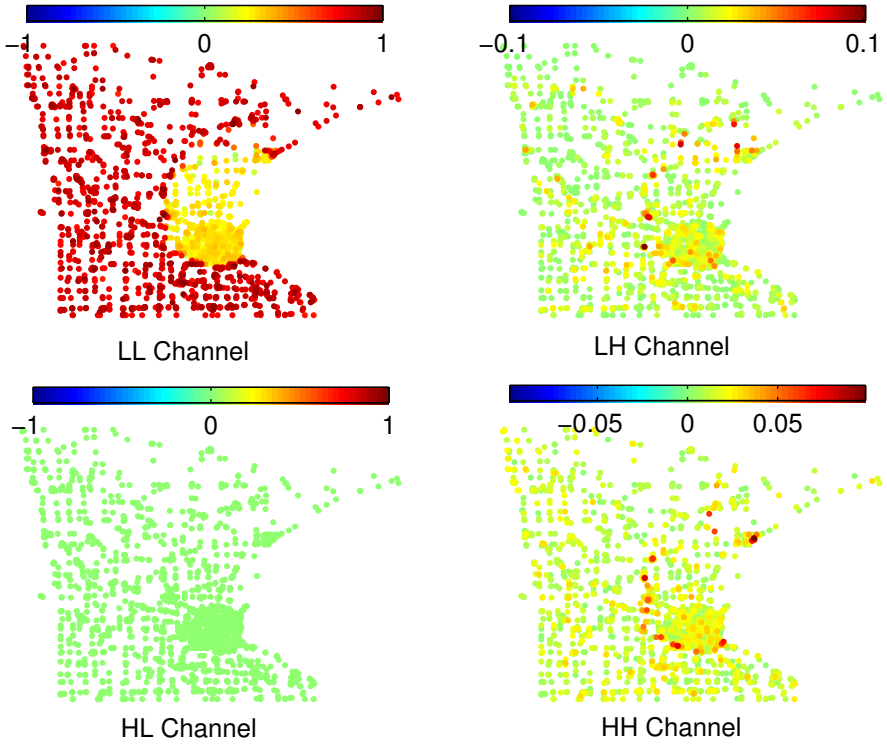


Figure 7.3: Reconstructed graph-signals from the graph-QMF wavelet coefficients of individual channels. As before the node-color reflects the value of the coefficients at that node. Top-left: reconstruction from LL channel only, top-right: reconstruction from LH channel only, and bottom-right: reconstruction from HH channel only. Since, HL channel is empty the reconstruction is an all-zero signal (bottom-left figure). The reconstruction SNR of sum of all four channels is 50.2 dB.

provide knowledge of the location and type of discontinuities in the graph-signal. For example, in a *traffic sensing scenario*, in the Minnesota graph example, the stations (nodes), with high-magnitude samples in the HH channel in Figure 7.3 and Figure 7.5, provide a good location to install traffic sensors. Further, since the filters in our proposed designs can be computed iteratively in a few steps, with local one-hop operations at each node in each step, the resulting filterbanks can be very useful in detecting anomalies in large distributed networks.

From the compression point of view, the output coefficients in the LL channel in Figure 7.2 and Figure 7.4, provides a downsampled representation of original graph-signal, with only 45% of the samples. This can be extended to multiple levels, by reconnecting the LL nodes to obtain a downsampled graph, and applying the proposed filterbanks on the downsampled graph by treating LL output coefficients as new signal, and so on. In the next section, we consider a compression

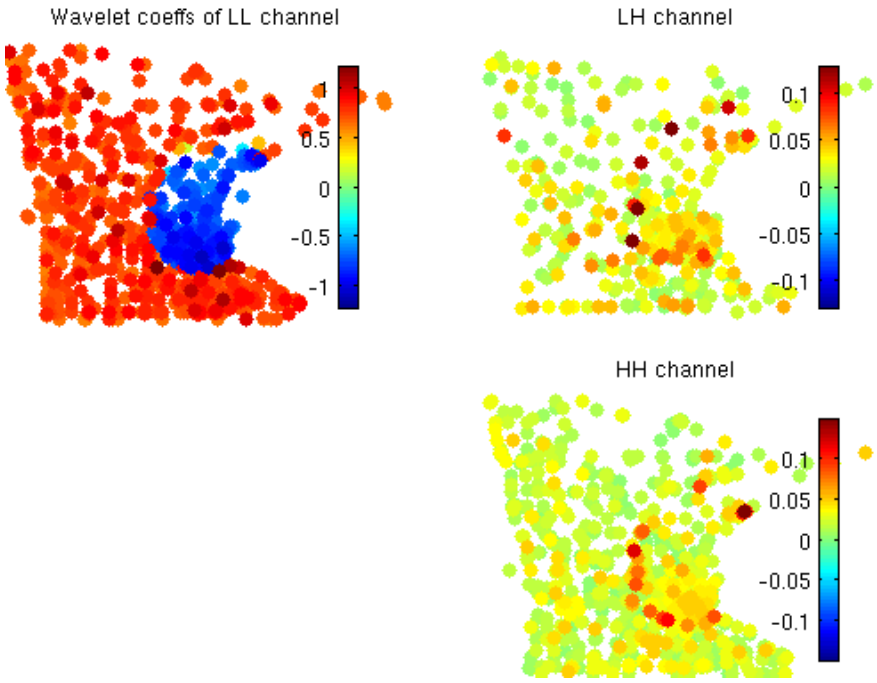


Figure 7.4: Output coefficients of the graph-Bior filterbanks with parameter $(k_0, k_1) = (7, 7)$. The node-color reflects the value of the coefficients at that point. Top-left: LL channel wavelet coefficients, top-right: absolute value of LH channel wavelet coefficients, and bottom-right: absolute value of HH channel wavelet coefficients

application for 2-D images, where we define multi-level implementations of proposed filterbanks, on graph representations of images.

7.2 Edge Aware Image Processing

In this section, we propose a novel method for image-analysis using graph wavelets. While standard separable extensions of wavelet filterbanks to higher dimensional signals, such as 2-D images, provide useful multi-resolution analysis, they do not capture the intrinsic geometry of the images. For example, these extensions can capture only limited (mostly horizontal and vertical) directional information. This means that if the object boundaries in an image are neither horizontal nor vertical, e.g., diagonal or round shape, the resulting transform coefficients tend not to be sparse and high pass wavelet components can have significant energy. Therefore, more powerful representations are sought for images, in which basis functions can adapt to the directionality

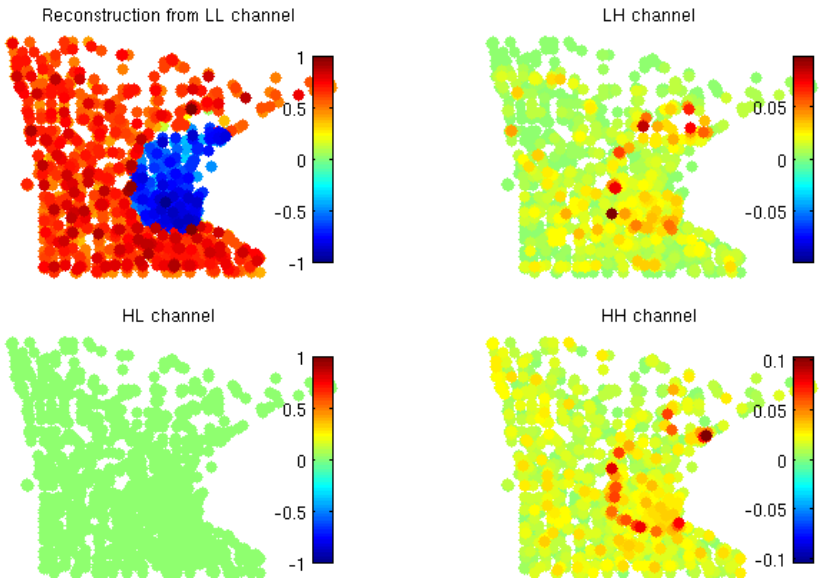


Figure 7.5: Reconstructed graph-signals from the graph-Bior wavelet coefficients of individual channels. As before the node-color reflects the value of the coefficients at that node. Top-left: reconstruction from LL channel only, top-right: reconstruction from LH channel only, and bottom-right: reconstruction from HH channel only. Since, HL channel is empty the reconstruction is an all-zero signal (bottom-left figure). The reconstruction SNR of the sum of four channels is 168.57 dB.

and edge-information contained in the image. Among the various solutions proposed, some transforms, such as 2-D Gabor wavelets [25] and complex wavelets [22], provide extra dimensionality at the cost of producing an over-sampled output. Other designs such as curvelets [2] and contourlet transforms [9], which provide a dictionary of anisotropic edge-aware basis functions, require higher complexity and suffer from the same problem of oversampling. Some other designs such as bandlets [34], directionlets [43] and tree-based lifting transforms [37] provide critically sampled transforms based on side-information about geometric flows in the image.

Images can also be viewed as graphs, by treating pixels as nodes, pixel intensities as graph-signals, and by connecting pixels with their neighbors in various ways. The advantage of formulating images as graphs is that different graphs can represent the same image, which offers flexibility of choosing the graphs that have useful properties. In particular, the weights of the links can be adjusted at each node, in order to take into account local edge-information present in the image. An example of weighted image-graph formulation is the anisotropic diffusion based image smoothing considered in [51]. In Chapter 5, we designed two-channel wavelet-filterbanks for *any*

undirected weighted graphs, with vertices (nodes) as data-sources. These filterbanks are critically sampled and provide basis elements which are localized in both spatial and frequency domain of the graph [2]. Further, they can be implemented using an iterated separable filterbank structure, and thus provide a multi-resolution analysis of graph-signals. We now apply these filterbanks to undirected unweighted graph representations of images, and show that the interpretation of resulting graph-wavelets transforms is analogous to classical wavelet decompositions.Since the DC signal in images corresponds to an all constant signal, we design graph filterbanks using asymmetric Laplacian matrix, as described in Section 5.6. We provide preliminary results related to image non-linear approximation that show promising gains over standard separable wavelet transforms[3].

7.2.1 Graph representation of images

Digital images are 2-D regular signals, but they can also be viewed as graphs by connecting every pixel (node) in an image with its neighboring pixels (nodes) and by interpreting pixel values as the values of the graph-signal at each node. Graph representations of the regularly sampled signals have been shown to be promising in practice recently [35, 11]. In our experiments, we use an 8-connected representation G of an image as shown in Figure 7.6. In this representation, each pixel has two types of connections with its neighbors: (a) rectangular connections with NWSE neighbors, and (b) diagonal connections with its diagonal neighbors. Note that adding more directions to the graph, for example, by linking each pixel with its 2-hop neighbors, is possible but is not considered in our present work. In the 8-connected image graph G, separating out

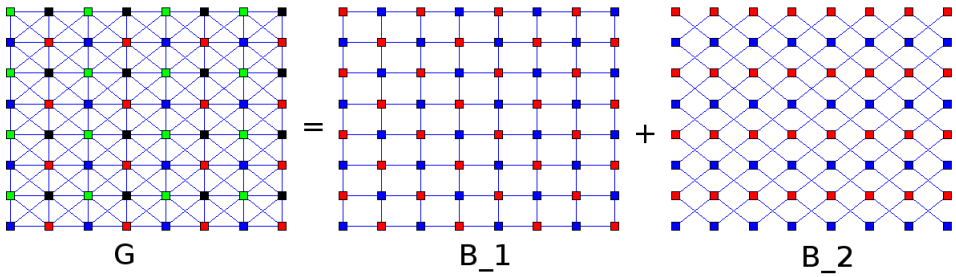


Figure 7.6: Two dimensional decomposition of 8-connected image-graph

rectangular and diagonal links into separate graphs leads to two bipartite subgraphs $\mathcal{B}_1$ and $\mathcal{B}_2$ as shown in Figure 7.6. The importance of each dimension can be changed by adjusting the weights

[2]The frequency of graph is defined in terms of eigenvalues of the normalized graph Laplacian matrix
[3]For results related to denoising, see [28].

of the links in each bipartite-subgraph. Given such a decomposition, we can implement a two-stage ("two-dimensional") graph-wavelet filterbank, as described in Chapter 5, where the filtering operations in the first dimension capture the variations along rectangular directions and those in the second dimension capture the variations along the diagonal directions. The overall wavelet filterbank has 4 output channels, and the downsampling pattern in each channel is identical to a downsampling-by-4 pattern for standard separable case. The nodes sampled in different channels are shown by different colors in the 8-connected graph G in Figure 7.6.

7.2.2 Graph Filter-banks on Images

The graph-based approach provides additional degrees of freedom (directions) to filter/downsample the image while still having a critically sampled output. To demonstrate this, we implement a graph wavelet filterbank on the 8-connected image-graph G of a given image, as shown in Figure 7.6. Here we assume the graph to be unweighted (i.e., all the links in the graph have equal weight). Figure 7.7 shows the one-level output wavelet coefficients of proposed 2-dim filterbank on a toy image which has both diagonal and rectangular edges. In this figure, the energy of wavelet

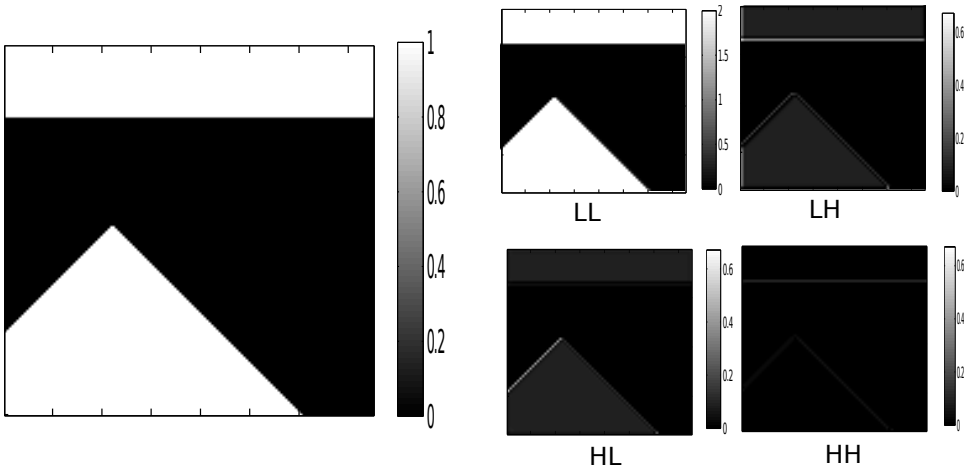
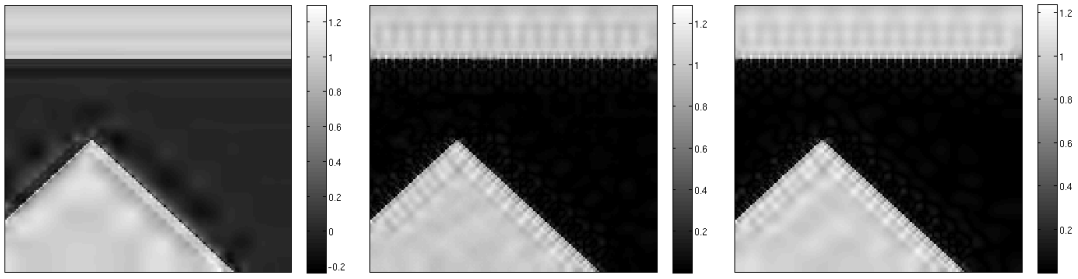


Figure 7.7: Separable two-dim two channel graph filterbank on a toy image with both rectangular and diagonal edges.

coefficients in the LH channel (low-pass on $\mathcal{B}_1$, high-pass on $\mathcal{B}_2$) is high around the rectangular edges, which is reasonable, since subgraph $\mathcal{B}_2$ is diagonally connected and its low-pass spectral frequencies are oriented along diagonal links. Similarly we observe that the high-energy wavelet coefficients in the HL channel (high-pass on $\mathcal{B}_1$, low-pass on $\mathcal{B}_2$) lie around the diagonal edges,

since $\mathcal{B}_1$ is rectangularly connected and its low-pass spectral frequencies are oriented towards horizontal and vertical directions. In Figure 7.8, we compare the proposed graph-based filterbanks with standard separable implementation with CDF 9/7 filters used in JPEG2000, on the binary image in the above example. We observe in Figure 7.8(a), that the reconstructed image, using standard CDF filters, has a lot of distortion near diagonal edges. This is because the wavelet filters in the standard case are only oriented in either horizontal or vertical direction. Therefore, they produce a lot of detail coefficients at the diagonal discontinuities. On the other hand, the reconstructed images in Figures 7.8(b) and 7.8(c), corresponding to graph-QMF filterbanks and graph-Bior filterbanks respectively, have less distortion (roughly 0.5dB in this case) than the standard case. This is because, the graph-based filters are oriented in both rectangular and diagonal directions, and therefore produce less detail coefficients in these directions. However, we also observe in Figures 7.8(b) and 7.8(c) that these graph-based filters also produce artifacts near the edges, which is because the filtering operations still cross edges in one direction (rectangular/diagonal in graph case) or the other. Therefore, filtering operations need to be made edge-aware. Note that in the graph-based formulation, more directions can be added to downsample/filter by increasing the connectivity of the pixels in the image-graph. Moreover, since graph-based transforms operate only over the links between nodes, the graph formulation is useful in designing edge-aware transforms (which avoid filtering across edges) by removing links between pixels across edges. We discuss the edge-aware representation of images in the next section.



(a) CDF 9/7 (PSNR = 64.16 dB) (b) graph-QMF (PSNR = 64.72 dB) (c) graph-Bior (PSNR = 64.63 dB)

Figure 7.8: Reconstruction of binary image shown in Figure 7.7, using only 4^{th} level LL-channel wavelet coefficients, using (a) 2-D separable CDF 9/7 filterbanks, (b) proposed graph-QMF filterbanks with filter length ($m = 28$), and (c) proposed graph-Bior filterbanks with filter length ($k_0 = 20$, $k_1 = 21$).

7.2.3 Edge-aware graph representations

Graph representation of images provide a simple way to accommodate the edge-information present in the images, by adjusting the weights of the pixels near the edges. In this approach, first the pixels at the edges (i.e., pixel whose intensities change sharply from their neighboring pixels), are detected using standard edge-detection algorithms. Subsequently, the links between each edge-pixel and its neighbors are tagged as either *regular* or *less-reliable*, depending on the difference between pixel intensities across the link being low or high, respectively. Then in the *edge-aware* graph representation, the *less-reliable* links are either completely removed or assigned a lower link-weight than the regular links. Similar constructions have been proposed in recent work [11, 21], but these constructions use lifting transforms and block transforms respectively, and do not use graph-filterbanks. In our proposed design, we choose to assign a lower link-weight for less-reliable links, as completely removing links around edge-pixels sometimes create isolated pixels (holes) in the graph, which do not participate in computing the wavelet transform, and thus need to be separately accounted for. Consequently, the graph-wavelet filters on the resulting weighted graph have most of their energy on one side of the edge, and produce less number of non-zero wavelet coefficients at the edges than in the case of unweighted image graphs. This is demonstrated by an example in Figures 7.9 and 7.10.

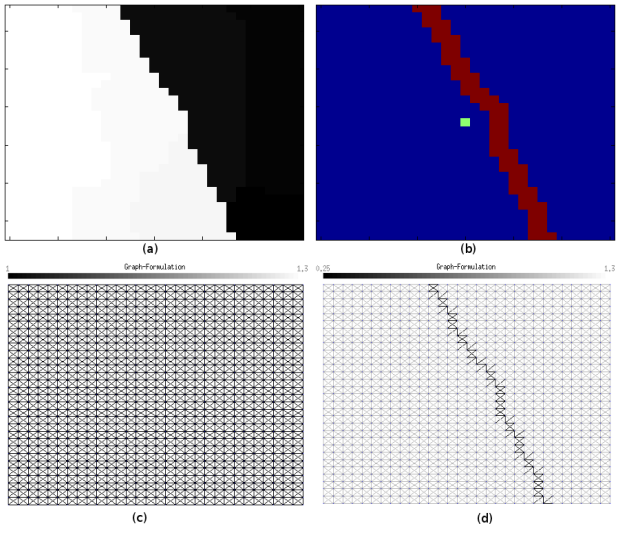


Figure 7.9: Example demonstrating importance of edge-weighted graph formulation of images: (a) input image (b) edge-information of the image and a highlighted pixel v, (c) unweighted 8-connected image-graph formulation (d) edgemap-weighted 8-connected image-graph formulation

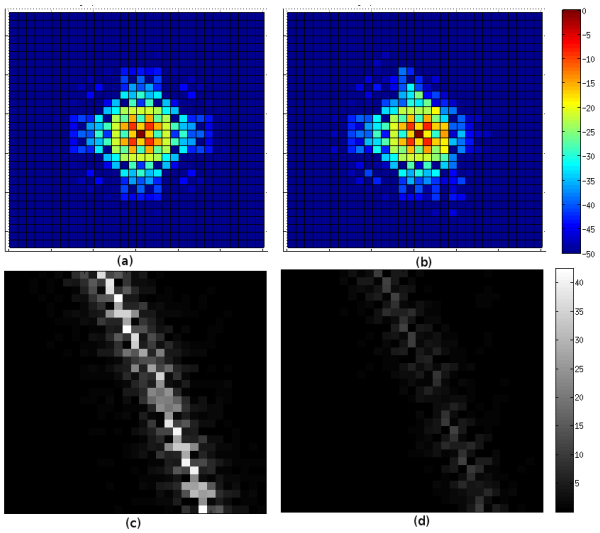


Figure 7.10: (a) HH wavelet filter (dB scale) on the pixel v on the unweighted graph (b) HH wavelet filter (dB scale) on the pixel v on the weighted graph, (c) undecimated HH band coefficients using unweighted graph and (d) undecimated HH band coefficients using edge-weighted graph.

7.2.4 Downsampling image graphs

Similar to standard wavelet transforms, the graph-wavelet filterbanks can be recursively applied at multiple levels, treating the LL channel output coefficients to be the new graph-signal, operating upon the downsampled graph constructing using the LL channel nodes only. In the proposed 8-connected image graph representation, since the LL channel nodes are uniformly sampled, the downsampled graph using LL nodes is made 8-connected by connecting each LL pixel to its neighboring 8 LL pixels. Further, the link weight between two neighbors in a given orientation (horizontal, vertical, diagonal, or off-diagonal) in the downsampled graph is equal to the weight of the path in the same orientation, between the two nodes in the original graph. Here the weight of the path is product of the weights of the links that it consists of. For example, the edge weight between two horizontal neighbors u and v in the downsampled graph, is the product of the weights of horizontal set of links connecting u and v in the original graph. The graphs obtained for 4 levels of decomposition for the Lena image are shown in Figure 7.11.

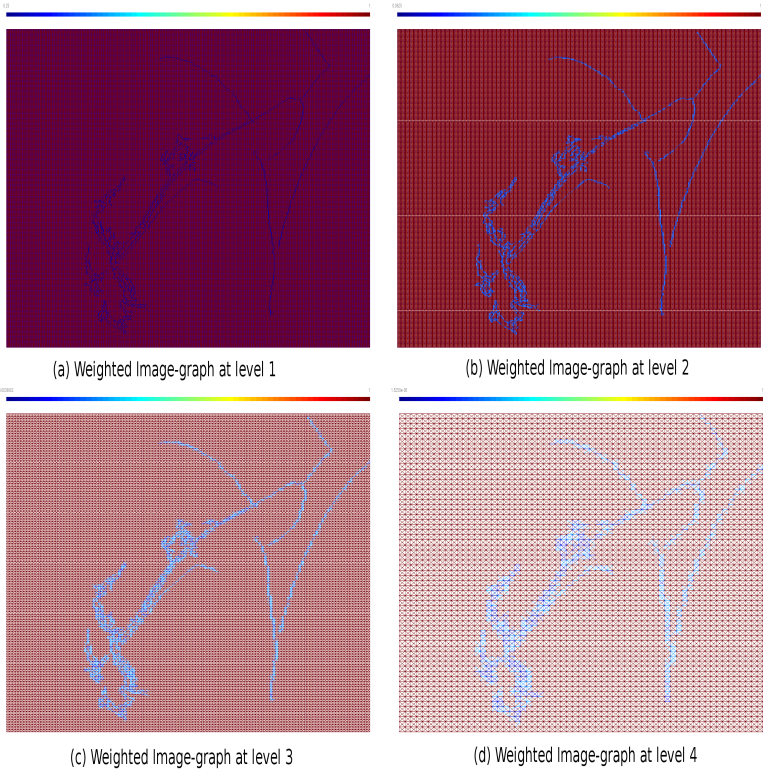


Figure 7.11: The weighted-graphs computed for Lena image, in 4 levels of decomposition

7.3 Experiments

In our experiments, we choose an undirected 8-connected representation of images as described in Section 7.2.1. For edge-detection in an image, we use standard Gaussian filtering followed by thresholding. In addition, we perform a connected component analysis to weed out small clusters of edge-pixels (of size less than 200), and dilate the remaining edges using a 2×2 structuring element to fill out the empty corners in the edges. For each edge-pixel the links between the pixel and its 8 neighbors are divided into two sets by applying a two-class clustering, based on the intensity difference. The links in the cluster with high intensity difference are declared *less-reliable* and their weights are adjusted to one-fourth of the weights of regular links (which is set to 1). The resulting graph has a binary weight distribution of links (regular/ less-reliable). The graphs in the subsequent levels of decomposition are generated from downsampling the first level graph as described in the Section 7.2.4, and have a more varied link-weight distribution. The graph at each level of decomposition is further decomposed into a rectangular-link only and a diagonal-link only bipartite graph as shown in Figure 7.6, and a two-stage two-channel graph-QMF filterbank is then

applied at each level. The filters in the filterbank are chosen to be polynomial approximations of graph-QMF filters in (5.28) with prototype kernel $h_0(\lambda)$ to be Meyer kernel in (5.33) with parameter $m = 30$ (for m^{th} order of approximation).

7.3.1 Image non-linear approximation

We now compare the proposed graph-based filterbanks with existing CDF 9/7 filters used in JPEG2000, using non-linear approximation with k-largest wavelet coefficients. Figure 7.12, shows PSNR and SSIM [49] values plotted against fraction of detail coefficients used in the reconstruction of Lena (512×512) image. It can be seen from both the plots that graph-QMF filterbanks achieve better compression than the standard CDF 9/7 filterbanks. This is because the graph-QMF filterbanks capture signal variation in more orientations than the separable case. Among the graph-based wavelet transforms, the edge-weighted formulations perform better than unweighted formulation. This makes sense, as the weighted graph-formulations of the image are edge-aware and produce fewer wavelet coefficients compared to unweighted graphs near the edges. However, the performance gain does not include additional edge-map information which can eclipse the gain. Recent work [35, 21] using transforms based on similar edge-map information have shown that this trade-off is favorable. Formulating the trade-off between extra performance gain using edge-weighted graphs and the edge-information needed, as an optimization problem constitutes part of our ongoing work. Figure 7.13 shows the reconstructed image with largest 1% detail coefficients in all described cases. It can be seen that perceptually the reconstructed images using both graph-QMF wavelets look sharper than the reconstruction with standard CDF 9/7 wavelet reconstruction. However, the reconstructions using the unweighted graph-QMF wavelets have ringing artifacts near some edges, which disappear when we use the edge-weighted graph formulation. Thus, the edge-weighted graphs and corresponding graph-wavelet filterbanks produce a sparser representation of edges, than the standard separable wavelets.

7.4 Summary

In this chapter, we have proposed some applications of our proposed spectral wavelet filterbanks. The first application is based on a multiresolution decomposition of arbitrary graphs, where graphs are downsampled and filtered into smaller graphs, with data on the graphs representing a (smooth

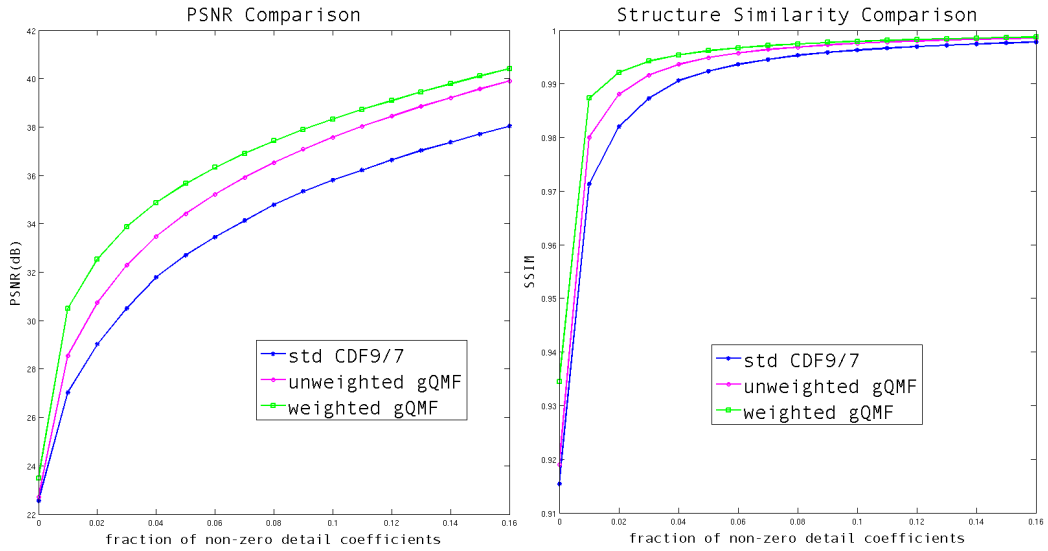


Figure 7.12: Performance comparison: non-linear approximation

or sharp) approximation of the original data. This method can be useful in graph compression and anomaly detection applications. Next we discussed, a novel method of processing 2D images using proposed graph-based wavelet filterbank design. We have proposed a graph representation of images in which pixels are connected with their neighbors to form undirected graphs. The graph formulation captures the geometric structure of the image by linking pixels in different directions and by adjusting the weights of the links near edges. Preliminary results show gains in the image non-linear approximation application over standard wavelet filterbank.

Figure 7.13: Reconstruction of "Lena.png" (512×512) from 1% detail coefficients

Chapter 8

Conclusions and Future Work

8.1 Main Contributions

In this thesis, we have proposed wavelet filterbanks for analyzing data defined on graphs. We termed the data on the vertices of graphs as *graph signals* and extended regular DSP techniques such as basis decomposition, filtering and downsampling to these graph-signals. The graphs in our research are undirected with no self loops or multiple edges. While many wavelet transform have been proposed in literature for graphs (see Chapter 2 for discussion), a common drawback of most of these designs is lack of critical sampling, which limits their applications in compression and denoising tasks.

Therefore, we have proposed critically sampled wavelet filterbanks for graphs. These filterbanks are implemented as "one-dimensional" two-channel filterbanks on bipartite graphs, and extended as "multi-dimensional" separable filterbanks on arbitrary graphs. We have found *bipartite graphs* to be the natural choice for implementing critically sampled two-channel wavelet filterbanks, because of their spatial and spectral properties. For lifting wavelet transforms, discussed in Chapter 3, we showed that any even-odd assignment strategy is equivalent to finding a *bipartite subgraph approximation* of the graph. Further, in Chapter 4, we showed that downsampling in bipartite graphs leads to a *spectral folding phenomenon*, which is analogous to *aliasing* in regular signals.

The spectral folding phenomenon allowed us to implement critically sampled spectral wavelet filterbanks on *any* bipartite graph, by computing filters which satisfy simple constraints (discussed in Chapter 5). In particular, we proposed wavelet filters, which are based on spectral kernels

and provided necessary and sufficient conditions for aliasing cancellation, perfect reconstruction and orthogonality in the resulting filterbanks. As a practical solution, we proposed *graph-QMF* designs for bipartite graphs which satisfy all the above mentioned properties. The exact orthogonal filterbanks are not compact support. They can however be realized as compact support filters by using Chebychev polynomial approximations at the cost of small reconstruction error and loss of orthogonality. As an alternative, we proposed *graph-Bior* filterbanks which are not orthogonal, but have compact support and provide perfect reconstruction. These filterbanks are critically sampled and invertible and offer a multi-level subband decomposition of graph-signals.

For arbitrary graphs we proposed three choices: a) approximate the graph G as a single bipartite graph $\mathcal{B}$, and implement the "one-dimensional" designs proposed in Chapter 5, (b) decompose the graph into K edge-disjoint bipartite subgraphs whose union is G via *bipartite subgraph decomposition*, discussed in Chapter 6, which leads to separable "multi-dimensional" filterbanks on graphs, and c) a combination of both (a) and (b) in which we find K edge-disjoint bipartite subgraphs, whose union is not exactly G, but very close to it. There are edge losses in approaches (a) and (c). However, the edge losses in (c) can be minimized by suitably choosing K-bipartite subgraphs. A comparison of our proposed design vis-a-vis existing transforms is shown in Table 8.1.

Method	DC response	CS	PR	Comp	OE	GS
Wang & Ramchandran [47]	non-zero	No	Yes	Yes	No	No
Crovella & Kolaczyk [7]	zero	No	No	Yes	No	No
Lifting Scheme [18, 38, 45]	zero for wavelet basis	Yes	Yes	Yes	No	Yes
Diffusion Wavelets [6]	zero for wavelet basis	No	Yes	Yes	Yes	No
Spectral Wavelets [14]	zero for wavelet basis	No	Yes	Yes	No	No
graph-QMF filterbanks (Sec: 5.3)	zero for wavelet basis[1]	Yes	Yes	No[2]	Yes	No
graph-Bior filterbanks (Sec: 5.5)	zero for wavelet basis[1]	Yes	Yes	Yes	No	No

Table 8.1: Evaluation of graph wavelet transforms. CS: Critical Sampling, PR: Perfect Reconstruction, Comp: compact support, OE: Orthogonal Expansion, GS: Requires Graph Simplification.

We applied lifting wavelet filterbanks using approach (a) in a *data-gathering application* in wireless sensor networks in Chapter 3. Here, we formulated the even-odd assignment problem as a *minimum dominating set* problem, which led to about 44% reduction in communication cost,

[1] When designed using asymmetric normalized Laplacian matrix.

[2] The exact Graph-QMF solutions are perfect reconstruction and orthogonal, but they are not compact support. Localization is achieved with a matrix polynomial approximation of the original filters, which incur some loss of orthogonality and reconstruction error, which can be arbitrarily reduced by increasing the degree of approximation.

compared to raw-transmissions, and about 10% reduction compared to state of the art tree-based lifting transforms.

Further, we implemented spectral wavelet filterbanks in Chapter 7 on graph representation of images, in which pixels are connected with their neighbors to form undirected graphs. The graph formulation captures the geometric structure of the image by linking pixels in different directions and by adjusting the weights of the links near edges. Preliminary results showed gains in the image non-linear approximation and denoising application over standard wavelet filterbank.

8.2 Future Work

The proposed wavelet filterbanks in this thesis can be operated on any arbitrary undirected graphs. The building blocks of our design are filterbanks on *bipartite graphs* which provide a "one-dimensional" analysis of graph-signals. The spectrum of bipartite graphs (using normalized Laplacian matrix) lies in the closed set [0 2], and has eigenvalues symmetrically placed on either side of $\lambda = 1$. Further, the eigenvectors of bipartite graphs, corresponding to any pair of symmetric eigenvalues, are identical to each other (except some sign changes). These properties are not found in any other graph. Therefore, one question which shapes our future direction is whether some form of aliasing occurs in graphs with higher chromaticity. More precisely, can the two-channel filterbank constraints discussed in Section 5.2, be extended to any non-bipartite graphs?

Moreover, the bipartite graph based designs themselves have lots of degrees of freedom. The choices to be made for implementing proposed filterbanks on any graph can be split into three major parts: (i) which bipartite subgraph decompositions to choose for a given graph, (ii) what filters designs to choose (orthogonal or biorthogonal, shorter or longer, spectral or non-spectral etc.), and (iii) how to compute graphs after downsampling, so that they are meaningful and approximate the properties of original graph. Our future work includes working on optimizing all these design choices. For deciding (i), we have proposed choosing bipartite subgraphs which provide mutually disjoint neighborhood sets at each node, which leads to *orthogonal filtering operations* on bipartite graphs. We also proposed two algorithms: Harary's decomposition and MCWMC decomposition, to compute bipartite subgraphs, which performed well on some of the graphs we studied. However, in some applications, other bipartite subgraph decompositions, such that those favoring more links in the low-pass channels, may be more favorable. Therefore, finding

optimal bipartite subgraph decomposition for any given application is what we plan to investigate in future.

In the edge-aware image processing application, preliminary results showed gains in the image non-linear approximation using proposed graph-based filterbanks over standard wavelet filterbanks. In the future, we would like to implement proposed graph-based filterbanks using $H.264$ encoders to better estimate the gains compared to the standard designs. Further, our future work includes studying a more heterogeneous distribution of link weights in the image-graphs and its impact on the graph-formulation.

References

[1] B. Aspvall and J. R. Gilbert. Graph coloring using eigenvalue decomposition. Technical report, Ithaca, NY, USA, 1983. 28, 93

[2] E. J. Cands and D. L. Donoho. Curvelets and curvilinear integrals. *J. of Approx. Theory*, 2001. 103

[3] Fan R. K. Chung. *Spectral Graph Theory (CBMS Regional Conf. Series in Math., No. 92)*. American Mathematical Society, February 1997. 2, 11, 30, 48

[4] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4:233–235, 1979. 32

[5] A. Cohen, I. Daubechies, and J.-C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 45(5):485–560, 1992. 54, 72

[6] R. Coifman and M. Maggioni. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, 21:53–94, 2006. 3, 19, 21, 114

[7] M. Crovella and E. Kolaczyk. Graph wavelets for spatial traffic analysis. In *INFOCOM 2003*, volume 3, pages 1848–1857, Mar 2003. 1, 3, 10, 17, 21, 114

[8] E. B. Davies, G. M. L. Gladwell, J. Leydold, and P. F. Stadler. Discrete nodal domain theorems. *Linear Algebra and its Applications*, 336(1-3):51 – 60, 2001. 11

[9] M.N. Do and M. Vetterli. The contourlet transform: an efficient directional multiresolution image representation. *Image Proc., IEEE Transactions on*, dec. 2005. 103

[10] D.L. Donoho. De-noising by soft-thresholding. *Information Theory, IEEE Trans. on*, 41(3):613–627, May 1995. 30

[11] E. M. Enriquez, F. D. Maria, and A. Ortega. Video encoder based on lifting transforms on graphs. In *Intl. conf. image proc. ICIP*. IEEE, Sep 2011. 104, 107

[12] S. Fitzpatrick and L. Meertens. An experimental assessment of a stochastic, anytime, decentralized, soft colourer for sparse graphs. In *In Proc. SAGA'01*, pages 49–64, 2001. 28, 29

[13] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proc Natl Acad Sci U S A*, 99(12):7821–7826, June 2002. 1, 2

[14] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, Mar 2011. 3, 12, 19, 21, 54, 55, 63, 97, 114

[15] F. Harary, D. Hsu, and Z. Miller. The biparticity of a graph. *Journal of Graph Theory*, 1(2):131–133, 1977. 91

[16] Shi J. and Malik J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997. 93

[17] Dmitry Jakobson, Stephen D. Miller, Igor Rivin, and Zev Rudnick. Eigenvalue spacings for regular graphs. In *IN IMA VOL. MATH. APPL*, pages 317–327. Springer, 1999. 45

[18] M. Jansen, G. P. Nason, and B. W. Silverman. Multiscale methods for data on graphs and irregular multidimensional situations. *Journal of the Royal Statistical Society*, 71(1):97125, 2009. 3, 18, 21, 23, 114

[19] I. M. Johnstone and B. W. Silverman. Wavelet threshold estimators for data with correlated noise. *Royal Statistical Society: Series B (Statistical Methodology)*, 59:319–351, 1997. 30

[20] D. Kempe and F. McSherry. A decentralized algorithm for spectral analysis. *ACM symposium on Theory of computing*, pages 561–568, 2004. 2

[21] W.S. Kim, S.K. Narang, and A. Ortega. Graph based transforms for depth video coding. In *in ICASSP'12*, Mar 2012. 107, 110

[22] Nick Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Applied and Computational Harmonic Analysis*, 10, 2001. 103

[23] W. Klotz. Graph coloring algorithms. *Mathematik-Bericht*, 5:1 – 9, 2002. 92

[24] R.I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proc. ICML*, pages 315–322, 2002. 2

[25] Tai Sing Lee. Image representation using 2d gabor wavelets. *Pattern Anal. and Mach. Intel., IEEE Trans. on*, 18(10):959 –971, oct 1996. 103

[26] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007. 2, 11, 30

[27] R. Mersereau and T. Speake. The processing of periodically sampled multidimensional signals. *ITASS*, 31(1):188 – 194, feb. 1983. 50

[28] S. K. Narang, Y. H. Chao, and A. Ortega. Graph-wavelet filterbanks for edge-aware image processing. *to appear SSP*, Aug. 2012. 97, 104

[29] S. K. Narang and A. Ortega. Local two-channel critically-sampled filter-banks on graphs. *ICIP*, pages 333 –336, Sep. 2010. 22, 55

[30] S.K. Narang and Ortega A. Perfect reconstruction two-channel wavelet filter-banks for graph structured data. *IEEE trans. on Signal Processing*, 60(6):2786–2799, June 2012. 93

[31] S.K. Narang, G. Shen, and A. Ortega. Unidirectional graph-based wavelet transforms for efficient data gathering in sensor networks. In *In Proc. of ICASSP'10*, March 2010. 22

[32] J. P.-Trufero, S.K. Narang, and A. Ortega. Distributed transforms for efficient data gathering in arbitrary networks. In *ICIP '10.*, pages 1829– 1832, Sept 2010. 22

[33] G. Pandey, M. Steinbach, R. Gupta, T. Garg, and V. Kumar. Association analysis-based transformations for protein interaction networks: a function prediction case study. In *KDD '07*, pages 540–549. ACM, 2007. 29

[34] E. Le Pennec and S. Mallat. Sparse geometric image representations with bandelets. *IEEE Trans. on Image Proc.*, 14(4), 2005. 103

[35] G. Shen, W.S. Kim, S.K. Narang, A. Ortega, J. Lee, and H.C. Wey. Edge-adaptive transforms for efficient depth map coding. In *Picture Coding Symposium (PCS), 2010*, Dec 2010. 104, 110

[36] G. Shen, S. K. Narang, and A. Ortega. Adaptive distributed transforms for irregularly sampled wireless sensor networks. *ICASSP '09*, 0:2225–2228, 2009. 22, 35

[37] G. Shen and A. Ortega. Compact image representation using wavelet lifting along arbitrary trees. *ICIP'08*, 2008. 103

[38] G. Shen and A. Ortega. Optimized distributed 2D transforms for irregularly sampled sensor network grids using wavelet lifting. In *ICASSP'08*, pages 2513–2516, April 2008. ix, 1, 3, 5, 18, 21, 22, 23, 24, 26, 30, 34, 35, 36, 114

[39] G. Shen and A. Ortega. Tree-based wavelets for image coding: Orthogonalization and tree selection. In *PCS'09*, Chicago, IL, May 2009. 22

[40] G. Shen and A. Ortega. Transform-based distributed data gathering. *Sig. Proc., IEEE Trans. on*, 58(7):3802 –3815, july 2010. 3, 5, 18

[41] G. Shen, S. Pattem, and A. Ortega. Energy-efficient graph-based wavelets for distributed coding in wireless sensor networks. *ICASSP' 09*, 0:2253–2256, 2009. 35

[42] TinyOS-2. Collection tree protocol. http://www.tinyos.net/tinyos-2.x/doc/. 34

[43] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P.L. Dragotti. Directionlets: Anisotropic multidirectional representation with separable filtering. *IEEE Trans. on Image Proc.*, 15(7), 2006. 103

[44] M. Vetterli and J. Kovačevic. *Wavelets and subband coding*. Prentice-Hall, Inc., NJ, USA, 1995. 2, 71

[45] R. Wagner, R. Baraniuk, S. Du, D.B. Johnson, and A. Cohen. An architecture for distributed wavelet analysis and processing in sensor networks. In *IPSN '06*, pages 243–253, April 2006. 3, 5, 18, 21, 22, 114

[46] A. Wang and A. Chandraksan. Energy-efficient DSPs for wireless sensor networks. *IEEE Signal Processing Magazine*, 19(4):68–78, July 2002. 36

[47] W. Wang and K. Ramchandran. Random multiresolution representations for arbitrary sensor network graphs. In *ICASSP*, volume 4, pages IV–IV, May 2006. 1, 3, 10, 16, 21, 114

[48] M. Weber and S. Kube. Robust perron cluster analysis for various applications in computational life science. In *CompLife*, pages 57–66, 2005. 1

[49] H. R. Sheikh Z. Wang, A. C. Bovik and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Proc.*, 13(4), 2004. 110

[50] W.W. Zachary. An information flow model for conflict and fission in small groups. *Journ. of Anthropological Research*, 33:452–473, 1977. ix, 29

[51] F. Zhang and E. R. Hancock. Graph spectral image smoothing using the heat kernel. *Pattern Recogn.*, 41(11), 2008. 2, 30, 103